Published in final edited form as:
*Nat Methods*. 2021 October ; 18(10): 1169–1180. doi:10.1038/s41592-021-01283-4.

# Differentiable Biology: Using Deep Learning for Biophysics-Based and Data-Driven Modeling of Molecular Mechanisms

**Mohammed AlQuraishi**[1,2], **Peter K. Sorger**[2]

[1]Department of Systems Biology, Columbia University, 622 West 168th St., New York, NY 10032

[2]Laboratory of Systems Pharmacology, Department of Systems Biology, Harvard Medical School, 200 Longwood Avenue, Boston, MA 02115

## Abstract

Deep learning using neural networks relies on a class of machine learnable models constructed using "differentiable programs." These programs can combine mathematical equations specific to a particular domain of natural science with general-purpose machine-learnable components trained on experimental data. Such programs are having a growing impact on molecular and cellular biology. In this Perspective, we describe an emerging "differentiable biology" in which phenomena ranging from the small and specific (e.g. one experimental assay) to the broad and complex (e.g. protein folding) can be modeled effectively and efficiently, often by exploiting knowledge about basic natural phenomena to overcome the limitations of sparse, incomplete, and noisy data. By distilling differentiable biology into a small set of conceptual primitives and illustrative vignettes, we show how it can help address long-standing challenges in integrating multi-modal data from diverse experiments across biological scales. This promises to benefit fields as diverse as biophysics and functional genomics.

Machine learning (ML) (and its applications in artificial intelligence; AI) has undergone dramatic changes over the past decade, led by rapid advances in differentiable programming. The development of better algorithms, introduction of more powerful computers, and increased data availability have allowed neural networks to transform one ML subfield after another. What began as deep learning has now evolved into a broader class of learnable models, often termed "*differentiable programs*" (where *differentiable* literally means functions with defined derivatives). Like traditional mathematical or computational models from physics and chemistry, differentiable programs can be specified in part using logic or equations particular to the problem under investigation. Unlike traditional models, which are commonly parameterized using a handful of fitted variables (often having physical meanings), differentiable programs can have millions or even billions of variables as part of their neural networks. Training involves learning these parameters. "End-to-end

Address correspondence to: Peter Sorger and Mohammed AlQuraishi, WAB 438, Harvard Medical School, 200 Longwood Avenue, Boston MA 02115, Tel: 617-432-6901, peter_sorger@hms.harvard.edu, m.alquraishi@columbia.edu.

Outside Activities
PKS is a member of the SAB or Board of Directors of Merrimack Pharmaceutical, Glencoe Software, Applied Biomath, RareCyte and NanoString and has equity in several of these companies; Sorger declares that none of these relationships are directly or indirectly related to the content of this manuscript. MAQ is a member of the SAB of FL2021-002, a Foresite Labs company, and consults for Interline Therapeutics.

differentiable” programs have the important feature that they can be jointly optimized from input to output to achieve a desired behavior; in current practice they frequently comprise neural networks but they need not do so.

The emergence of differentiable programming has been driven by four interrelated developments in basic science and software engineering: better pattern recognizers, the rise of bespoke (fit-to-purpose) modelling, joint optimization, and robust automatic differentiation frameworks and programming interfaces (*e.g.*, TensorFlow[1], PyTorch[2] and JAX[3]). In this Perspective, we discuss how ML models that incorporate biological and chemical knowledge have the potential to overcome limitations commonly encountered when modeling sparse, incomplete, and noisy experimental data. We distill the essential features of differentiable programming into conceptual primitives and use two illustrative vignettes to show how these primitives can be leveraged to solve a range of biologically significant problems. The vignettes cover the small and specific (*e.g.*, one experimental assay) and the broad and general (*e.g.*, protein folding). Taking stock of “differentiable biology”, we consider its implications for functional genomics and “multi-omic” biology, and how the long-standing challenges in data integration might be addressed by differentiable programming. We conclude with outstanding challenges and a description of new frontiers in ML, arguing that the full impact of end-to-end differentiable biology has yet to be realized.

## Four Key Developments

### Pattern Recognizers

The first and most visible development in differentiable programming is the emergence of powerful algorithms for processing (“recognizing”) sensory inputs such as images and speech. Deep neural networks are central to the success of these algorithms. Neural networks were first formulated decades ago, subjected subsequently to intensive study and then largely dismissed; their widespread adoption in the last decade represents a dramatic and important shift for the ML field[4]. Prior to the emergence of deep learning, which was made possible by faster processors and better optimization techniques, supervised ML relied heavily on convex functions, a simpler class of mathematical functions than neural networks. Convex functions have a single global minimum that is reachable by moving in the direction of descent irrespective of where the function is evaluated. The use of deep neural networks[4] has yielded a dramatic advance in the expressivity of ML algorithms: they need no longer be convex. When combined with large amounts of data, this has made it possible to generate hierarchical representations of data and capture the gestalt of vision and sound, in some cases with super-human performance[5].

Deep learning is sometimes dismissed in a biological context as mere pattern recognition[6] (a viewpoint challenged below) but pattern recognition has long been a holy grail of AI and is itself valuable in biomedicine[7]. Moreover, recent and rapid progress in pattern recognition is historically unprecedented: as recently as 2011 the classification error for image recognition (measured by the difference between labels such as cat, flower, *etc*. assigned by humans and machines using ImageNet[8] data) was >25%; by 2016 that figure had fallen to ~3%[8] (Figure 1). This is below the error for a single human trained to distinguish ImageNet classes and therefore represents super-human performance[8]. In speech recognition, the error rate for the Switchboard Corpus[9], long stalled at ~30%, fell to 5% in 2017, similar to human transcribers[10]. These well-known examples illustrate the degree to which pattern recognition has thoroughly changed our expectation of what is possible in AI, with biomedical applications in fields as diverse as ophthalmology[11-13] and digital pathology[14,15].

AI subfields that tackle cognition, such as symbolic systems, planning, and optimization have also benefited from advances in pattern recognition, in many cases bringing wider attention to under-appreciated advances: with the sudden advent of robust pattern recognizers, there is an opportunity to exploit an untapped reservoir of classical AI research. For example, the AlphaGo game playing agent[16] fused the classical AI approach of Monte Carlo Tree Search[17] with neural networks. Similarly, robotics, which has long been resistant to machine learning, is now benefiting from greatly improved vision systems[18]. Once classical AI systems are given the ability to process sensory inputs, they gain the ability to perform a variety of complex real-world tasks including relational reasoning[19].

## Bespoke Modelling

The application of ML to scientific problems has been accompanied by the emergence of *bespoke* ML models designed to capture salient aspects of a specific physical or chemical process. Bespoke models contrast with general purpose ML algorithms such as support vector machines[20] or random forests[21] by encoding prior knowledge about the structure of input data. Ideally, this is accomplished using general-purpose abstractions that deemphasize complex feature engineering in favor of architecture engineering. In feature engineering, transformations are made on primary inputs to extract information judged to be relevant based on past experience; such transformations often involve complex and potentially opaque data “pre-processing”. In architecture engineering the emphasis is on incorporating high-level aspects of the phenomenon being modeled in the design of the learning systems itself. This can involve fundamental concepts such as translational or rotational invariance of molecules in solution, the known chemistry of polypeptide backbones, and established relationships between data types of measurement methods.

A simple example is provided by convolutional networks[22] that assume translational equivariance in an image (Figure 2). Convolutional networks are effective in this setting because they reuse the same set of parameters, and hence the same pattern recognition components, across a receptive field. Bespoke models can also encode priors that facilitate the learning process, for example by including hierarchy in the organization of neural networks[23] to capture short and long-range interactions in the inputs. A model that generates image captions provides an early example[24] (circa 2015): visual and text-processing neural networks were paired so that one processes an image and the other a caption. During training, the joint model learned a latent representation that captured both the visual and textual aspects of inputs; the latent representation is subsequently used to generate new captions conditioned on unseen images. Bespoke models are now commonplace[25] and can be constructed using reusable templates and libraries of interoperable building blocks, creating a rich ecosystem for innovation.

### Joint Optimization of Differentiable Systems

By exclusively using mathematically differentiable functions as building blocks, differentiable models can be jointly optimized, from input to output, using backpropagation[22]. Joint optimization ensures that all model parameters are tuned simultaneously to optimize global behavior. Regardless of how complex a model becomes, the same backpropagation algorithm can be used. In contrast, earlier generations of ML algorithms typically involved many separate components, each of which needed to be optimized independently. Updating such algorithms is technically demanding and jointly optimized models are not only simpler, they are also more performant and robust. The power of end-to-end differentiable models has spurred renewed interest in first-order numerical optimization[26], a class of simple optimization techniques that have proven to be the most effective for deep models.[27] Their continued development and elucidation has led to resurgent interest in the theory of nonconvex optimization[28-30] as well as new techniques[31] that have made training deep differentiable models relatively routine.

### Automatic Differentiation Frameworks

A fourth advance in differentiable programming is the wide-spread availability of industrial-quality software tools known as *automatic differentiation* (autodiff) frameworks; these include TensorFlow[1],PyTorch[2] and JAX[3] among others. Using autodiff frameworks it is possible to use a few lines of code in a high-level programming interface such as Keras[32] (which runs on top of TensorFlow) to combine off-the-shelf neural network building blocks with custom mathematical transformations. Crucially, it is necessary to specify only the “forward” view of a model, *i.e.*, the direct relationship between inputs and outputs. Autodiff frameworks automatically compute the “backward” pass, which specifies how changes in parameter values affect model input-output relationships. These are precisely the derivatives used during training. Autodiff frameworks greatly simplify the practical task of model construction, making computation of derivatives a programming detail rather than the purview of a specialized branch of computer science. Modern autodiff frameworks come preloaded with a variety of ready-to-use neural network and mathematical operations, a suite of optimization algorithms for fitting parameters to data, and continuously updated documentation. The existence of a few widely used frameworks promotes model reuse and has democratized bespoke model building, making it possible for a large community of scientists to contribute. Democratization is potential cause for concern because subtle errors can still be made. Ideally, greater transparency in model formulation, which is enhanced by the use of high-level languages and common frameworks, will make it easier to document and error-check new applications of ML.

## Primitives for a Differentiable Biology

In the biological context, differentiable programming provides primitives for tackling three conceptually distinct classes of information: biological patterns, physical and phenomenological priors, and experimental and data acquisition priors. Priors constrain the space of possible models and enable use of smaller datasets. The most useful and defensible priors are those based on well-understood features of physical and chemical systems, such as the range of allowable bond angles in a polypeptide chain[33]. Priors can be interspersed with pattern recognizers that learn from data mappings that are too complex or poorly understood to be modeled explicitly.

## Biological Patterns

As a class, pattern recognizers are the most mature differentiable programming tools. They have been used to learn essential aspects the retinal fundus,[11] identify and segment cell boundaries in crowded environments such as tissues,[34] and predict new cell states from multiplexed immunofluorescence images[35]. The complexity of a pattern recognizer is often determined by the structure of the inputs. Images represented by 2D grids of pixels having fixed dimensions (e.g. images collected by conventional electronic cameras) are among the simplest inputs and they exhibit shift invariance, which can be used as a prior in model training via data augmentation.[36] “Images” need not be restricted to visual patterns. For example, intra-protein contact maps encoding residue co-evolution have been used as inputs to convolutional neural networks (CNNs) to predict protein structure[37-40]. Generalizing 2D grids to higher dimensions, *e.g.*, by discretizing 3D space into equal-sized cubes, has yielded pattern recognizers that can operate on high molecular weight macromolecules to predict protein functions[41,42] and the affinity of protein-drug complexes[43,44]. Some of the features learned by these models are human-interpretable but some are not, due to their size, complexity, or counter-intuitive nature, but neural networks are still able to learn them.

Variable-sized grids whose dimensions vary with the input data, such as one-dimensional grids comprising DNA sequences of varying length, represent another step up in complexity. For example, the patterns underpinning transcription factor binding motifs in DNA have long eluded a simple probabilistic code[45], but convolutional neural networks have modelled them with success[46,47]. Trees and other types of graphs, which can represent phylogenies, interaction networks and molecules, vary not only in length but in structure and can be learned using graph convolutional networks (GCNs).[48] GCNs have been used to learn mappings from molecules to protein-binding affinities[44] and to perform *in silico* chemical retrosynthesis[49]. In all of these cases, the key advantages of neural networks are their ability to recognize multi-way interactions occurring at both small and large scales.

While most contemporary ML applications focus on the relationship between complex inputs such as protein structure and simple outputs such as binding affinity, differentiable programming allows for richer input-output mappings. For example, we have developed an end to end differentiable “ recurrent geometric network” (RGN) that learns protein structure directly from sequence, taking a variable-length protein sequence as input and generating a variable-sized set of atomic coordinates as output[33]. More recently, AlphaFold2 developed by Google’s sister company DeepMind, uses an end-to-end differentiable system to predict single domain protein structures with accuracy approaching that of experimental methods such as crystallography (Figure 1).[50,51] The ability to generate complex outputs (*e.g.*, 3D folded proteins) from simple inputs (primary sequence) demonstrates one significant advantage of differentiable programs vis-à-vis conventional ML methods used for regression and classification. The latter are limited to narrow ranges of simple outputs types, most commonly categorical variables or real-valued scalars.

## Phenomenological Priors

ML research in biology increasingly incorporates prior knowledge about structure, chemistry, and evolution into differentiable programs (Figure 3a). Prior information can range in scope and generality from enumeration of genes or proteins and their interactions[52] to fundamental biophysics, including features of space itself.[53] For example, interactions within and between macromolecules are translationally and rotationally invariant and this can be formalized by generalizing CNNs from fixed grids (which have no guarantee of rotational invariance) to mathematical objects known as Lie groups[54] which capture rotational symmetry in three (or higher) dimensions. In modeling protein-protein interaction (PPI) networks, protein folding, and similarly complex biological phenomena, the incorporation of such priors makes it easier to capture distance-dependent physical interactions (*e.g.*, rotationally-invariant electrostatic forces). Recent progress in equivariant networks on Lie groups has been swift[55], including applications in molecular sciences[56] - most prominently protein folding - but the problem and attendant approaches remain far from solved. When such approaches become broadly deployable, they may prove to be as consequential for molecular systems as convolutional networks have been for image data: the analogy here is between shift invariance and rotational symmetry (in practice there are subtleties even within shift invariance, *e.g.*, local vs. global invariance[57]).

A valuable aspect of bespoke ML models in biomedical applications is that they can incorporate detailed information on the structural and chemical properties of macromolecules. For example, due to divergent and convergent evolution, many proteins utilize similar structural features for binding other biomolecules[58-60]. These features constitute the vocabulary of protein binding surfaces and, once learned, can be reused across domain families to increase predictive power. Phenomenological priors formalized mathematically force models to distill interactions across a protein family to a compact set of binding surfaces, or to prefer that evolutionarily-related proteins share binding partners. Pursuing this line of reasoning, we recently developed a model[61] for predicting the ligands of peptide-binding domains (PBDs) involved in signal transduction (*e.g.*, Src Homology 2 and 3 domains). We incorporated the concept of shared and reused binding surfaces by sharing energy potentials across PBD families, implicitly creating an energetic *lingua franca* for this type of macromolecular interaction. Energy potentials were learned, not prescribed (only the notion of reuse was assumed) and were found to improve model accuracy, particularly in data poor domains. Incidentally, our PBD-ligand interaction model[61] was fully differentiable but did not make use of neural networks. Models incorporating geometrically-aware neural networks and the concept of binding surface reuse are also showing promise[59,62]. A related approach, based on the simple idea that a protein's active site uses the same set of atoms to bind diverse small molecules resulted in substantial advances in predicting protein-ligand interactions[63].

When modeling biological networks, yet more specialized priors are possible. For example, joint modeling of transcriptional, proteomic, and phospho-proteomic time series data can be enhanced by imposing time separation between phospho-signaling and transcriptional regulation, as the former often occurs on a more rapid time scale than the latter, or by encouraging signaling cascades to terminate on a transcriptional change (which is implicitly another form of time-scale separation). Such high-level knowledge can be combined with molecular data on specific signaling pathways (*e.g.*, the structure of the MAPK kinase cascade) or transcription factor binding motifs (Figure 3b). In such a hypothetical model, the matrix of all possible protein-protein and protein-DNA interactions would be inferred, with some interacting pairs already pre-determined (*e.g.*, from the literature or focused experimentation), and some merely encouraged or discouraged based on knowledge of the archetypical interactions they represent.

## Data Priors

Most modeling in biology involves analysis of incomplete, noisy, and heterogeneous data. Incorporating priors that account for the data generation process are needed to minimize the effects of error and fuse disparate data types. Data normalization is another process that is *ad hoc* to a problematic degree. Data process invariably include adjustable parameters that are fit heuristically, typically one step at a time. Differentiable programming offers a fundamentally different approach: adjustable parameters can be optimized within a broader problem framework that involves evaluation of a hypothesis or prediction of outcome (*e.g.*, cell state). Parameters of both the experimental and computational aspects of the model can then be jointly fit to maximize predictive power. Few examples of such joint learning have appeared, but pre-processing steps for microscopy-based imaging (*e.g.*, image segmentation[64] and classification[65]) already incorporate learning elements. This is not a "glamorous" application of ML but it will prove to be one of the more consequential areas for differentiable programming if it can make y the connection between data and models more accurate, robust and informative.

Random error is present in all real-world data and most molecular measurements are also subject to poorly understood systematic error. Physics-based error modeling is common in structural biology and high-resolution optical microscopy, domains in which enough is known about the measurement process and the range of expected physical phenomena that many types of uncertainty can be quantified and modelled. While this approach is in principle transferable to other biological assays,[66,67] sophisticated error models are relatively rare in biomedical research, usually because the underlying physical processes are not sufficiently understood. In this case, simple parametrizations of the error may be possible, *e.g.*, when normalizing high-throughput RNA-seq measurements. However, the approach can be extended to describing the physical processes underpinning experimental assays. This is important when the biophysical quantity being sought, such as a disassociation constant, derives from an indirect measurement. For example, experimental characterization of protein-protein affinity involves a variety of analytical methods that measure different physical parameters (*e.g.*, on or off rates, equilibrium binding, heat emitted or required for binding, inhibition of activity, competition between substrates, *etc.*) Both simple and complex equations exist to describe the relationship between experimental observables and underlying biophysical parameters and these equations can be incorporated into differentiable programs. Backpropagating through these equations during optimization makes it possible to estimate unknown parameters in a robust manner, because the optimization is jointly accounting for all aspects of the model. Even when simple analytical formulas are unavailable, recent progress in incorporating ordinary differential equation solvers within neural networks[68] suggests the feasibility of encoding elementary mass action rate laws as differential equations within differentiable programs[69].

## Illustrative Vignettes

To illustrate the concepts described above we consider two situations in which the building blocks of end-to-end differentiable biology are combined to address complex and significant research questions. The first vignette illustrates how prior biological knowledge can be reflected in the architecture of a bespoke ML model; the second focuses on data homogenization in the context of protein-protein interactions.

### Protein Structure Prediction

The goal of protein structure prediction is to construct models that maps protein sequence (a variable-length string of discrete symbols) to the tertiary structure of the protein (variable-length sequence of 3D coordinates). Recent ML-based approaches make use of both phenomenological priors and pattern recognizers. In principle, off-the-shelf pattern recognizers such as recurrent neural networks can perform this mapping but in practice, achieving high performance has required leveraging features of protein geometry learned from 70 years of solving and analyzing protein structures (Figure 4a).[33,51,70] For example, the knowledge that protein backbones are covalently bonded polypeptide chains with nearly fixed bond lengths and angles but sequence-dependent torsion angles[71].

There exists a one-to-one mapping between torsion angles and 3D coordinates using known (differentiable) mathematical transformations[72]. To predict a 3D structure, it is sufficient to predict torsion angles from the amino acid sequence and optimize model parameters to maximize agreement between predicted and known angles. Fixing bond lengths and angles is a seemingly simple addition to an ML model but it has an important effect on learning efficiency and accuracy and it also helps ensure that local protein geometry is almost always correct.

Unfortunately, a purely local approach of this type—trained and judged exclusively by the accuracy of predicted torsion angles—performs poorly because minor local errors accumulate to generate large errors at the level of a complete protein. A better approach involves conversion of local torsion angles to 3D protein coordinates as part of the modelling process itself using parameters that maximize agreement between predicted and known coordinates from the Protein Data Bank. Here too we are faced with a choice. The simplest loss function penalizes deviations between predicted and known coordinates, *e.g.*, by averaging the sum of their differences. However, this is not translationally or rotationally invariant. A loss function that is defined in terms of distances, for example one that averages the sum of differences of inter-atom distances between predicted and known structures, circumvents this problem.

Designing a custom loss function also permits more sophisticated treatments of protein structure data, which frequently suffers from missing (disordered) side-chain atoms and stretches of sequence, in large part because unstructured domains are integral to protein function. Eliminating such proteins from consideration reduces the amount of training data by up to 50% and may also bias it. A custom loss function can ignore unresolved atoms or residues, penalizing only well-resolved parts of structures, making available for training all but a dozen of the ~100,000 unique structures in the Protein Data Bank[73] (Figure 4a).[33] The approach can be adapted to predict structures from individual protein sequences, without using any explicit co-evolutionary information, a valuable capability for protein design.[53]

More generally, the use of chemical and geometric that has historically been a key advantage of physics-based modeling is now available in a data-driven, learnable setting. This was leveraged in AlphaFold2[51] which uses specialized forms of (Transformer-based) attention[74,75] to reason over multiple protein sequence alignments, thereby implicitly learning the idea of co-evolution and perhaps also phylogeny.[76] Transformers have the ability to learn from local and remote features, a powerful capability in structure prediction in which residues distant in the primary sequence interact directly in the folded protein to stabilize it. AlphaFold2 also refines protein structures in 3D space in a rotationally and translationally-invariant matter, leveraging recent efforts in differentiable programming to add features of physical space as constraints (Figure 4b). This has led to the speculation that attention and symmetry are essential features of AlphaFold2, explaining its remarkable performance.[50] We anticipate that it will be possible to add other aspects of protein chemistry in future years, further increasing the performance of ML algorithms while also making them more interpretable.

## Homogenizing Protein-Protein Interaction Data

We illustrate the use of differentiable programs for data fusion with a model that learns protein-protein interaction (PPI) affinities from diverse types of experimental data collected using different methods at different times (Figure 5). The sheer diversity of PPIs, particularly those involving peptide-binding domains such PDZ domains[77] and their peptidic ligands, makes it highly unlikely that more than a small fraction of affinities can all be ascertained experimentally. Instead we must work from distinct types of information (we discuss four below), each incomplete and involving a different measurement method. “Quantitative binding data” involve direct measurement using quantitative biophysical methods such as surface plasmon resonance (SPR), isothermal titration calorimetry (ITC), or peptide arrays. These are the most quantitative but limited types of data. A second source of data are acquired from cell extracts using affinity purification mass spectrometry[78,79] (pull-down assays) or *in vivo* using yeast two-hybrid assays[80] (“high throughput binding data”). These data cover an increasing proportion of the proteome but are neither direct (two proteins that pull down together can both bind a third protein) nor particularly quantitative. A third type of data involves functional interactions inferred from genetic or over-expression studies[81]; these data are numerous but the least direct. A fourth type of data comprises high-resolution co-complex structures of individual PPIs to use for training or validation of biophysical hypotheses inferred by ML.

Our goal is to use the entirety of available data to create a model that predicts equilibrium association constants ($K_a$) of as many PPIs as possible using a differentiable model with two components: (i) an energy function that maps inputs (pairs of protein sequences) to predicted $K_a$ and (ii) a set of data homogenizers that map raw experimental data to a form that can be compared to predicted $K_a$ values; the latter is relevant to the current discussion on data fusion. We start with a key assumption: namely that the number of distinct experimental conditions or assay types is much smaller than the number of individual measurements and that we know which assay gave rise to each data point. We begin by homogenizing quantitative binding data from methods for which the same set of equations maps a measured value to $K_a$; we allow for variation in scaling and other parameters by permitting the mapping equations to have different and unknown parameter values. The entire model is then jointly optimized so that parameters in the data homogenizers maximize the performance of the energy function (used to predict PPI affinities). A more complex version of this scenario, in which experiments are different but still describable by known equations (*e.g.*, ITC vs. SPR) are handled with assay-specific transformation equations.

Homogenization is more difficult when the relationship between biophysical parameters and measured features are not describable in equations known *a priori* (*e.g.*, if it is unknown whether binding data involve competition between substrates). If we assume that at least one member of a family of equations can map raw values to reported affinities, the homogenizer can be constructed to draw from the family of equations as needed, for example by formalizing the mapping as a convex mixture of equations whose weights are penalized using a sparsemax[82] penalty. Alternatively, a general purpose map such as a Gaussian process[83] can be used with range and monotonicity constraints that reflect universal aspects of protein-protein binding biophysics without assuming a specific functional form for the conversion of measured values to binding constants.

In the foregoing examples, the model can be optimized by penalizing deviation between predicted and data-derived $K_a$ values on a continuous landscape of parameter values because the data-derived quantities have precise values. This is not true for binary interactions obtained from high-throughput binding data. These data typically involve a call of binding or not binding based on a statistical test. It is reasonable to assume that positive PPIs have a $K_a$ higher than some presumed but unknown threshold, but the absence of interaction may or may not be interpretable in biophysical terms (it might reflect either a true absence of binding or a technical error in the measurement). Direct comparison between predicted $K_a$ values and binary data is therefore not possible, but binary data can be used by estimating detection thresholds. To accomplish this we define a loss function that penalizes predictions in which the $K_a$ is below the positive threshold for positive PPIs and when applicable, above the negative threshold for negative PPIs; in other cases, no penalty is imposed. This results in a one-sided loss function for binary (high-throughput) data and a two-sided loss for quantitative data. If the detection thresholds are unknown but can be assumed to be fixed within any given assay or experimental condition, they can be treated as parameters and fit jointly with the rest of the model.

The last class of data to consider involves indirect interactions, which in extant PPI databases covers a large variety of possible assays including epistatic genetic interactions, patterns of gene or protein co-expression across tissue or cell types, and evolutionary conservation. While a positive score does not demonstrate physical interaction between proteins, it is positively correlated with it. We formulate this transformation process as a binary assessment of interaction with a single unknown positive threshold. Predicting low affinities for indirect PPIs is not penalized, but predicting high affinities is encouraged. The (learned) detection threshold depends on the strength of the statistical correlation and is therefore correct only in expectation, not for individual measurements. If the correlation is high however, the model may be able to extract meaningful information. On the other hand, if the correlation is low, the learned threshold will get set to a very high number, in which case no information is extracted and the model is unaffected.

A key feature of the approaches described above is that parameters of the data homogenizer are learned jointly with those of the energy function. This runs the danger of unwanted interaction between the parameters used to homogenize data and those being learned by the energy function, leading to degenerate solutions in which performance is misleadingly high simply because all homogenization parameters are set to zero. To avoid such problems, constraints must be imposed on the data homogenizer: parameters must be non-zero and fall within meaningful ranges based on prior knowledge (for example single-domain PPI affinities will fall in the nanomolar to micromolar range). More generally, a hierarchical approach to learning the two sets of parameters (sometimes called “meta learning”) is usually the most robust. In this case, an inner backpropagation loop fits parameters of the energy potential to a training set while an outer backpropagation loop fits parameters of the data homogenizers to a validation set, and the entirety of the process is assessed through a second validation set. Such metaparameter fitting has been successfully used in other applications[84,85]. Under these circumstances two validation sets are needed to avoid overfitting. As is true in all ML applications, careful selection of training, test, and validation datasets is important.

## Differentiable Programming as a Framework for Functional Genomics

The last decade has seen a dramatic increase large datasets characterizing whole genomes, transcriptomes, metabolomes, and proteomes and new technologies are extending this to differences in space (*e.g.*, spatial transcriptomics)[86] and time[87]. The analysis of such functional genomic data aims to better understand cell signaling, development and disease but dramatic increases in instrumentation and data have not, in general, been matched by commensurate increases in our understanding of biological systems. This gap stems in part from the lack of analytical frameworks to integrate different data types, make use of complex and irregular prior knowledge, and extract useful insights from the resulting mixture. Differentiable programming promises to provide much needed new tools.

### Integrating Multi-Omic Data

By definition, multi-omic biology involves multiple types of data, typically spanning different classes of biological entities (*e.g.*, mRNA and protein abundance) and different ways of measuring the same entity (*e.g.*, RNA sequencing vs. microarrays). In principle, such data provide rich and complementary views of a biological system but they are typically related in complex ways (*e.g.*, mRNA is used to generate proteins but the levels of the two are not well correlated)[88] and involve incongruous quantities that are not directly comparable (*e.g.*, protein expression vs. gene amplification). When tackling such data, off-the-shelf ML methods face three major obstacles that cannot be simultaneously resolved:

generation of interpretable models, capture of epistatic or multi-step causal effects, and incorporation of prior knowledge. For example, the most interpretable models for identifying the molecular origins of differential drug responses in cell lines [89,90] are linear regressions mapping multiple input features, *e.g.*, gene expression levels to drug response. The relative weighting of input features is often used to score a gene's importance[91] but what does it mean when the weight associated with a mutation in one gene is twice as large that associated with the expression level of a second gene? Biological mechanism is necessarily obfuscated in such approaches in an attempt to achieve data fusion. Furthermore, linear models do not capture epistasis in the input space, such as the combined effects of having a mutant allele and a change in expression of the same gene. Off-the-shelf nonlinear models, such as deep neural networks, make a different tradeoff. By virtue of their nonlinearity, they can capture complex epistatic effects but they are less interpretable: it is difficult to know what a weighted mixture of gene expression and allelic variants means in a mechanistic sense, and even less so when this mixture has been non-linearly transformed. Finally, in both linear and nonlinear models, the use of prior knowledge is largely restricted to feature engineering (or feature selection), for example using gene set enrichment analysis (GSEA), in which genes previously observed to co-vary are used to reduce the dimensionality of a dataset involving many genes or proteins into a smaller set of grouped features.[91] Unfortunately, this type of feature engineering cannot capture the richness and nuance of accumulated biological knowledge and it is itself often hard to understand.

One way to combine different data types while incorporating prior knowledge, and – ideally – revealing causality is to use mechanistic models such as systems of ordinary differential equations (ODEs) and related formalisms grounded in chemical reaction theory. A key advantage of these models is that they are both interpretable and executable.[92] However, while conventional ML-based approaches are in a sense too flexible, since they allow *ad hoc* integration of disparate quantities, ODE-type models suffer from the opposite problem of being time consuming to create and have difficulty incorporating diverse types of data without careful pre-processing into a common set of units (*e.g.*, molecules per cell). We propose that differentiable programming serve as a bridge between the ML and ODE paradigms. Differentiable programs have the ability to incorporate mechanistic models, including ODE models based on reaction theory, with black box pattern recognizers: ODE solvers can themselves be made differentiable and optimized through within existing autodiff frameworks[68]. In fact, development of new ODE-based primitives is a very active area of ML research[93]. Pattern recognizers can also be used to tackle aspects of a problem that are too complex to be modelled mechanistically, or whose details are irrelevant to the mechanistic question under study, while maintaining a mechanistic approach to the salient aspects of a problem. Integration of ML and ODE frameworks allows for joint optimization of all aspects of the model by backpropagating through the dynamic simulation itself. For example, training an ML model to predict protein stability and binding affinity based on genetic variation would conventionally be done separately from optimization of an ODE model whose parameters are products of the former. Training of a joint model can also capture emergent properties of the system not explicitly encoded in its parameters[69].

The ability to mix and match model components based on different mathematical formalisms facilitates integration of prior knowledge in a far deeper way than traditional feature engineering, which largely involves pre-processing of input data. Differentiable programming enables knowledge incorporation at multiple levels of abstraction. At the most basic level, well established properties about biological systems can be “hard coded” into differentiable programs, for example known transcription factor binding sites in a model of transcriptional regulation. These models can be parameterized in a way such that they are partially learned, *e.g.*, for transcription factors with unknown motifs, and partially fixed based on biological data[94]. A prior might also impose time-scale separation between regulatory factors, model discrete events in time with recurrent architectures,[95] or sample rate parameters from a probability distribution based on known enzymology[96]. Facilitating this, so-called probabilistic programming extensions have emerged for essentially all major autodiff frameworks[97,98], allowing probabilistic primitives to be stochastically backpropagated through during model optimization. Finally, prior knowledge can be reflected in the learning process, such as “curriculum learning” techniques in which easier to fit data points are presented early in training to learn simpler patterns before more complex data are learned compositionally.

## Integrating Irregularly Shaped Data

A parallel challenge is the irregular nature (incompleteness) of most biological data. One common scenario involves joint learning from data sets that were generated independently with a focus on similar but not identical questions. In this scenario “inputs” are likely to vary with the data set, for example the subset of genes whose expression levels have been measured, as will the “outputs” —*e.g.*, $IC_{50}$ or fraction of cells killed for the effects of drug perturbation.[99] One approach to combining such data is to consider only quantities and measurement methods that are common to all data sets, but this results in loss of usable data. Implicitly, a tradeoff is being made between the number of data points retained and the richness and diversity of measurements being utilized for each data point. Differentiable programming can help integrate irregularly shaped and partially overlapping data sets despite limitations in data imputation techniques[100] through the use of learned latent spaces. In this approach raw input features are mapped to rich internal latent spaces with the ability to extract complex multi-way interactions (*i.e.*, epistatic effects) between raw input features.

For regularly shaped data a single latent space is typically used. With irregularly shaped data, the key concept is creation of a compositional latent space made up of subspaces shared between data sets. For example, if two data sets measure the levels of partially overlapping sets of proteins, three latent spaces can be learned; one for the shared proteins and two for the proteins unique to each data set. Predictions based on inputs present in either data set can then be based on a composite latent space formed by stitching together the shared latent space with a data set-specific one. When multiple data sets are used, with more complex overlapping patterns, correspondingly more complex arrangements of latent spaces can be constructed. In this way, information sharing is enabled at a granular level; latent representations of input features common to all data sets are learned using all data sets, maximizing data utilization, while representations of input features specific to individual data sets are only used in making predictions for these data sets, maximizing the breadth of input feature utilization. On the output side, missing data can be generally treated as described in the protein structure vignette: for any given data point, custom loss functions are constructed to ignore contributions that arise from components of the output that are unavailable, while retaining contributions that arise from available components. This approach also serves to maximize information utilization at the level of model output or prediction.

## Recent and Upcoming Developments

### Self-supervised Representation Learning

Automatic learning of useful representations of input data is a defining characteristic of deep learning and marks the shift from feature engineering to architecture engineering. Most representation learning is done implicitly within a supervised learning framework in which the prediction task drives pattern recognizers to identify aspects of the input most relevant for accurate prediction. This requires inputs and labelled outputs (corresponding to the prediction task), which can be difficult to obtain. Self-supervised learning is an alternative approach for learning that does not require labelled outputs, unlocking the potential of very large unlabeled data[101]. It relies on artificial learning tasks, for example the imputation of randomly masked regions in the input, which, when combined with the information bottleneck present in learnable models having limited parameters, induces models to be compressive, *i.e.*, to identify recurrent input patterns that efficiently capture the data. We and others have used self-supervised learning to induce representations of protein sequence from very large sequence databases[102-106], and once learned, applied them to unrelated downstream tasks, including protein design[107] and prediction of protein structure[53,108] and function[109-111]. Similarly, learning representations for sets of homologous proteins found in multiple sequence alignments[112] was used as an auxiliary learning task in AlphaFold2[51]. Beyond protein sequences, the structures of organic chemical compounds represent a massive source of unlabeled data[113], and self-supervised learning approaches are now being applied in chemical applications[114-118], including protein-ligand binding.

### Generators

Generative models, the most prominent of which are generative adversarial networks[119] variational autoencoders[120], normalizing flows[121], and diffusion models[122], use neural networks to learn a generative model of data: that is, a model able to generate new data points. These approaches have garnered headlines due to the ultra-realistic quality of the images they can synthesize for use in computer-animated film and deepfakes.[123] Such models have also found applications in biology and chemistry, including generation of new molecules[124-126] and macromolecular sequences[127-129], and imputation of RNA-seq data[130]. A key feature of generative methods is the ability to capture higher-order correlations within individual samples, for example the consistency of gender or skin color across a face, overcoming the problem of blurry samples caused by implicit averaging in maximum likelihood models. This capability can be useful for biological systems with missing or unknown elements, potentially including biological networks, for which it is desirable to generate specific instantiations of a network (*e.g.*, physiologically relevant combinations of correlated protein concentrations) as opposed to averaged network states that do not reflect the ground truth of any individual cell or cell type.

## Simulators

Simulators can be used to generate synthetic data as a substitute for real data when training ML models and can even be integrated within differentiable programs as fixed or learnable components. Simulators also find application in testing models, by direct analogy with other types of synthetic data. Examples of the first approach include realistic simulated driving environments (*e.g.*, games) for training autonomous vehicles[131], and highly accurate quantum chemistry models, such as coupled clusters and density functional theory[132], used to generate synthetic data for training neural network-based force fields[133,134]. Such models can predict molecular properties at nearly the same accuracy as the high-level theories but with orders of magnitude less computation time[135]. While differences between real and synthetic data can be difficult to bridge, as no simulation can perfectly replicate reality, there have been many impressive successes, particularly in robotics[136].

Examples of the second approach include so-called inverse graphics applications in which simulated 3D worlds are visualized using a differentiable rendering engine and augmented with neural networks that learn a mapping from rendered 2D images back to the underlying 3D structures[137]. In protein folding, the method of contrastive divergence[138] has been used to fold small proteins using a (non-learnable) molecular dynamics simulator coupled to a learnable forcefield[139], and more recently, differentiable and learnable Langevin dynamics simulators have been coupled to a learnable energy-based model to do the same[74,140,141]. The incorporation of simulations within a learning framework enables the inference of much richer objects, including for example the trajectory of protein motion or the kinetics of molecular binding events.

## Probabilistic Programs

Bayesian models are particularly useful for capturing uncertainty; they include Bayesian nonparametric[142] approaches that capture models with a potentially infinite number of parameters. With the rise of deep learning, hybrid approaches combining neural networks with Bayesian modeling have proliferated[143]. Moreover, the advent of autodiff frameworks with probabilistic programming capabilities such as TensorFlow Probability[97] and Pyro[98] have made it easier to build bespoke probabilistic models. Such frameworks combine the best of probabilistic programming, *i.e.*, a concise way of constructing Bayesian models having complex interdependencies between random variables, with modern neural networks, using stochastic generalizations of automatic differentiation. Such models will likely play an important role in capturing uncertainty and causality of measurements and natural phenomena in the future.

## Non-Differentiable Learning

This perspective focuses on end-to-end differentiable modelling in biology because differentiable programming has shown the greatest promise in building bespoke models of complex natural phenomena. However, just as convexity ultimately proved to be an overly limiting constraint on learnable mathematical models, it may well be that differentiability will prove to be similarly limiting, especially in applications involving discrete reasoning. Many old and new approaches to learning go beyond differentiable programming, including ones dating back to the genesis of the AI field. One of the most promising current directions is reinforcement learning[144], which involves learning agents that perform actions in real or simulated environments. Training of such agents often, but not always, involves discrete non-differentiable actions. Reinforcement learning, when combined with differentiable pattern recognizers, has proven effective in simulated environments such as game playing[145,146], as well as in burgeoning applications in chemistry such as organic synthesis[49,147] and RNA design[148]. We expect that as these methods mature and migrate to real-world applications, their importance in the life sciences will continue to grow.

## Obstacles and Opportunities

Several challenges remain before end-to-end differentiable programming will be broadly adopted in biomedicine. First, the steps that comprise the “art” of modeling, from formulating a problem, mathematically encoding the correct priors, building pattern recognizers, and selecting training, test and validation data do not yet permit automation. The need for scientific intuition and data wrangling remain unchanged. Second, many entities in biological systems, including molecules, networks, and dynamical reactions, are structurally richer than the data types used in most contemporary ML research, particularly in large corporations, resulting in a need for more algorithmic development of differentiable programming frameworks. Data availability is a challenge as are labelled datasets for supervised learning. However, deep learning can be as data efficient as so-called shallow approaches[149] and bespoke differentiable models are more, rather than less, data efficient than conventional ML models because they use prior knowledge to extract more information from data. Mathematical machinery for representing complex input modalities such as graph neural networks[150,151] also makes more data types available for learning.

The requirement that all model components be made differentiable can be difficult to reconcile with the fundamentally discrete nature of biological entities such as DNA and protein sequence, particularly in a generative context in which algorithms are tasked with designing new biological constructs. Recently, ML researchers have made major strides in inventing differentiable versions of discrete concepts including sentences[152] and mathematical statements[153], computational data structures such as stacks, queues, and lists[154], computational processes such as sorting[155], recursion[156], and arithmetic logic units[157] as well as physical objects such as molecules[158]. Further development of differentiable primitives is a general challenge in ML with major implications for natural science.

As computing demands continue to grow, in some cases exponentially[159] a large gap has appeared between resources available to academic and industrial labs. For academia to remain meaningfully competitive in ML, particularly with large-scale models, renewed national or multi-institutional investments in computing power and software engineering must be made, by direct analogy with the efforts that have made traditional supercomputing widely available. We believe that this is a worthwhile investment for governments and corporations because academic research remains essential in undergraduate and graduate education and is published and available for reproduction and improvement. There is no guarantee that this will also be true of industrial research.

There is legitimate concern that ML models can be difficult to understand and this has given rise to calls for "intelligible" or "interpretable" ML models and systems.[160]. Interpretability in the context of models typically revolves around our ability to understand the relationship between inputs and outputs.[161] A problem arises because presence of large numbers of learned parameters in differentiable programs obscures their meaning. In some clinical applications this might not be an issue, but the goal of most scientific research is not just to predict outcomes from a set of inputs but to generalize conclusions in terms of physical and chemical principles. However, the incorporation of physical principles in differentiable models not only improves performance, it also increases intelligibility.

In conclusion, differentiable programming and other forms of deep learning are growing rapidly in sophistication and scope and they promise to accelerate research in multiple areas of experimental and molecular biology – not just protein structure prediction. Bespoke differentiable models are well suited to fragmentary, confounded and noisy data. In general, they are not displacing a previous generation of mechanistic or physics-based models but instead merging with such models while also tackling a wealth of topics that have historically proven computationally intractable.

## Acknowledgements

We thank Nazim Bouatta for comments on early versions of this manuscript. This work is supported by the DARPA PANACEA program grant HR0011-19-2-0022 and NCI/NIH grant U54-CA225088.

## References

1. Abadi Martín et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. http://tensorflow.org/ (2015).
2. Paszke A et al. Automatic differentiation in PyTorch. (2017).
3. Bradbury James, Frostig Roy, Hawkins Peter, Matthew James Johnson. JAX: Autograd and XLA. (Google, 2021).
4. LeCun Y, Bengio Y & Hinton G Deep learning. Nature 521, 436–444 (2015). [PubMed: 26017442]
5. He K, Zhang X, Ren S & Sun J Deep Residual Learning for Image Recognition. ArXiv151203385 Cs (2015).
6. Marcus G Deep Learning: A Critical Appraisal. ArXiv180100631 Cs Stat (2018).
7. Travers Ching et al. Opportunities and obstacles for deep learning in biology and medicine. J. R. Soc. Interface 15, 20170387 (2018). [PubMed: 29618526]
8. Russakovsky O et al. ImageNet Large Scale Visual Recognition Challenge. Int. J. Comput. Vis 115, 211–252 (2015).
9. Godfrey JJ, Holliman EC & McDaniel J SWITCHBOARD: telephone speech corpus for research and development. in Proceedings of the 1992 IEEE international conference on Acoustics, speech and signal processing - Volume 1 517–520 (IEEE Computer Society, 1992).
10. Han KJ, Chandrashekaran A, Kim J & Lane I The CAPIO 2017 Conversational Speech Recognition System. ArXiv180100059 Cs (2018).
11. Gulshan V et al. Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs. JAMA 316, 2402–2410 (2016). [PubMed: 27898976]
12. Ting DSW et al. Artificial intelligence and deep learning in ophthalmology. Br. J. Ophthalmol 103, 167–175 (2019). [PubMed: 30361278]
13. Milea D et al. Artificial Intelligence to Detect Papilledema from Ocular Fundus Photographs. N. Engl. J. Med 0, null (2020).

14. Serag A et al. Translational AI and Deep Learning in Diagnostic Pathology. Front. Med. 6, (2019).
15. Zhang Z et al. Pathologist-level interpretable whole-slide cancer diagnosis with deep learning. Nat. Mach. Intell 1, 236–245 (2019).
16. Silver D et al. Mastering the game of Go without human knowledge. Nature 550, 354–359 (2017). [PubMed: 29052630]
17. Coulom R Efficient selectivity and backup operators in Monte-Carlo tree search. in In: Proceedings Computers and Games 2006 (Springer-Verlag, 2006).
18. Lipson H Robots on the run. Nature 568, 174–175 (2019). [PubMed: 30962550]
19. Santoro A et al. A simple neural network module for relational reasoning. ArXiv170601427 Cs (2017).
20. Cortes C & Vapnik V Support-Vector Networks. in Machine Learning 273–297 (1995).
21. Tin Kam Ho. Random decision forests. in Proceedings of 3rd International Conference on Document Analysis and Recognition vol. 1 278–282 vol.1 (1995).
22. Goodfellow I, Bengio Y & Courville A Deep Learning. (The MIT Press, 2016).
23. Chung J, Ahn S & Bengio Y Hierarchical Multiscale Recurrent Neural Networks. ArXiv160901704 Cs (2017).
24. Karpathy A & Fei-Fei L Deep Visual-Semantic Alignments for Generating Image Descriptions. ArXiv14122306 Cs (2015).
25. Introducing the Model Garden for TensorFlow 2. https://blog.tensorflow.org/2020/03/introducing-model-garden-for-tensorflow-2.html.
26. Boyd S & Vandenberghe L Convex Optimization. (Cambridge University Press, 2004).
27. Saarinen S, Bramley R & Cybenko G Ill-Conditioning in Neural Network Training Problems. SIAM J. Sci. Comput 14, 693–714 (1993).
28. Dauphin Y et al. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. ArXiv14062572 Cs Math Stat (2014).
29. Pascanu R, Dauphin YN, Ganguli S & Bengio Y On the saddle point problem for non-convex optimization. ArXiv14054604 Cs (2014).
30. Lee JD et al. First-order Methods Almost Always Avoid Saddle Points. ArXiv171007406 Cs Math Stat (2017).
31. Kingma DP & Ba J Adam: A Method for Stochastic Optimization. ArXiv14126980 Cs (2017).
32. Chollet F Keras. (2015).
33. AlQuraishi M End-to-End Differentiable Learning of Protein Structure. Cell Syst. 8, 292–301.e3 (2019). [PubMed: 31005579]
34. Sadanandan SK, Ranefall P, Guyader SL & Wählby C Automated Training of Deep Convolutional Neural Networks for Cell Segmentation. Sci. Rep 7, 1–7 (2017). [PubMed: 28127051]
35. Gut G, Herrmann MD & Pelkmans L Multiplexed protein maps link subcellular organization to cellular states. Science 361, eaar7042 (2018). [PubMed: 30072512]
36. Shorten C & Khoshgoftaar TM A survey on Image Data Augmentation for Deep Learning. J. Big Data 6, 60 (2019).
37. Wang S, Sun S, Li Z, Zhang R & Xu J Accurate De Novo Prediction of Protein Contact Map by Ultra-Deep Learning Model. PLOS Comput. Biol 13, e1005324 (2017). [PubMed: 28056090]
38. Liu Y, Palmedo P, Ye Q, Berger B & Peng J Enhancing Evolutionary Couplings with Deep Convolutional Neural Networks. Cell Syst. 6, 65–74.e3 (2018). [PubMed: 29275173]
39. Xu J Distance-based protein folding powered by deep learning. Proc. Natl. Acad. Sci 116, 16856–16865 (2019). [PubMed: 31399549]
40. Senior A et al. AlphaFold: Improved protein structure prediction using potentials from deep learning. Nature.
41. Torng W & Altman RB High precision protein functional site detection using 3D convolutional neural networks. Bioinformatics 35, 1503–1512 (2019). [PubMed: 31051039]
42. Gligorijevic V et al. Structure-Based Function Prediction using Graph Convolutional Networks. bioRxiv 786236 (2019) doi:10.1101/786236.

43. Wallach I, Dzamba M & Heifets A AtomNet: A Deep Convolutional Neural Network for Bioactivity Prediction in Structure-based Drug Discovery. ArXiv151002855 Cs Q-Bio Stat (2015).

44. Gomes J, Ramsundar B, Feinberg EN & Pande VS Atomic Convolutional Networks for Predicting Protein-Ligand Binding Affinity. ArXiv170310603 Phys. Stat (2017).

45. Benos PV, Lapedes AS & Stormo GD Is there a code for protein–DNA recognition? Probab(ilistical)ly…. BioEssays 24, 466–475 (2002). [PubMed: 12001270]

46. Alipanahi B, Delong A, Weirauch MT & Frey BJ Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. Nat. Biotechnol 33, 831–838 (2015). [PubMed: 26213851]

47. Avsec Z et al. Deep learning at base-resolution reveals motif syntax of the cis-regulatory code. bioRxiv 737981 (2019) doi:10.1101/737981.

48. Wu Z et al. A Comprehensive Survey on Graph Neural Networks. ArXiv190100596 Cs Stat (2019).

49. Segler MHS, Preuss M & Waller MP Planning chemical syntheses with deep neural networks and symbolic AI. Nature 555, 604 (2018). [PubMed: 29595767]

50. Bouatta N, Sorger P & AlQuraishi M Protein structure prediction by AlphaFold2: are attention and symmetries all you need? Acta Crystallogr. Sect. Struct. Biol 77, 982–991 (2021).

51. Jumper J et al. Highly accurate protein structure prediction with AlphaFold. Nature 1–11 (2021) doi:10.1038/s41586-021-03819-2.

52. Muzio G, O'Bray L & Borgwardt K Biological network analysis with deep learning. Brief. Bioinform 22, 1515–1530 (2021). [PubMed: 33169146]

53. Chowdhury R et al. Single-sequence protein structure prediction using language models from deep learning. bioRxiv 2021.08.02.454840 (2021) doi:10.1101/2021.08.02.454840.

54. Hall B Lie Groups, Lie Algebras, and Representations: An Elementary Introduction. (Springer, 2004).

55. Cohen T, Geiger M & Weiler M A General Theory of Equivariant CNNs on Homogeneous Spaces. ArXiv181102017 Cs Stat (2018).

56. Weiler M, Geiger M, Welling M, Boomsma W & Cohen T 3D Steerable CNNs: Learning Rotationally Equivariant Features in Volumetric Data. ArXiv180702547 Cs Stat (2018).

57. Zhang R Making Convolutional Networks Shift-Invariant Again. ArXiv190411486 Cs (2019).

58. Gao M & Skolnick J Structural space of protein–protein interfaces is degenerate, close to complete, and highly connected. Proc. Natl. Acad. Sci 107, 22517–22522 (2010). [PubMed: 21149688]

59. Gainza P et al. Deciphering interaction fingerprints from protein molecular surfaces. bioRxiv 606202 (2019) doi:10.1101/606202.

60. Akbar R et al. A finite vocabulary of antibody-antigen interaction enables predictability of paratope-epitope binding. bioRxiv 759498 (2019) doi:10.1101/759498.

61. Cunningham J, Koytiger G, Sorger PK & AlQuraishi M Biophysical prediction of protein-peptide interactions and signaling networks using machine learning. Nat. Methods (2020).

62. Townshend RJL, Bedi R, Suriana PA & Dror RO End-to-End Learning on 3D Protein Structure for Interface Prediction. ArXiv180701297 Cs Q-Bio Stat (2019).

63. Paggi JM et al. Leveraging non-structural data to predict structures of protein–ligand complexes. bioRxiv 2020.06.01.128181 (2020) doi:10.1101/2020.06.01.128181.

64. Berg S et al. ilastik: interactive machine learning for (bio)image analysis. Nat. Methods (2019) doi:10.1038/s41592-019-0582-9.

65. Krueger R et al. Facetto: Combining Unsupervised and Supervised Learning for Hierarchical Phenotype Analysis in Multi-Channel Image Data. IEEE Trans. Vis. Comput. Graph (2019) doi:10.1109/TVCG.2019.2934547.

66. Bialek W Biophysics: Searching for Principles. (Princeton University Press, 2012).

67. Nguyen TH et al. Bayesian analysis of isothermal titration calorimetry for binding thermodynamics. PLOS ONE 13, e0203224 (2018). [PubMed: 30212471]

68. Chen TQ, Rubanova Y, Bettencourt J & Duvenaud DK Neural Ordinary Differential Equations. in Advances in Neural Information Processing Systems 31 (eds. Bengio S et al.) 6571–6583 (Curran Associates, Inc., 2018).

69. Yuan B et al. CellBox: Interpretable Machine Learning for Perturbation Biology with Application to the Design of Cancer Combination Therapy. Cell Syst. 12, 128–140.e4 (2021). [PubMed: 33373583]

70. Baek M et al. Accurate prediction of protein structures and interactions using a three-track neural network. Science (2021) doi:10.1126/science.abj8754.

71. Branden C & Tooze J Introduction to Protein Structure. (Garland Science, 1999).

72. Parsons J, Holmes JB, Rojas JM, Tsai J & Strauss CEM Practical conversion from torsion space to Cartesian space for in silico protein synthesis. J. Comput. Chem 26, 1063–1068 (2005). [PubMed: 15898109]

73. AlQuraishi M ProteinNet: a standardized data set for machine learning of protein structure. BMC Bioinformatics 20, 311 (2019). [PubMed: 31185886]

74. Fuchs FB, Worrall DE, Fischer V & Welling M SE(3)-Transformers: 3D Roto-Translation Equivariant Attention Networks. ArXiv200610503 Cs Stat (2020).

75. Devlin J, Chang M-W, Lee K & Toutanova K BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. ArXiv181004805 Cs (2019).

76. Vaswani A et al. Attention Is All You Need. ArXiv170603762 Cs (2017).

77. Lee H-J & Zheng JJ PDZ domains and their binding partners: structure, specificity, and modification. Cell Commun. Signal 8, 8 (2010). [PubMed: 20509869]

78. Song J, Hao Y, Du Z, Wang Z & Ewing RM Identifying Novel Protein Complexes in Cancer Cells Using Epitope-Tagging of Endogenous Human Genes and Affinity-Purification Mass Spectrometry. J. Proteome Res 11, 5630–5641 (2012). [PubMed: 23106643]

79. Chatr-aryamontri A et al. The BioGRID interaction database: 2017 update. Nucleic Acids Res. 45, D369–D379 (2017). [PubMed: 27980099]

80. Luck K et al. A reference map of the human binary protein interactome. Nature 580, 402–408 (2020). [PubMed: 32296183]

81. Szklarczyk D et al. STRING v11: protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. Nucleic Acids Res. 47, D607–D613 (2019). [PubMed: 30476243]

82. Martins AFT & Astudillo RF From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification. ArXiv160202068 Cs Stat (2016).

83. Rasmussen CE & Williams CKI Gaussian Processes for Machine Learning. (The MIT Press, 2005).

84. Maclaurin D, Duvenaud D & Adams RP Gradient-based Hyperparameter Optimization through Reversible Learning. ArXiv150203492 Cs Stat (2015).

85. Lorraine J & Duvenaud D Stochastic Hyperparameter Optimization through Hypernetworks. ArXiv Prepr. ArXiv180209419 (2018).

86. Burgess DJ Spatial transcriptomics coming of age. Nat. Rev. Genet 20, 317–317 (2019). [PubMed: 30980030]

87. Reddy RJ et al. Early signaling dynamics of the epidermal growth factor receptor. Proc. Natl. Acad. Sci. U. S. A 113, 3114–3119 (2016). [PubMed: 26929352]

88. Maier T, Güell M & Serrano L Correlation of mRNA and protein in complex biological samples. FEBS Lett. 583, 3966–3973 (2009). [PubMed: 19850042]

89. Garnett MJ et al. Systematic identification of genomic markers of drug sensitivity in cancer cells. Nature 483, 570–575 (2012). [PubMed: 22460902]

90. Barretina J et al. The Cancer Cell Line Encyclopedia enables predictive modelling of anticancer drug sensitivity. Nature 483, 603–607 (2012). [PubMed: 22460905]

91. Costello JC et al. A community effort to assess and improve drug sensitivity prediction algorithms. Nat. Biotechnol **advance online publication**, (2014).**advance online publication**

92. Aldridge BB, Burke JM, Lauffenburger DA & Sorger PK Physicochemical modelling of cell signalling pathways. Nat Cell Biol 8, 1195–203 (2006). [PubMed: 17060902]

93. Rackauckas C et al. Universal Differential Equations for Scientific Machine Learning. ArXiv200104385 Cs Math Q-Bio Stat (2020).

94. Yang J, Li A, Li Y, Guo X & Wang M A novel approach for drug response prediction in cancer cell lines via network representation learning. Bioinformatics 35, 1527–1535 (2019). [PubMed: 30304378]

95. Neil D, Pfeiffer M & Liu S-C Phased LSTM: Accelerating Recurrent Network Training for Long or Event-based Sequences. ArXiv161009513 Cs (2016).

96. Eydgahi H et al. Properties of cell death models calibrated and compared using Bayesian approaches. Mol. Syst. Biol 9, 644 (2013). [PubMed: 23385484]

97. Dillon JV et al. TensorFlow Distributions. ArXiv171110604 Cs Stat (2017).

98. Bingham E et al. Pyro: Deep Universal Probabilistic Programming. ArXiv181009538 Cs Stat (2018).

99. Hafner M, Niepel M & Sorger PK Alternative drug sensitivity metrics improve preclinical cancer pharmacogenomics. Nat. Biotechnol 35, 500–502 (2017). [PubMed: 28591115]

100. Saar-Tsechansky M & Provost F Handling Missing Values when Applying Classification Models. J. Mach. Learn. Res 8, 1623–1657 (2007).

101. Bepler T & Berger B Learning the protein language: Evolution, structure, and function. Cell Syst. 12, 654–669.e3 (2021). [PubMed: 34139171]

102. Bepler T & Berger B Learning protein sequence embeddings using information from structure. (2018).

103. Alley EC, Khimulya G, Biswas S, AlQuraishi M & Church GM Unified rational protein engineering with sequence-based deep representation learning. Nat. Methods 16, 1315–1322 (2019). [PubMed: 31636460]

104. Elnaggar A et al. ProtTrans: Towards Cracking the Language of Life's Code Through Self-Supervised Deep Learning and High Performance Computing. bioRxiv 2020.07.12.199554 (2020) doi:10.1101/2020.07.12.199554.

105. Madani A et al. ProGen: Language Modeling for Protein Generation. ArXiv200403497 Cs Q-Bio Stat (2020).

106. Rives A et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. Proc. Natl. Acad. Sci 118, (2021).

107. Biswas S, Khimulya G, Alley EC, Esvelt KM & Church GM Low-N protein engineering with data-efficient deep learning. Nat. Methods 18, 389–396 (2021). [PubMed: 33828272]

108. Weißenow K, Heinzinger M & Rost B Protein language model embeddings for fast, accurate, alignment-free protein structure prediction. bioRxiv 2021.07.31.454572 (2021) doi:10.1101/2021.07.31.454572.

109. Bileschi ML et al. Using Deep Learning to Annotate the Protein Universe. bioRxiv 626507 (2019) doi:10.1101/626507.

110. Lai B & Xu J Accurate Protein Function Prediction via Graph Attention Networks with Predicted Structure Information. bioRxiv 2021.06.16.448727 (2021) doi:10.1101/2021.06.16.448727.

111. Gligorijević V et al. Structure-based protein function prediction using graph convolutional networks. Nat. Commun 12, 3168 (2021). [PubMed: 34039967]

112. Rao R et al. MSA Transformer. bioRxiv 2021.02.12.430858 (2021) doi:10.1101/2021.02.12.430858.

113. Sterling T & Irwin JJ ZINC 15 – Ligand Discovery for Everyone. J. Chem. Inf. Model 55, 2324–2337 (2015). [PubMed: 26479676]

114. Hu* W et al. Strategies for Pre-training Graph Neural Networks. in (2019).

115. Liu S, Demirel MF & Liang Y N-Gram Graph: Simple Unsupervised Representation for Graphs, with Applications to Molecules. 13.

116. Chithrananda S, Grand G & Ramsundar B ChemBERTa: Large-Scale Self-Supervised Pretraining for Molecular Property Prediction. ArXiv201009885 Phys. Q-Bio (2020).

117. Wang Y, Wang J, Cao Z & Farimani AB MolCLR: Molecular Contrastive Learning of Representations via Graph Neural Networks. ArXiv210210056 Phys. (2021).

118. Zhu J et al. Dual-view Molecule Pre-training. ArXiv210610234 Cs Q-Bio (2021).

119. Goodfellow I et al. Generative Adversarial Nets. in Advances in Neural Information Processing Systems 27 (eds. Ghahramani Z, Welling M, Cortes C, Lawrence ND & Weinberger KQ) 2672–2680 (Curran Associates, Inc., 2014).

120. Kingma DP & Welling M Auto-Encoding Variational Bayes. ArXiv13126114 Cs Stat (2013).

121. Kobyzev I, Prince SJD & Brubaker MA Normalizing Flows: An Introduction and Review of Current Methods. IEEE Trans. Pattern Anal. Mach. Intell 1–1 (2020) doi:10.1109/TPAMI.2020.2992934.

122. Sohl-Dickstein J, Weiss EA, Maheswaranathan N & Ganguli S Deep Unsupervised Learning using Nonequilibrium Thermodynamics. ArXiv150303585 Cs Stat (2015).

123. Karras T, Aila T, Laine S & Lehtinen J Progressive Growing of GANs for Improved Quality, Stability, and Variation. ArXiv171010196 Cs Stat (2017).

124. De Cao N & Kipf T MolGAN: An implicit generative model for small molecular graphs. ArXiv180511973 Cs Stat (2018).

125. Gómez-Bombarelli R et al. Automatic chemical design using a data-driven continuous representation of molecules. ACS Cent. Sci 4, 268–276 (2018). [PubMed: 29532027]

126. Zhavoronkov A et al. Deep learning enables rapid identification of potent DDR1 kinase inhibitors. Nat. Biotechnol 37, 1038–1040 (2019). [PubMed: 31477924]

127. Killoran N, Lee LJ, Delong A, Duvenaud D & Frey BJ Generating and designing DNA with deep generative models. ArXiv171206148 Cs Q-Bio Stat (2017).

128. Anand N, Eguchi R & Huang P-S Fully differentiable full-atom protein backbone generation. (2019).

129. Ingraham J, Garg VK, Barzilay R & Jaakkola T Generative Models for Graph-Based Protein Design. (2019).

130. Marouf M et al. Realistic in silico generation and augmentation of single cell RNA-seq data using Generative Adversarial Neural Networks. bioRxiv 390153 (2018) doi:10.1101/390153.

131. Johnson-Roberson M et al. Driving in the Matrix: Can Virtual Worlds Replace Human-Generated Annotations for Real World Tasks? ArXiv161001983 Cs (2017).

132. Martin RM Electronic Structure: Basic Theory and Practical Methods. (Cambridge University Press, 2008).

133. Smith JS, Isayev O & Roitberg AE ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. Chem. Sci 8, 3192–3203 (2017). [PubMed: 28507695]

134. Brockherde F et al. Bypassing the Kohn-Sham equations with machine learning. Nat. Commun 8, 872 (2017). [PubMed: 29021555]

135. Zhang L, Han J, Wang H, Car R & E W Deep Potential Molecular Dynamics: A Scalable Model with the Accuracy of Quantum Mechanics. Phys. Rev. Lett 120, 143001 (2018). [PubMed: 29694129]

136. Open AI et al. Solving Rubik’s Cube with a Robot Hand. ArXiv191007113 Cs Stat (2019).

137. Kulkarni TD, Whitney WF, Kohli P & Tenenbaum JB Deep Convolutional Inverse Graphics Network. in Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2 2539–2547 (MIT Press, 2015).

138. Carreira-Perpinan MA & Hinton GE On contrastive divergence learning. in Aistats vol. 10 33–40 (Citeseer, 2005).

139. Jumper JM, Faruk NF, Freed KF & Sosnick TR Trajectory-based training enables protein simulations with accurate folding and Boltzmann ensembles in cpu-hours. PLoS Comput. Biol 14, (2018).

140. Ingraham J, Riesselman A, Sander C & Marks D Learning Protein Structure with a Differentiable Simulator. in ICLR (2019).

141. Wu J et al. EBM-Fold: Fully-Differentiable Protein Folding Powered by Energy-based Models. ArXiv210504771 Cs (2021).

142. Bayesian Nonparametrics. (Cambridge University Press, 2010).

143. Rezende DJ, Mohamed S & Wierstra D Stochastic Backpropagation and Approximate Inference in Deep Generative Models. in Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 II-1278–II–1286 (JMLR.org, 2014).

144. Sutton RS & Barto AG Reinforcement Learning: An Introduction. (A Bradford Book, 2018).

145. Suarez J, Du Y, Isola P & Mordatch I Neural MMO: A Massively Multiagent Game Environment for Training and Evaluating Intelligent Agents. ArXiv190300784 Cs Stat (2019).

146. Vinyals O et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature 575, 350–354 (2019). [PubMed: 31666705]

147. Mikulak-Klucznik B et al. Computational planning of the synthesis of complex natural products. Nature 1–6 (2020) doi:10.1038/s41586-020-2855-y.

148. Eastman P, Shi J, Ramsundar B & Pande VS Solving the RNA design problem with reinforcement learning. PLOS Comput. Biol 14, e1006176 (2018). [PubMed: 29927936]

149. Cho J, Lee K, Shin E, Choy G & Do S How much data is needed to train a medical image deep learning system to achieve necessary high accuracy? ArXiv151106348 Cs (2016).

150. Zhou J et al. Graph Neural Networks: A Review of Methods and Applications. ArXiv181208434 Cs Stat (2021).

151. Wu Z et al. A Comprehensive Survey on Graph Neural Networks. IEEE Trans. Neural Netw. Learn. Syst 32, 4–24 (2021). [PubMed: 32217482]

152. Bowman SR et al. Generating Sentences from a Continuous Space. ArXiv151106349 Cs (2016).

153. Anonymous. Deep Learning For Symbolic Mathematics. (2019).

154. Grefenstette E, Hermann KM, Suleyman M & Blunsom P Learning to Transduce with Unbounded Memory. ArXiv150602516 Cs (2015).

155. Grover A, Wang E, Zweig A & Ermon S Stochastic Optimization of Sorting Networks via Continuous Relaxations. (2018).

156. Graves A Adaptive Computation Time for Recurrent Neural Networks. ArXiv160308983 Cs (2016).

157. Trask A et al. Neural Arithmetic Logic Units. ArXiv180800508 Cs (2018).

158. Jin W, Barzilay R & Jaakkola T Junction Tree Variational Autoencoder for Molecular Graph Generation. (2018).

159. Amodei D & Hernandez D AI and Compute. Heruntergeladen Von Httpsblog Openai Comaiand-Compute (2018).

160. Weld DS & Bansal G The Challenge of Crafting Intelligible Intelligence. (2018).

161. Chakraborty S et al. Interpretability of deep learning models: A survey of results. in 2017 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computed, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI) 1–6 (2017). doi:10.1109/UIC-ATC.2017.8397411.

162. Machine Learning Meets Quantum Physics. (Springer International Publishing, 2020). doi:10.1007/978-3-030-40245-7.

163. Kryshtafovych A, Schwede T, Topf M, Fidelis K & Moult J Critical assessment of methods of protein structure prediction (CASP)—Round XIII. Proteins Struct. Funct. Bioinforma 87, 1011–1020 (2019).

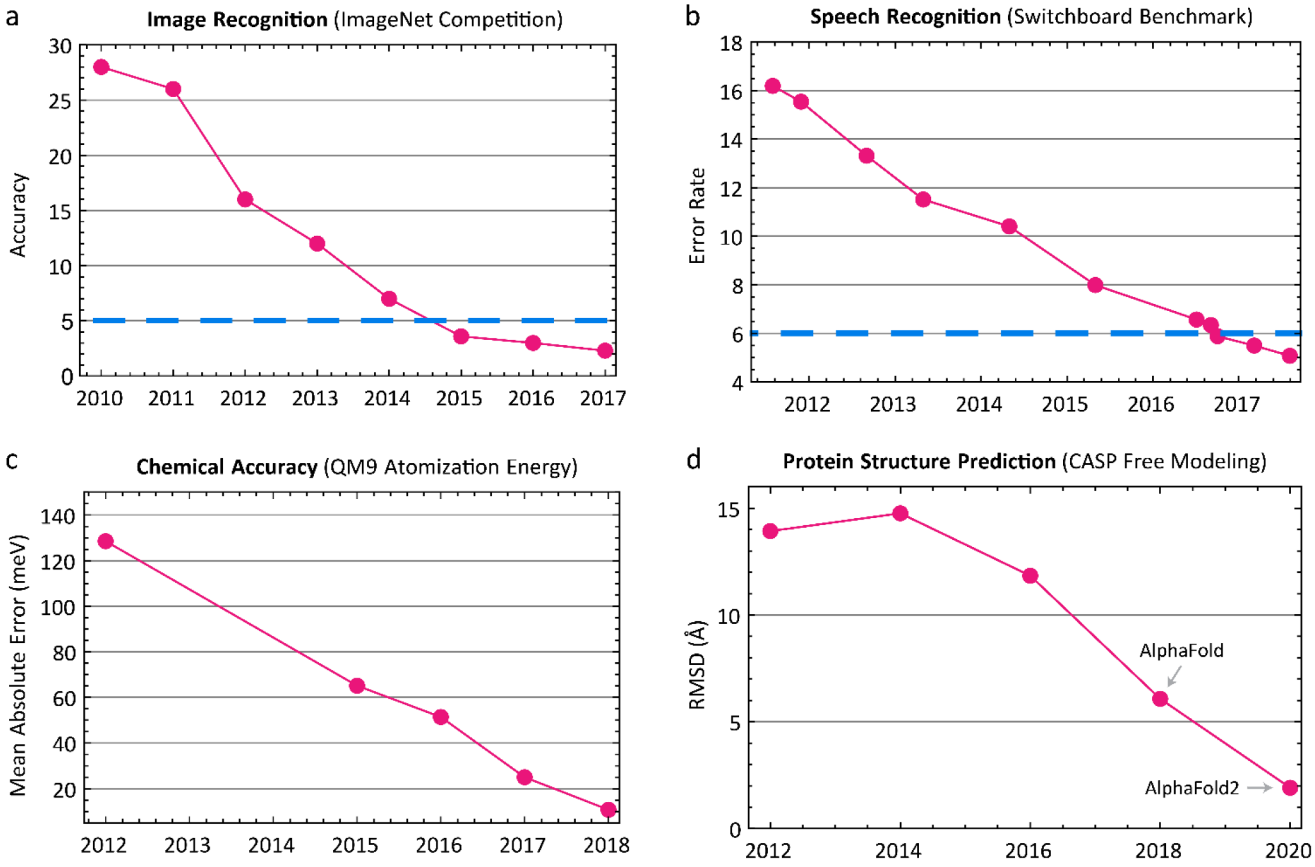

**Figure 1: Deep Learning Revolution.**

Improvements in prediction accuracy driven by deep learning over the last decade in **(a)** image recognition tasks[8], **(b)** speech recognition[9,10], **(c)** quantum chemical calculations[162], and **(d)** protein structure prediction[163]. Human baselines based on expert curators are shown as dashed blue lines.

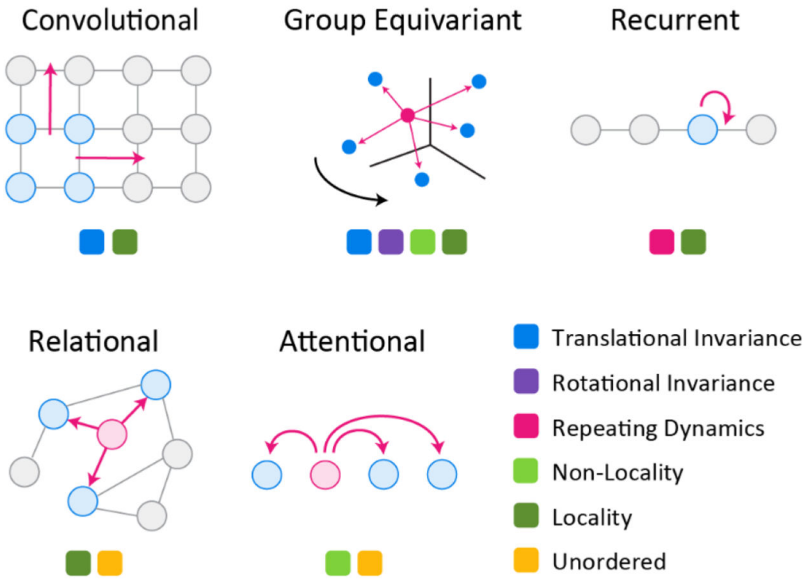

**Figure 2: Neural Network Primitives.**

A powerful set of neural network building blocks makes it possible to build learnable models that encode a variety of inductive priors. Convolutional networks model regular grids such as images or sequences, inducing local structure and limited forms of spatial invariance such as indifference to shifts in images. They are generalized by group-equivariant networks that operate on arbitrary point clouds and induce local and global structure as well as more general spatial invariances including rotational and translational shifts, important in molecular applications. Recurrent networks model sequences with repeating dynamics such as time series, music, or the actions of a computational agent. Relational or graph networks reflect highly structured objects with rich interrelationships such as phylogenetic trees. Attention networks on the other hand essentially assume no underlying structure and are capable of inferring arbitrarily complex relationships, including long-range interactions that have historically been difficult to capture with conventional mathematical models. This ability has been crucial to the development of accurate methods for protein structure prediction. These primitives can be combined to yield even more complex combinations, for example group-equivariant attentional networks[74].

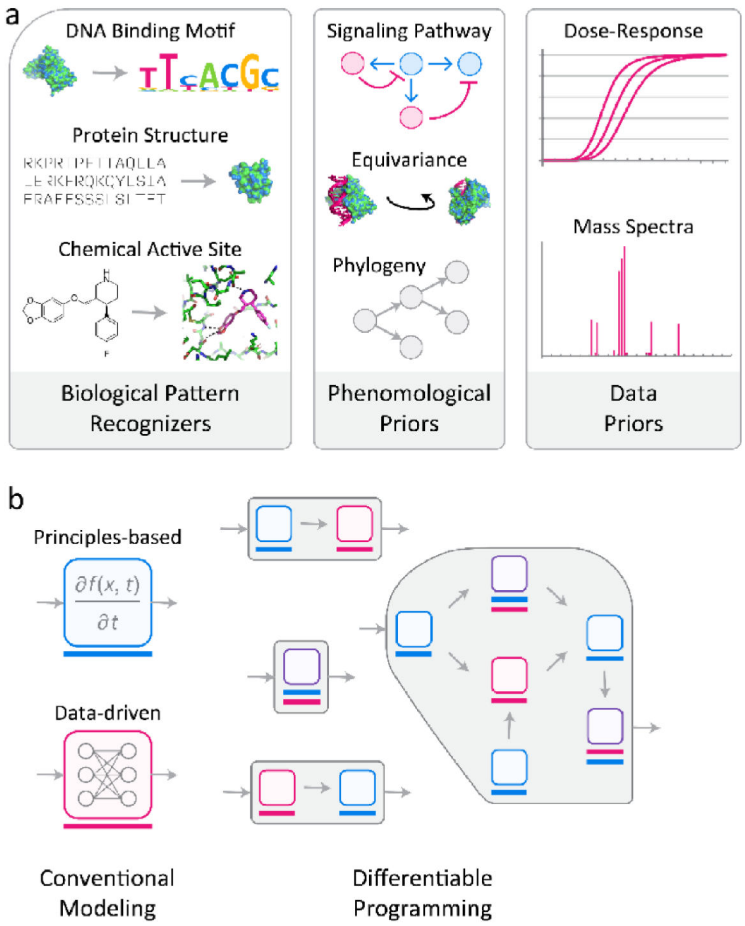

**Figure 3: Differentiable Programming Fuses Principles-based and Data-driven Modeling.**

**(a)** Three types of primitives underlie the emerging field of differentiable biology: (i) biological pattern recognizers that perform mappings too complex to be interpretable, such as predicting the DNA binding motif of a transcription factor from its structure, (ii) phenomenological priors that encode existing biological knowledge, such as known signaling pathways, and (iii) data priors that capture the data acquisition process, for example the physical process underlying mass spectrometry. **(b)** In conventional modeling, principles-based and data-driven approaches are used largely independently. Differentiable programming makes it impossible to build bespoke systems that intermingle the two types of approaches in a manner that best reflects the desired modeling task.

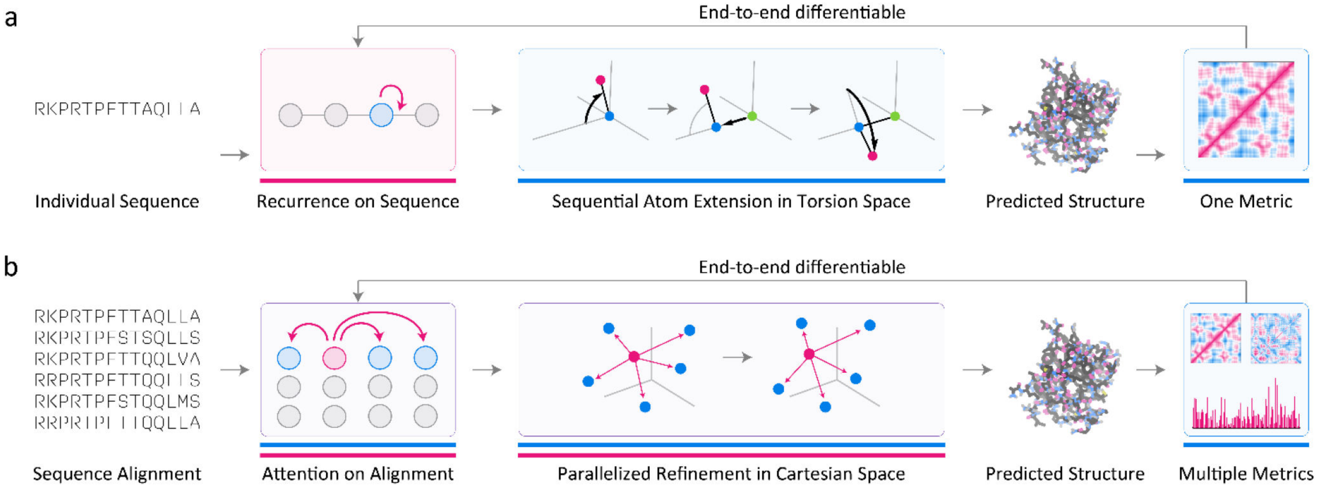

**Figure 4: Protein Structure Prediction Vignette.**

**(a)** A minimal end-to-end differentiable system[33] for protein structure prediction accepts a variable-length protein sequence and processes it recurrently, implicitly learning sequence-torsion patterns (pink underlines indicate purely data-driven processes that do not rely on prior knowledge). These learned patterns are then converted sequentially into 3D coordinates using known (fixed) equations for converting sequences of torsion angles to Cartesian coordinates (blue underlines indicate purely knowledge-based processes that do not utilize learning.) After the final structure is produced, a rotationally- and translationally-invariant error metric computes its deviation from an experimental structure, feeding this information back into the learning loop. **(b)** A more advanced system for protein structure prediction, based on reported features of AlphaFold2, would accept multiple sequence alignments of protein sequences, using attention to reason over individual sequences and residues in the alignment. Based on learned sequence-structure patterns, an initial set of 3D coordinates are predicted then refined using attention mechanisms that operate directly on the 3D structure and that are equivariant to both translations and rotations. The predicted structure is then assessed using multiple error metrics which are fed back into the learning loop.

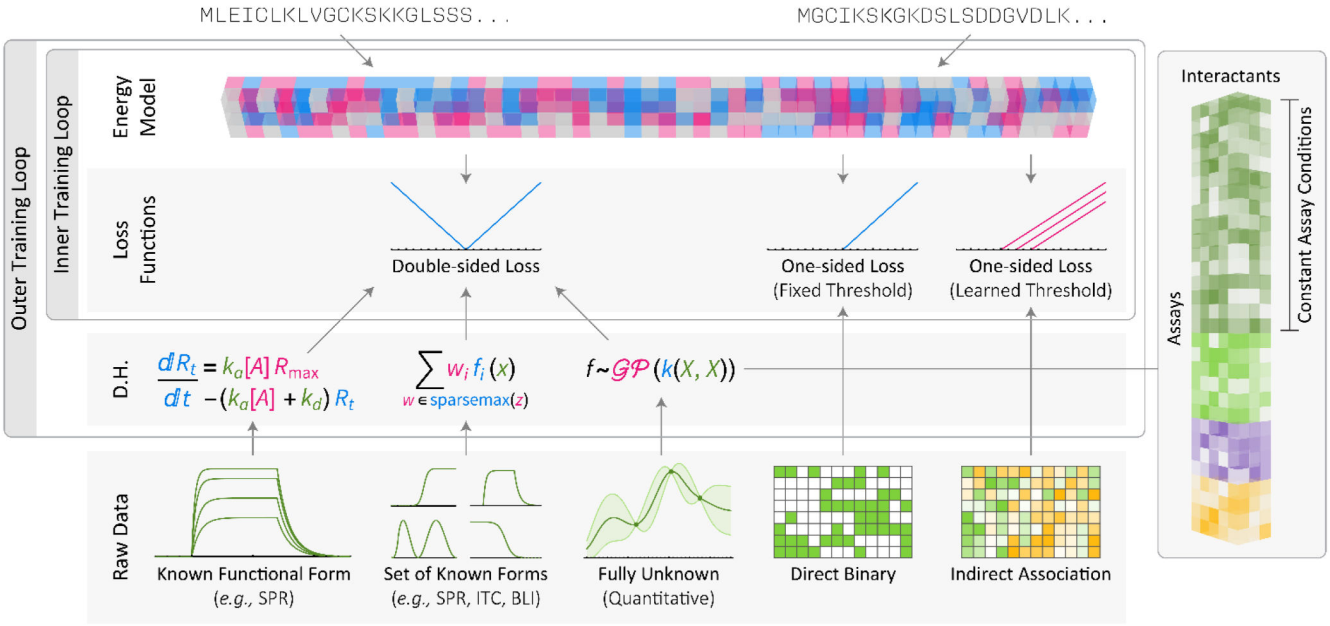
MLEICLKLVGCKSKKGLSSS. . .
MGCIKSKGKDSLSDDGVDLK. . .
Outer Training Loop
Inner Training Loop
Energy Model
Loss Functions
Double-sided Loss
One-sided Loss (Fixed Threshold)
One-sided Loss (Learned Threshold)
D.H.
$\frac{dR_t}{dt} = k_a[A]R_{max} - (k_a[A] + k_d)R_t$
$\sum_{w \in \text{sparsemax}(z)} w_i f_i(x)$
$f \sim \mathcal{GP}(k(X, X))$
Raw Data
Known Functional Form (e.g., SPR)
Set of Known Forms (e.g., SPR, ITC, BLI)
Fully Unknown (Quantitative)
Direct Binary
Indirect Association
Interactants
Assays
Constant Assay Conditions

**Figure 5: Protein-Protein Interaction Vignette.**

An integrated system for data homogenization and prediction of protein-protein binding affinity is illustrated. The system accepts sequences of two proteins (top) that are fed to a learned energy model to quantitatively predict their disassociation rate. To train the model, multiple data types with varying degrees of precision, directness, and physical characterization are used (bottom). Depending on the data type, a different data homogenizer (D.H.) is used to bring all data modalities into congruence. For quantitative data, conventional double-sided loss functions are used to train the model whenever its predictions deviate from the ground truth. For binary data, one-sided and potentially learnable loss functions are used (see main text) to only penalize predictions that are clearly in conflict with the ground truth. The entire model, including the parameters of the energy model and the data homogenizers, is trained jointly using an inner loop for the energy model and an outer loop for the data homogenizers to ensure correct training behavior. A key assumption of the model is that the number of distinct experimental conditions and assays is substantially smaller than the number of distinct data points (right). Otherwise, the model is non-identifiable. Throughout the illustration green indicates raw data, blue indicates terms coming from principles-based modeling, and pink indicates learnable quantities.