Gene Expression and Protein Function: A Survey of Deep Learning Methods

Saket Sathe* ssathe@us.ibm.com
Sayani Aggarwal† sayaniaggarwal@gmail.com
Jiliang Tang‡ tangjili@msu.edu

ABSTRACT

Deep learning methods have found increasing interest in recent years because of their wide applicability for prediction and inference in numerous disciplines such as image recognition, natural language processing, and speech recognition. Computational biology is a data-intensive field in which the types of data can be very diverse. These different types of structured data require different neural architectures. The problems of gene expression and protein function prediction are related areas in computational biology (since genes control the production of proteins). This survey provides an overview of the various types of problems in this domain and the neural architectures that work for these data sets.

Since deep learning is a new field compared to traditional machine learning, much of the work in this area corresponds to traditional machine learning rather than deep learning. However, as the sizes of protein and gene expression data sets continue to grow, the possibility of using data-hungry deep learning methods continues to increase. Indeed, the previous five years have seen a sudden increase in deep learning models, although some areas of protein analytics and gene expression still remain relatively unexplored. Therefore, aside from the survey on the deep learning work *directly* related to these problems, we also point out existing deep learning work from other domains that has the *potential* to be applied to these domains.

1. INTRODUCTION

Deep learning has become increasingly popular in recent years because of its uses in predictive applications, especially in the image and sequential domain [Goodfellow et al. 2016]. Deep learning models are generalizations of traditional machine learning models like linear regression and logistic regression. In particular, deep learning methods work well when the data are richly structured in terms of the relationships among different attributes. A classical example of this type of data is the image domain. In such cases, deep learning methods can leverage the network depth in order to engineer features from the underlying data.

Biological data represent a natural candidate for deep learning algorithms because of the highly structured nature of such data. Some of the biological data occurs in the form of sequences, others as complex folded compounds, and yet others as microarrays. All these different types of data require different types of deep learning algorithms. The diversity of deep learning methods in computational biology is quite vast, which is inherited from the rich diversity of different types of biological data. In this survey, we examine two related areas of computational biology, which are gene expression and protein function prediction. These areas are related because proteins are created using gene templates, and the specific sequence in gene defines the eventual function of the protein. There is a rich diversity of data and problem types that are frequently studied in this domain.

This survey assumes the basic knowledge of neural networks and deep learning as a prerequisite. Although a brief overview of neural networks is provided, it is not the focus of the survey. The primary focus is on *how* these models can be used in the context of protein and gene expression data. Where needed, some background is also provided on how biological concepts relate to the underlying machine learning problems. We refer the reader to [Bishop 1995; Goodfellow et al. 2016] for overviews on neural networks.

1.1 Types of Data Addressed in this Survey

Although diverse types of biological data exist, the focus of this survey is primarily on protein data and genomic data. As discussed in section 3.2, proteins in the human body are manufactured from a base template inside the DNA. Therefore, there is a natural connection between proteomics and geonomics. Indeed, genes are almost always *expressed* in order to create proteins with specific functions. Indeed, the primary way in which DNA works is via the manufacturing of different types of proteins. In particular, the following aspects are studied in this survey:

- *Gene expression:* Genomic data are among the most popular types of biological data. Typically, genes occur as sequences, which can be analyzed for structure and prediction. Genomic data also occur as microarrays, which have a somewhat different type of 2-dimensional structure. A multitude of applications exist for such data, such as the discovering of various types of diseases. This survey studies how specific aspects of the structure of genes are related to expressed characteristics, such as the occurrence of diseases. For example, gene microarrays are often used to perform studies of the gene mutations that lead to various types of cancers, tumors, and other diseases.
- *Deep learning of proteins:* The chemical properties of

*IBM T. J. Watson Research Center, Yorktown Heights, NY
†Lakeland Senior High School, Shrub Oak, NY
‡Michigan State University, East Lansing, MI

biological compounds are closely related to their structures. For example, the function of a protein is closely related to its structures. Proteins occur as complex 3-dimensional shapes that are leveraged for analysis and prediction. The properties of proteins are often associated with their structures and with functions. Deep learning methods can help in relating biological compounds to their structure and function. This survey presents a discussion of deep learning methods that predict the structures, functions, and shapes of proteins together with the interrelationships between these aspects.

- *Connections to other types of data:* In many cases, protein data and gene data do not occur in isolation, but they may be annotated with various types of text and may co-occur with other data types. Much of the scientific literature of biomedical discoveries is available in the form of text. In many cases, useful results are obscured by the sheer volume of the publications available in the literature. Therefore, combining the analyses of these large volumes of text with the insights obtained directly from protein and gene data requires a great deal of work. Furthermore, some types of genetic data and protein data occur in combination with various types of annotations that can be exploited for analytics. Some of these methods are discussed in this survey.

Important common property of most types of biological data (such as proteins and genetic data) is that it has a rich amount of structure. Proteins and genes are often modeled as sequences or graphs, which are highly structured data sets. Such data domains are naturally suited to deep learning algorithms. This survey provides a discussion of these different types of applications along with the various types of neural networks that support these applications.

1.2 Previous Surveys

Numerous surveys on machine learning methods in computational biology [Caragea and Honavar 2009; Wang et al. 2005; Schölkopf et al. 2004; Noble et al. 2004], although many of these surveys were written before deep learning methods became popular. Another point that we mention is that computational biology is a diverse field, and a survey of all the areas of computational biology would be worthy of a book rather than a survey. It is often hard to provide a focused overview of such a broad area (and also provide specific insight and positioning of the works already done) without focusing on specific domains. This survey is, therefore, focused on the subject area of protein data analytics and gene expression. The following is an overview of related surveys in the literature:

1. A survey on deep learning for biological data may be found in [Angermueller et al. 2016]. This survey focuses on the basics of deep learning methods and also the broad classes of biological problems that can be addressed by deep learning. In contrast, our survey takes basic deep learning knowledge as a prerequisite and builds on it in the context of proteomics and genomics. Therefore, the article is arranged around *protein analytics and gene expression applications*; of course, the broader principles of deep learning methods used for various applications are provided to provide better insights.

2. A review of deep learning methods on health informatics is provided in [Ravì et al. 2017]. However, this article is restricted in its scope to health care-related topics. Computational biology is a distinct field in its own right, although it has significant overlaps with health care. Health care is generally much broader, and it encompasses areas such as patient diagnosis prediction. This is not the focus of this article.

3. An overview article [Webb 2018] in *Nature* provides some interesting perspectives on deep learning methods for biology. However, this article is intended to be an overview article in computational biology (which provides an excellent bird's-eye view), but it does not provide details at the survey level. An important reason for this is that it focuses rather broadly on computational biology, which makes it difficult to provide details in specific areas. Furthermore, the survey article in [Webb 2018] is not *specifically* focused on genomic or protein data.

4. An overview of deep learning methods for genomic data may be found in [Yue and Wang 2018]. The focus of genomic data intersects with some aspects of this review, although many aspects of [Yue and Wang 2018] are not directly related to this survey. Furthermore, [Yue and Wang 2018] do not discuss deep learning methods for proteomics.

It is noteworthy that deep learning is a relatively young field, and many obvious avenues for using deep learning in computational biology have not been explored. Therefore, where possible, the obvious avenues and directions for research in using deep learning for protein analytics and genomics are also pointed out in this survey. For this reason, this survey should also be considered a position paper, which provides numerous connections between known techniques and explores obvious avenues for research.

1.3 Organization of Survey

This survey is organized as follows. The next section provides an overview of the key classes of deep learning methods. Although it is impossible to provide a comprehensive overview of the different types of deep learning methods, we provide an overview of how important classes of neural models relate to biological data. Section 3 offers a discussion of deep learning methods for protein data. The connections between protein data and gene data are discussed in this section, as genes provide the blueprints for generating proteins. Numerous algorithms for protein interaction, function, and structure are discussed in this section. Deep learning models for genetic data are discussed in section 4. The underlying deep learning methods include techniques for predicting gene expression and clustering. Furthermore, gene regulatory networks are also discussed in this section. A summary and discussion is given in section 5.

2. AN OVERVIEW OF DEEP LEARNING METHODS

In this section, we provide an overview of the basics of neural networks and deep learning. The field of neural networks is

an extension of the broader field of machine learning. An overview of the broader field of machine learning may be found in [Bishop 2006].

All neural networks are essentially *computational graphs* containing nodes that can perform computations. Such computational graphs are usually directed, acyclic graphs, and they are often organized in a layer-wise fashion. All nodes are either input nodes, hidden nodes, or output nodes. The input nodes simply accept the input to the machine learning problem, whereas the output nodes output the final predictions. The number of input nodes is equal to the number of attributes d in the data set, whereas the number of output nodes is equal to the number of attributes to be predicted. For example, for a regression or classification problem, there might be only one output. The intermediate nodes accept inputs from other nodes and propagate them to the next layer of nodes after performing computations on them. Such nodes are considered hidden because their computations are not part of the input or output (although they can be explored if needed). The input nodes do not perform any computations, but they simply transmit their inputs to the next layer. The nodes are connected to one another with weights on the edges, and the learning of the weights in a data-driven manner provides the primary mechanism with which a neural network is able to model prediction functions. The weights on edges are modified whenever there are errors in the predictions of outputs, and the weights are modified in order to reduce this error.

Each node in a computational graph typically performs two operations in succession. The first is a simple linear computation on its d inputs, which is followed by the application of an activation function. In other words, the output of a node in a computational graph is as follows:

$$y = \Phi(\sum_{i=1}^{d} w_i x_i) \tag{1}$$

The function $\Phi(\cdot)$ is typically the sigmoid, tanh, or the ReLU function. These functions are defined as follows:

$$\Phi(z) = 1/(1+e^{-z}) \text{ (sigmoid function)}$$
$$\Phi(z) = (e^{2z}-1)/(e^{2z}+1) \text{ (tanh function)}$$
$$\Phi(z) = \max\{z, 0\} \text{ (Rectified Linear Unit)}$$

Another useful function is the *softmax* activation function, which is a generalization of the sigmoid function to multiple outputs. The softmax activation function has k inputs $z_i \ldots z_k$, and k outputs $o_1 \ldots o_k$, which can be interpreted as probabilities:

$$o_i = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)}, \quad \forall i \in \{1 \ldots k\} \tag{2}$$

The softmax activation function is used when the prediction is a set of k probabilities corresponding to k possible outcomes. These types of situations are common in multiway prediction problems.

It is also possible to have no activation function, which corresponds to a linear layer. Linear activation functions are often used in the output nodes of a neural network, when the final output is a numerical value. In fact, it is possible to simulate the linear regression model with such a node. In this model, we have a single-layer network with a single output node, where the output y is obtained by applying a combination of a linear function and an activation function to the inputs. An example of a neural network with a single output node is illustrated in Figure 1. Note that the input nodes do not perform any computations beyond transmitting the data, and it is only the output nodes that perform the computation. The output is then compared to an observed output, and a *loss function* is used to quantify the error of that example. The weights of the neural network are modified to minimize the error. The gradient of the loss function is used to update the weights. With the trained weights, one can then predict outputs for unseen examples. For example, when we use no activation, a single numerical output, and a squared loss, the resulting model is referred to as *least-squares regression*. In this case, for inputs $x_1, \ldots x_d$, the single output is given by $\hat{y} = \sum_{i=1}^{d} w_i x_i$, where x_i is the ith input and w_i is the weight of the edge joining the ith input to the output. Then, the loss function of least-squares regression for the input-output pair $(\overline{x}, y)$ is as follows:

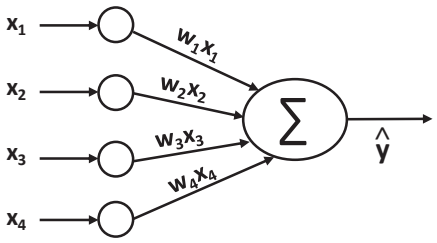


Figure 1: A single-layer network with one computational node for linear regression

$$L(\overline{x}, y) = (y - \hat{y})^2 \tag{3}$$

The losses over all input-output pairs in a data set S are aggregated to yield the final result. Therefore, the overall loss is computed as $L = \sum_{(\overline{x},y) \in S} L(\overline{x}, y)$.

When the outputs have 0/1 values, the sigmoid activation function can be used to predict a probability of the binary prediction. Depending on whether the output is 0 or 1, the negative logarithm of the probability of 0 or 1 is used as the loss. This model is exactly the same as that of logistic regression in machine learning. Therefore, simple cases of neural networks correspond to well-known classification and regression models in machine learning. The softmax activation is used with a logarithmic (cross-entropy) loss in the case of multi-way classification.

Neural networks become much more powerful when the nodes are arranged in multiple layers. In this case, the outputs of some nodes feed into other nodes, and the overall model becomes extremely powerful. Complicated functions of the input can be learned with this approach. Most of this power is gained as a result of the nonlinear activation functions in the intermediate layers. An example of a multi-layer network is illustrated in Figure 2. As in the case of the single-layer network, the inputs are $x_1 \ldots x_5$, and the single output is denoted by $\hat{y}$. Depending on the nature of the application, the network might have multiple outputs. This model is referred to as a *feed-forward network*. The depth of a network increases its power, and the success of deep networks for modeling has led to the term "deep learning." Most of the time, one is using the features of a specific datum (e.g., a protein molecule, a gene sequence, and clinical readings)

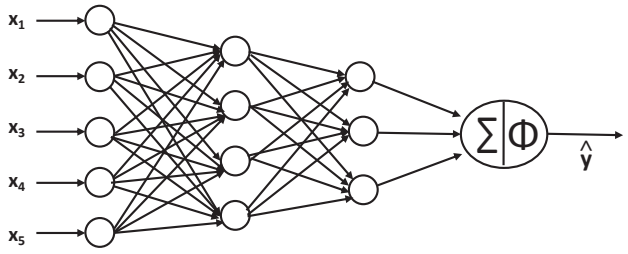


Figure 2: A multi-layer network with nodes structured in layers.

to make a prediction of a specific property (e.g., a molecular property, a genetic property, or a disease). This is the classical approach used in supervised learning.

In unsupervised learning, it is also possible to have outputs that are identical to the inputs. The loss function is designed by summing up the squared error between the predicted outputs and observed outputs (which are the same as the inputs). This approach corresponds to the reconstruction of a data point, which is the case in an *autoencoder*. These types of unsupervised learning methods are useful for applications such as clustering. In many applications, such as gene expression data, clustering methods are useful for grouping genes with similar properties together.

2.1 Importance of Structured Architectures

Many types of data, such as biological sequences and graphs, are inherently structured. For example, proteins can be represented as sequence data, and in some cases, they can also be *conceptually* represented as interaction networks. In these cases, the use of a straightforward feed-forward network does not yield optimal results. In such cases, it is helpful to design architectures that are specifically tailored to these data. There are two primary types of architectures that are suited to the kinds of structured data that occur often in biology. The two most common architectures used for processing biological data correspond to recurrent neural networks and convolutional neural networks.

2.1.1 Recurrent Architectures for Sequences

Recurrent neural networks are useful for biological data that occur as sequences. Many of the types of data that occur often in computational biology are sequential data with variable lengths. Examples include gene sequences and protein sequences. Biological compounds, which are graphs, can also be flattened into sequences. These types of data often have repeating (or *recurrent*) patterns in them, and the goal of the architecture is to learn these recurrent patterns. Therefore, recurrent architectures [Elman 1990; Hochreiter and Schmidhuber 1997] use the notion of a time-layered network, in which each position in the sequence is associated with a layer of the architecture. For example, an amino acid sequence with length k will have k time layers in which each layer receives an input from one position. All the layers in the architecture are identical, and they receive feedback from earlier layers. Each time layer in the network has an input, a set of hidden nodes, and an output. The number of time layers in the network depends on the number of elements in the input sequence, and therefore the architecture of the neural network depends on the length of the input sequence. Such networks have the following properties:

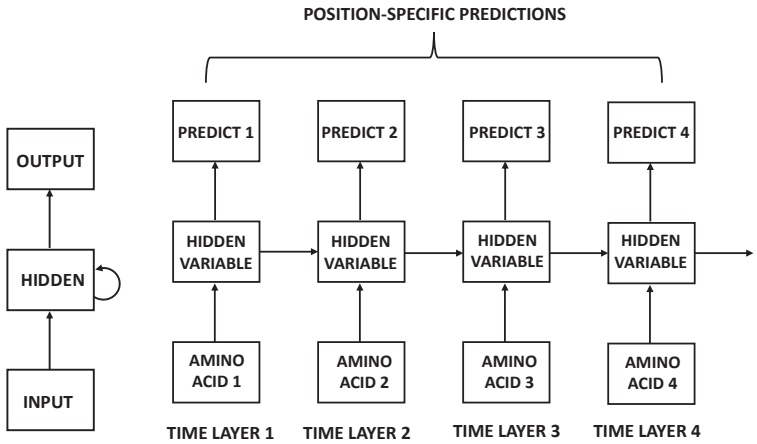
1. The different temporal layers share parameters because of the fact that the model at each time-stamp is identical. Note that it is important for the time layers to share parameters to ensure a fixed number of parameters because the number of time layers is input-specific.

2. The recurrent neural network can accept variable length inputs. This is because each time layer allows a fixed number of inputs, and the number of time layers depends on the length of the sequence. The requirement of variable-length inputs is quite common in the case of sequence data.

An example of a recurrent architecture is shown in Figure 3. Note that the loop in Figure 3(a) is of a conceptual nature, as it is unrolled into multiple temporal layers. The unrolled version of the network is shown in Figure 3(b).

2.1.2 Convolutional Neural Networks in Biology

The main application of convolutional neural networks in computational biology occurs for various types of proteins that are naturally expressed as graphs. Convolutional neural networks [Krizhevsky et al. 2012; LeCun et al. 1998] use a 3-dimensional spatial arrangement of the units, and sparse activations, referred to as *convolutions*, are used to propagate activations from layer to layer. Each layer in a convolutional neural network has spatially arranged activations, and these actions are individual pixels at the input layer. Furthermore, each layer of the network has multiple activation maps, and these different maps are viewed as features. In the input layers, these different maps correspond to channels such as red, green, and blue (RGB). A typical input image might be of size $32 \times 32 \times 3$, where the first two dimensions correspond to the spatial size, and the third dimension corresponds to the depth. Convolutions are done with the use of *filters* that have smaller dimensions, but the same number of maps as the layer. For example, a filter in the input layer might be of size $5 \times 5 \times 3$. Multiple filters might be associated with a layer, which create different features in the next layer. A convolution operation is a dot product between all the elements of a filter and a particular position in the image. Therefore, the number of "pixels" (or features) in the spatial representation of the next layer is equal to the number of valid positions at which a filter can be convolved with the spatially arranged features in a particular layer. A particular filter is associated with a single feature map in the next layer. Therefore, the number of filters in a layer determines the number of feature maps in the next layer. The convolution operations are often paired with a ReLU activation function, which has become the most common approach since the work in [Krizhevsky et al. 2012]. In some convolutional networks, *pooling* operations are also used, although this practice has become increasingly rare in recent years. A pooling operation outputs the maximum value over a spatial region of features. The pooling operation reduces the size of the spatial footprint of a layer because a spatial region is replaced with a single feature. In cases where pooling is not used, it is important to use *strided* convolutions, where the convolution is not done at each position in the spatial footprint, but the spatial positions are separated by an integer value. This integer value is referred to



(a) RNN (b) RNN (unrolled with time-layers shown)

Figure 3: An illustration of a recurrent neural network (RNN) with both rolled and unrolled representations

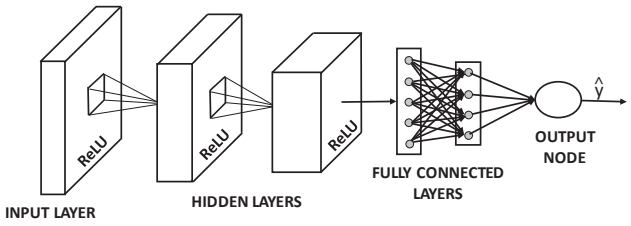


Figure 4: A convolutional neural network

as the *stride* parameter. The final set of layers are not spatially arranged and are referred to as *fully connected layers*. These layers are similar to the ones used in a conventional feedforward network. An overview of the basic structure of a convolutional neural network is provided in [Krizhevsky et al. 2012]. An example of a convolutional neural network that uses only convolutions and no pooling is illustrated in Figure 4. Note that a ReLU activation is placed at the end of each spatial layer.

Convolutional neural networks are used widely for image data, and their use in the biological domain has been limited. Nevertheless, some obvious avenues where convolutional neural networks can be used in computational biology are noted in this survey. An important point is that many biological compounds can be represented as graphs, and the analyses of these graphs have tremendous applications. In the graph representation of a chemical compound, each node is a simpler unit of the compound, and a bond is a connection between the nodes. In recent years, numerous methods have been proposed for extending the use of convolutional neural networks to graphs [Duvenaud et al. 2015; Kipf and Welling 2016a; Kipf and Welling 2016b; Henaff et al. 2015], even though they were originally proposed for image data. At least some of these works [Duvenaud et al. 2015; Henaff et al. 2015] have been shown to have applications in molecular biology.

3. PROTEIN DATA

In this section, we discuss methods that are used for deep learning of non-genomic biological compounds. These include areas of computational biology such as proteomics, in which proteins are analyzed for their structure and function. However, genes are not completely independent of proteins. Indeed, the building of proteins is deeply controlled by genes. Therefore, the study of proteins is deeply connected to that of genes in many ways. Since the research in protein analytics is closely related to that of genetics, it is important to understand these different areas and their relationships with one another.

3.1 What is Proteomics?

Proteins form the bedrock of most structures and functions in living organisms. The word "protein" comes from the Greek word "*proteos*," which means "first place." They take on many complex functions in the human body, including acting as enzymes, hormones, and catalysts, performing various forms of signalling and even providing the physical structure of muscles. Functions, such the transportation of oxygen in the body are performed by proteins, such as Hemoglobin. Defects in the structure of proteins can lead to corresponding problems in their functioning, and a variety of diseases are known to be caused by defects in proteins. Clearly, the structure of proteins plays a critical role in their functioning, and therefore, a natural avenue for the use of deep learning is to predict the function of proteins from their structure.

3.2 Connections of Proteomics and Genomics

Genetic data correspond to deoxyribonucleic acid (DNA) and ribonucleic acid (RNA), which are constructed from *nucleotides*. Nucleotides are monomers made of three components, which correspond to a 5-carbon sugar, a phosphate group, and a nitrogenous base. Therefore, DNA and RNA are chemically quite different from proteins. DNA is a double-stranded, stable, sequence of nucleic acids in which the main sugar is deoxyribose, whereas RNA is a single-stranded, unstable sequence of nucleic acids, in which the main sugar is ribose. For example, instead of amino acids, DNA are composed of the nucleic acids, which are adenine (A), guanine (G), thymine (T), and cytosine(C). Similarly, RNA uses roughly similar bases, except that thymine is replaced with uracil (U). The DNA occurs in the nucleus in

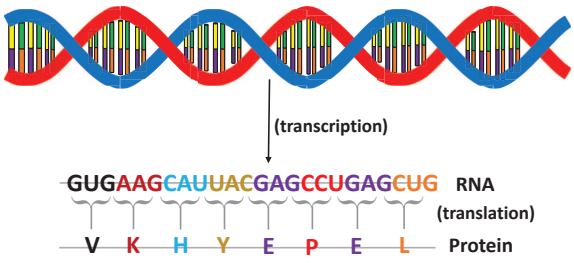


Figure 5: The process of transcription and translation

23 pairs of chromosomes, containing a total of 70,000 genes. Each gene comprises a long sequence of the aforementioned four bases. In spite of the significant chemical differences between genomic data and proteins, genes contain the instructions required for building proteins. For example, a protein is created by combining amino acids, and the ordering and choice of the amino acids is controlled by the gene for the protein. RNA transfers the genetic code from the nucleus to the ribosomes to make proteins. In addition to proteins, amino acids are created in ribosomes. RNA is less stable than DNA precisely because of the types of functions it has to perform, which require frequent reorganization.

What is the connection between DNA, RNA, and proteins? The main point is that DNA lies inside the nucleus of a cell, whereas proteins and amino acids are synthesized outside the nucleus (in the ribosomes present in the cytoplasm). RNA serves as the carrier of this information from within the nucleus to the cytoplasm. Messenger RNA plays the role of *transcription*, where it copies the DNA code from the nucleus, using the same sequence while replacing (T) with (U) and bringing it to the ribosomes, where the proteins are manufactured. Each subsequence of three positions in an RNA strand brought to the ribosome is referred to as a *codon*, and it codes for an amino acid. Therefore, there are $4^3 = 64$ possible codons, although multiple codons might correspond to the same amino acid. Furthermore, some codons are designated to mark the beginning or end of a protein sequence, and they do not specifically code for amino acids. As a result, there are only 20 distinct amino acids (instead of 64). All proteins represent a sequence of these 20 amino acids. An overview of the entire process of transcription and translation from genes to proteins is illustrated in Figure 5.

Proteins are, therefore, represented as long sequences of 20 (possibly and likely repeating) symbols in the FASTA format, [1] and DNA/RNA are represented as repeating sequences of four symbols. Each monomer (amino acid) in the protein sequence is also referred[2] to as a *residue*. The process of creating a protein by a genetic transcription is referred to as *gene expression*, and the entire set of proteins contained in an organism is referred to as the complete *proteome*. The human proteome is known to contain about 20,000 proteins, each of which is created by a different gene. The above number does not include the *splicing variants* of a protein produced by the same gene, including which the number rises to more than 90,000 proteins. It is also noteworthy that many genes do not produce proteins, but they produce RNA for other tasks as the final end product. Proteins have different functions, depending on the sequence of amino acids. Furthermore, proteins with similar functions often interact with one another at specific contact points in a network of protein-protein interactions.

In reality, the structure of proteins is much more complicated than sequences, as there is a 3-dimensional structure of these molecules. Proteins often show a folding behavior, which is critical to their structures and functions. For example, enzymes fit into substrates using a "lock-and-key mechanism," which is heavily dependent on the shape of the underlying proteins. Therefore, the problem of inferring the structures of proteins has been of significant interest in the broader literature. The *primary structure* of a protein corresponds to its sequence information, which comprises the identity and order of the amino acid residues. The *secondary structures* are caused by folding patterns stabilized by intermolecular hydrogen bonds; these correspond to α-helices and β-sheets. The *tertiary structure* corresponds to how the proteins react to the aqueous environment surrounding them. In some cases, particular portions of the protein (the hydrophilic side) prefer to face toward the aqueous environment, whereas other portions (the hydrophobic side) prefer to face away from the aqueous environment. Other proteins are apathetic to the aqueous environment surrounding them. The connections between DNA and proteins lead to interesting lines of research because many diseases are caused by defects in proteins, which in turn arise because of unusual variants in DNA [Alipanahi et al. 2015].

3.3 Protein-Protein Interaction Networks

An important point here is that most proteins do not act in isolation in carry out their functions. Rather, most proteins act in concert via interacting with one another in the form of protein-protein interaction networks (PPI networks) [Szklarczyk et al. 2014; Von Mering et al. 2002]. At its core, these are networks in which the nodes correspond to proteins, and the edges correspond to interactions among them. Protein-protein interactions correspond to physical contacts between two or more protein molecules, and they enable the creation of large interaction networks.

An important problem in computational biology is that of protein-protein interaction and function prediction. Numerous conventional machine learning methods have been used for protein-protein interaction prediction [Qi et al. 2006]. Three typical types of problems are observed in this domain:

1. In the prediction of protein functions from interactions, we are given a network of protein-protein interactions, and information about the functions of some subsets of the proteins. Using this information, we would like to predict the unknown functions of proteins.

2. In the prediction of *molecular modules* [Chen et al. 2013], one is trying to isolate parts of the protein-protein interaction network that often function as a single unit. Often, such portions have large levels of interconnectivity. This problem is closely related to

[1] `https://en.wikipedia.org/wiki/FASTA_format`

[2] In general biochemistry, a monomer within a chain of a polysaccharide, protein, or nucleic acid is referred to as a residue. Therefore, an instance of adenine (A) might be residue in an RNA chain, an instance of glucose might be a residue in a complex carbohydrate like glycogen, and an amino acid is a residue in a complex protein.

that of clustering the protein-protein interaction network.

3. In protein-protein interaction prediction, we are already given some feature representations of proteins (which might include their function or sequence information) and interactions between some subsets of proteins. Given this information, one would like to predict interactions among them.

The last of these problems is quite fundamental, and it is serves as the basis of both building PPI networks and also the basis for predicting the structure of proteins. In the following sections, we will discuss each of the above problems.

Protein Function Prediction from Protein-Protein Interaction Networks

To explain the relationship between protein-protein interaction and protein function, we provide a portion of an interaction network from [Vazquez et al. 2003] in Figure 6. Here, the nodes correspond to the proteins, and the edges correspond to the interactions among proteins. The functions of some subsets of the proteins are known, whereas others are not (and shaded gray in Figure 6). An important point is that protein-protein networks show the property of homophily, wherein proteins with similar function are often connected. Using this fact, one can infer that the protein YNL127W has the function of budding, cell polarity, and filament formation in Figure 6. Note that this type of problem can be reduced to that of *collective classification* in machine learning, wherein some subsets of the nodes in a network are labeled and other labels are inferred from them [London and Getoor 2014]. The basic idea in these methods is that the connected nodes in the network have either similar or related functions, which can be inferred in a data-driven manner by propagating (function) labels from specified nodes to unspecified nodes. Although collective classification methods can be used for problems beyond protein-protein interaction networks, some of the proposed networks have been explicitly designed to work with PPI data [Bogdanov and Singh 2010; Bilgic and Getoor 2008; Desrosiers and Karypis 2009; Wu et al. 2014]. Most of these methods either extract neighborhood features for modeling a classification problem [Bogdanov and Singh 2010; London and Getoor 2014; Neville and Jensen 2003], or they use label propagation methods [Bilgic and Getoor 2008; London and Getoor 2014; Zhu et al. 2003]. Some of the methods even use explicit and implicit edges for prediction [Xiong et al. 2013]. [Deng et al. 2003] use Markov random fields, which can be viewed as probabilistic variants of neural networks. An overview of many of the classification methods used for protein function prediction with PPI network data is provided in [Sharan et al. 2007]. The similarity of functions between connected proteins has frequently been used to predict function [Schwikowski et al. 2000; Vazquez et al. 2003]. [Vazquez et al. 2003] design an energy function based on the similarities between proteins to make predictions.

Although the vast majority of techniques for function prediction use traditional machine learning techniques[Bogdanov and Singh 2010; Desrosiers and Karypis 2009; Qi et al. 2006; Sharan et al. 2007; Vazquez et al. 2003], recent progress has been made on the use of deep learning methods, especially when the function is related to a 3-dimensional

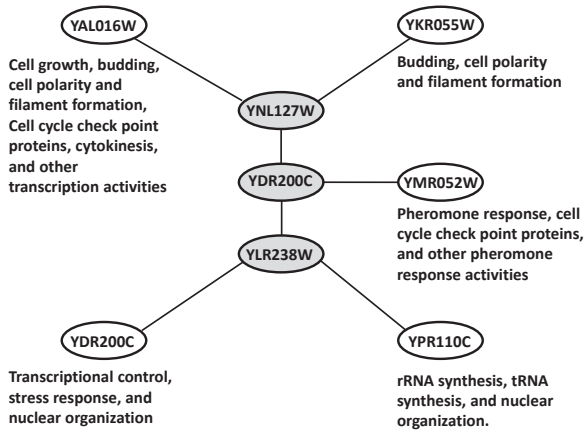


Figure 6: A portion of a protein-protein interaction networks [Vazquez et al. 2003]

structure [Wang et al. 2017]. A natural approach to modeling protein-protein interaction networks with deep learning is to create node embeddings of the interaction network with the use of structure of both the network and the protein itself [Grover and Leskovec 2016; Kipf and Welling 2016a; Hamilton et al. 2017a; Huang et al. 2017]. Indeed, some of these embedding methods [Kipf and Welling 2016a; Hamilton et al. 2017a] have been shown to perform very well for various collective classification settings, and some of them even take attributes into account [Huang et al. 2017] (as is common in PPI interaction networks). Furthermore, a graph convolutional network model has been proposed [Kipf and Welling 2016b] for predicting the link interactions among the nodes in a network. However, the methods in [Kipf and Welling 2016a; Kipf and Welling 2016b; Hamilton et al. 2017a] have mostly been designed in the context of social networks; they have not yet been tested in the computational biology (or PPI interaction/function prediction) settings. Most likely, their use in the PPI interaction prediction setting would require further modifications. As such, these methods represent a natural avenue for further research in the area. An overview of various types of node-embedding techniques may be found in [Hamilton et al. 2017b].

Integrating Sequence Information for Protein Function Inference

One can predict protein function from sequence and structure [Lee et al. 2007]. The main difference is that the network of interactions is not used in this case, although some of the methods extract the protein-protein interaction information and use it to develop features. A key point is that a substantial similarity in the underlying sequences is often used directly or indirectly in order to discover interactions [Liao and Noble 2003]. Often, there are subtle similarities in the sequence of amino acids when two proteins interact. Consequently, sequence-to-sequence alignment and similarity algorithms, such as BLAST and FASTA [Li and Homer 2010], have remained the methods of choice. An overview of machine learning methods for predicting protein functions from sequence and structure may be found in [Watson et al. 2005].

Numerous methods have also been proposed for predicting protein function from sequences [Cao et al. 2017; Kulmanov et al. 2017; Liu 2017; Murvai et al. 2001; Tang et al. 2018]. Some of this work uses traditional machine learning techniques, such as support vector machines [Tang et al. 2018] to perform the learning. This approach cannot use sequence information, and therefore it only uses dipeptide composition for creating features. This type of composition information is, nevertheless, sufficient to predict specific types of functions, such as finding whether a protein is a growth hormone.

An early method [Murvai et al. 2001] on the use of neural networks extracts features with the use of BLAST-based similarity [Li and Homer 2010] on the sequences and then applies a feedforward network on the extracted features. This type of approach of hand-crafted feature extraction is not in line with what one normally expects from neural networks. In most types of neural networks, one normally expects feature engineering to be done in an automated way.

The work in [Cao et al. 2017] is particularly interesting because it treats both the protein sequence and the functions of a protein as a "language." The protein sequence is referred to as "ProLan", and the function of the protein is referred to as "GOLan." The ProLan language simply segments the sequence of amino acids into a set of "words" to create a sentence, whereas "GOLan" creates a sentence of ordered identifiers based on protein functions. Given the sentences in the two languages, it is a relatively simple matter to perform the translation using a neural machine translation model. This model is not conceptually too different from the machine translation models used in systems[3], such as Google Translate. At the most basic level, this model hooks up two recurrent networks, one for each of the two languages. The first network encodes the protein language into an internal representation of the neural network, whereas the second network converts this internal representation into a sentence of the second language. The work in [Kulmanov et al. 2017] takes a somewhat different approach wherein it treats the problem as that of multilabel classification, where each possible function is a binary label of the classification problem. In this particular case, protein-protein interaction data were also used for classification purposes. The work in [Liu 2017] also takes the approach of using the protein sequence as input and the label as the output in the recurrent network.

There are several possible lines of research in function prediction. One of the most interesting avenues of research, which was briefly suggested in [Liu 2017], is the possibility of generating new proteins with specific functions with the use of hooked recurrent encoder-decoder architectures (just like a machine translation system). The training pairs could be proteins with the same function. Then, proteins with similar functions to a given protein P could be generated by inputting the protein P to the encoder, and generating the symbols of the generated protein by the decoder, just like a neural machine translation model. In a sense, the input protein provides the context to the generator.

Another direction of research has to do with how sequence information and protein-protein interaction information can be integrated to infer protein function. It is noteworthy that most of the techniques for sequence-based function classification either exclusively use sequence information, or protein-protein interaction features are extracted by using methods such as BLAST [Murvai et al. 2001]. Using hand-crafted features is not a natural approach from the perspective of neural network design. In practice, one can use the recently proposed network-embedding techniques [Grover and Leskovec 2016] (to apply them to the PPI interaction network) and combine them with the recurrent neural network techniques [Cao et al. 2017; Kulmanov et al. 2017; Liu 2017] to create an integrated and end-to-end framework for combining the sequence and the PPI information. The main challenge would be in combining the features of the embedding technique with the recurrent neural network for prediction. In this context, neural networks are ideal because they allow the fusion of the features from different inputs in a seamless way.

[3] `http://translate.google.com`

Molecular Modules in Protein-Protein Networks

An important class of tasks in protein-protein interaction analytics tries to find *molecular modules* from protein-protein interaction networks [Bader and Hogue 2003; Nepusz et al. 2012; Spirin and Mirny 2003]. These methods perform clustering on the nodes of the protein-protein interaction network to identify closely related sets of nodes. Such sets are referred to as molecular modules, and they are densely connected among themselves but sparsely connected with the remainder of the network. Such modules are typically of two types, as pointed out in [Spirin and Mirny 2003]. The first type comprises protein complexes, such as splicing machinery and transcription factors. The second type comprises dynamic functional units, such as signaling cascades and cell-cycle regulation. At the most basic level, one can use off-the-shelf community detection algorithms [Fortunato 2010] in order to identify clusters of nodes. One challenge is that the underlying protein complexes (clusters) are often overlapping [Nepusz et al. 2012]. Therefore, it is important to use clustering techniques that are robust to noise and overlap among clusters. This can be achieved to a large extent by using ensemble techniques [Asur et al. 2007]. However, most of these techniques use conventional machine learning methods rather than deep learning. However, deep learning is natural approach to use for this task because one can embed the nodes of a PPI network in multidimensional space. Once the embedding has been done, it can be used in conjunction with any off-the-shelf clustering algorithm. As a specific example, the original *node2vec* work proposed for embedding nodes uses PPI networks as one of the test data sets [Grover and Leskovec 2016]. Another approach [Tian et al. 2014] uses a sparse autoencoder in order to discover the embeddings of nodes of a PPI network. This approach works with the $n \times n$ similarity matrix S of an n-node graph. In this case, the idea is to treat S as a data set containing n instances, and each instance corresponds to the similarity of a node with other nodes. The similarity matrix is obtained by dropping the low-weight edges of the adjacency matrix and then normalizing the edges. The edges are normalized so that each edge (i, j) is normalized by the geometric mean of the weighted node degrees of nodes i and j. The sparse autoencoder is implemented by using a *sparsity penalty* in the hidden nodes. This encourages the hidden nodes to take on zero activation values. The resulting embeddings can then be clustered with any off-the-shelf algorithm. Considerable scope exists in integrating other types of data into the anal-

ysis, and an overview of such methods is found in [Chen et al. 2013]. Such integration methods are easily done in deep learning because inputs from multiple sources can be fused in a neural architecture.

3.4 Inferring Protein-Protein Interaction and Folding Structure

An important problem in the deep learning of proteins is that of predicting the 3-dimensional folding structure of proteins. The amino acid sequences comprising proteins regulate its 3-dimensional structure, depending on where the residues make contacts with one another. An important intermediate step in the prediction of the structure is the prediction of both intra-protein residue-residue interaction as well as inter-protein residue-residue contacts between a pair of interacting proteins (i.e., inter-protein contact prediction) [Zeng et al. 2018; Zhou et al. 2017]. Inferring these contact points can help in inferring the 3-dimensional structure of the protein.

An important point is that the functioning of a protein is heavily dependent on the shape taken on by the protein. For example, the specificity of enzymes to substrates is heavily dependent on the shape of the protein because the 3-d shape of the enzyme decides which substrates it binds to. The basics of the protein folding problem are discussed in [Dill et al. 2008]. Because of the importance of the problem, a biennial global competition was established for measuring and encouraging progress in the field in 1994. This competition is referred to as *Community Wide Experiment on the Critical Assessment of Techniques for Protein Structure Prediction (CASP)*, and an overview of some of the early work may be found in [Moult 2005]. This competition has catalyzed some of the well-known techniques in the field.

The precise shape of a protein is hard to (fully) observe at the molecular level, and one usually only has information about the sequence of amino acids comprising the protein. Although the shape can be partially observed using techniques, such as X-ray crystallography and electron microscopy, the reality is that these methods are extremely expensive and time-consuming. The precise shape of the protein depends on how this sequence of amino acids interacts with one another. Given the importance of shape in predicting the function of proteins, it is not particularly surprising that the problem of protein shape prediction has taken on a significant level of importance. In each case, one is assuming that the input is the sequence corresponding to the protein, and the output is the folding structure of the protein. Given that this approach is based on sequences, it is not particularly surprising that some of the prominent methods tackle this problem with the use of recurrent neural networks [Baldi et al. 1999]. [Baldi et al. 1999] use a recurrent neural network to predict the secondary structure associated with each position in the sequence. The problem is therefore reduced to a classification problem at each position, where one of three possibilities corresponding to α-helix, β-sheet, or coil is predicted. A bi-directional recurrent neural networks was used for prediction in this work. The work in [Spencer et al. 2015] propose methods for *ab initio* secondary structure prediction. Such methods are useful for predicting the tertiary structure of proteins. This is because the predictions about the secondary structure also feed into the predictions about the tertiary structure.

A closely related problem to that of the prediction of the structure of a protein is the problem of *contact map prediction*. In other words, the goal is to predict the interactions from one another. The shape of a protein is decided by the distance between all possible amino-acid residue pairs of the 3-dimensional protein structure. This distance structure is a simplified variant of the actual 3-dimensional structure. This is precisely the information captured in the protein contact map, which is a 2-dimensional matrix of distances, and the size of the matrix depends on the number of amino acid residues. An advantage of this type of approach is that it is more amenable to machine learning techniques. A protein contact map represents the distance between all the possible amino acid residue pairs of a 3-dimensional protein structure using a binary 2-dimensional matrix. A common simplification is that the matrix is assumed to be binary. In other words, the value of the matrix is assumed to be 1 if the distance is below a particular threshold, and 0, otherwise. This assumption turns the distance matrix into a similarity matrix, and the problem can also be modeled with *link prediction* techniques [Lü and Zhou 2011; Martin et al. 2004] in machine learning if a subset of similarities is already known. A neural network for predicting interaction sites from sequences is proposed in [Fariselli et al. 2002]. This approach uses a straightforward neural network (i.e., a feedforward network), in which the contiguous windows of 11 amino acids (residues) are used as the input to determine whether or not the central unit (among these 11 residues) is in contact with another protein. Each of the 11 inputs is one-hot encoded as a vector of size 20 to account for the 20 possibilities of amino acids at each position.

The contact maps between proteins provide useful information in order to infer the shape. Many of the methods of 3-dimensional structure prediction break up the problem into two parts. First, the distances between all the pairs of proteins are predicted. Subsequently, these distances are used to predict the 3-dimensional structure with a different model. Some techniques also compute the angles between the bonds to better capture the 3-dimensional structure.

[Di Lena et al. 2012] use a 2-dimensional bidirectional recurrent neural networks for contact map prediction. In this approach, coarse contact maps are predicted between secondary structure elements. The 2-dimensional nature of the structure helps in integrating spatial structure in the prediction process. The recurrent network is used to process the input sequence corresponding to the residues in the protein. However, instead of raw residues, various features are extracted, such as residue features, coarse features, and alignment features. The coarse features are themselves outputs of a coarse predictive phase. [Wang et al. 2017] use convolutional neural networks for predicting the structure of proteins in terms of the bidirectional contact maps. The recent idea of deep residual networks is used for this purpose [He et al. 2016]. Although it is more common to use recurrent neural networks rather than convolutional neural networks for protein structure prediction, the work in [Wang et al. 2017] shows that it is possible to use 1-dimensional convolutions for protein structure prediction.

Recently, a deep learning method referred to as *AlphaFold* was proposed in [Evans et al. 2018]. The work in [Evans et al. 2018] predicts both the distances between pairs of residues as well as the angles between pairs of residues. A neural network was trained to predict distances between the protein residues. These distances could be used to estimate

the closeness of a proposed protein structure to the correct answer. This was achieved with a separate neural network. These probabilities were then combined into a score that estimates how accurate a proposed protein structure is. The *AlphaFold* method provided competitive performance to state of the art in the *CASP* competition.

A related problem is to examine proteins in pairwise fashion and predict whether two proteins are from the same fold. The model uses the pairwise protein features as input, including information on sequence, family, and structural features. This problem is referred to as the *fold recognition problem*. The work in [Jo et al. 2015], refers to is as DNFold, uses an ensemble of neural network and conventional machine learning methods to achieve state-of-the-art performance. The underlying models were feed-forward networks containing between three and five layers.

Integrating Text with PPI Prediction

One problem in the computational biology domain is that much of the work on protein interaction and function prediction precedes the widespread use of machine learning and deep learning techniques. Complicating this fact is the issue that there are literally thousands of proteins, and the pairwise possibilities for interactions might range in the millions; simply speaking, knowledge is distributed across a vast amoiunt of literature, and much of what we know is incomplete. As a result, the known interactions and functions of proteins are often not organized in a systematic way. One of the approaches for being able to find protein-protein interactions is to integrate text mining with the previous tasks [Cohen and Hersh 2005; Donaldson et al. 2003; Simpson and Demner-Fushman 2012]. For example, the basic idea for finding protein-protein interactions is to tag pairs of proteins in a sentence together when they are known to have an interaction. Subsequently, information extraction methods can be applied to untagged text in to discover the relationships among them. Most of the current techniques focus on using vanilla machine learning techniques, although the approach is very much suited to deep learning. In particular, the work in [Hsieh et al. 2017] shows how one can combine deep learning techniques, such as recurrent neural networks, with these models in order to discover relevant interactions. Most of the analysis is at the sentence level. As a specific example, given in [Hsieh et al. 2017], consider the following sentence: "*STAT3 selectively interacts with Smad3 to antagonize TGF-β signaling.*" Given this type of sentence (in which the protein tokens have already been isolated), a recurrent neural network should be able to infer that the three proteins (STAT3, Smad3, and TGF-β) interact with one another. This can be easily achieved by using a recurrent neural network in which the input corresponds to the sequence of words in the sentence, and two of the proteins in the sentence are marked, The output is a binary result, depending on whether or not these pairs of proteins interact. In this particular case, the classification methodology would need to be applied three times to identify whether or not each of the three pairs of proteins interact with one another. Convolutional neural networks have also been used for this purpose [Peng and Lu 2017], although they have not achieved state-of-the-art performance on the interaction prediction task.

4. GENE EXPRESSION DATA

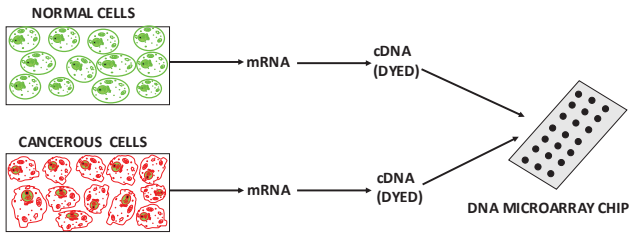


Figure 7: Capturing differential gene expression via microarrays

As discussed earlier in this survey, genes are frequently expressed in the form of the types of proteins they create. The sequence of any protein that is created is extracted from underlying DNA through the process of RNA transcription. Therefore, there is a natural connection between genomics and proteomics. An overview of machine learning methods for genomics may be found in [Libbrecht and Noble 2015].

A technology that allows biologists and machine learning scientists to analyze gene expression data is that of *microarrays*. So what is a DNA microarray? The way in which microarray chips work is that single strands of portions of the synthetic DNA are put on the chip, and when single strand of denatured DNA from human subjects (or normal control DNA) are added in, these extracted strands bind to those synthetic DNA strands containing complementary base pairs. Typically mixtures of different types of DNA are added at the same time, and they are dyed in order to be able to detect how much they bind to each position on the chip (containing a complementary strand). For example, the DNA microarray may contain "features" for both mutated DNA and normal DNA, and the control DNA would bind to the normal positions, whereas the DNA from the subject with a disease might bind to the mutated DNA. Each chip might contain thousands of short, single-strand DNA for both normal DNA and mutated DNA. This type of approach can also be used to do population studies where one determines the fraction of the population with a medical condition that also has a specific type of mutation. Because of the ability of putting a large number of features on the microarray chips (comprising large portions of the human genome), it has become possible to do large-scale studies with microarray data.

It is noteworthy that DNA microarrays have diverse applications, and one of these many applications is gene expression analysis (which is also a focus of this survey). A DNA microarray is able to perform an experiment in which each point represents the ratio of expression levels under two different experimental conditions. For example, some genes in a cell could be expressed more than others, as a result of which they will produce single-strand messenger RNA to produce proteins to perform the function corresponding to that gene. The other genes will be switched off, and will not produce messenger RNA to create functional proteins. At any given moment in time, only a subset of the genes will be producing the messenger RNA, which can be detected using microarray technology. In a gene profiling experiment, the activity levels of thousands of genes are simultaneously monitored. RNA from cancerous cells and normal cells might be also simultaneously introduced, and the differential expres-

sion with respect to the different genes will tell us which genes are active toward the disease. The messenger RNA strands are then reacted with an enzyme (such as reverse transcriptase) to produce the (single strand) complementary DNA, referred to as cDNA. These cDNA strands are dyed and reacted with the chip. The key point is that the (dyed) cDNA strands from the cancerous and normal cells will bind in a differential way to the single strands of different genes on the chip. Therefore, the color of a particular position on the DNA microarray will tell us the expression level of that gene for the two conditions (e.g., normal and cancerous). This overall process is illustrated in Figure 7. Similarly, one can compute expression levels over different conditions, such as different choices of drug treatments, by using independent experiments with different microarray chips. Another possibility is that each experiment could correspond to the gene expression of a particular patient, and the control DNA could correspond to (known) normal DNA. Therefore, by using n patients, one would have n sets of expression levels.

One repeats this process for n different experiments, which correspond to various patients or conditions for the same set of d genes. Therefore, each experiment corresponds to a data point, and the genes correspond to the features. As a result, the data for gene expression typically correspond to an $n \times d$ matrix, where each of the n rows corresponds to a single experiment (using a different microarray chip), and the d elements of that row contain the ratio of the expression levels under two conditions for all the d genes being tested. The value of d can be extremely large, and therefore, the problem is high dimensional. In many cases, it is possible to have situations where the number of dimensions is greater than the number of records, which leads to problems associated with overfitting. The features are often transformed by applying a logarithm function to each feature and then normalizing the data so that the Euclidean norm of each of the d columns is 1. In some settings, the normalization is done so that the Euclidean norm of each of the n rows is 1. The specific choice of normalization also depends on the application at hand.

Several machine learning applications could be associated with this type of data set. For example, clustering can be used to group similar genes [Gupta et al. 2015]. However, most of the known clustering methods for gene expression data do not use deep learning methods, and a survey on this can be found in [Jiang et al. 2004]. From an application-centric perspective, the classification problem is more interesting, where one is trying to classify genes based on an unknown property using the gene expression levels with labeled training data. The pioneering work in this area was proposed in [Brown et al. 2000], who used a support-vector machine in order to perform the classification of a gene expression data set with a yeast data set. Later work showed how this type of approach could be used for various diagnostic purposes, such as cancer classification [Khan et al. 2001; Guyon et al. 2002] or the discovery of pathogenic genetic variants [Quang et al. 2014]. The basic principle proposed in [Guyon et al. 2002] was to repeat an experiment for each patient to compute their expression levels for the different genes. In the following, we provide an overview of deep learning techniques for gene expression data.

4.1 Unsupervised Learning with Gene Expression Data

In the problem of clustering gene expression data, one interesting characteristic is that it is possible to pose the problem in a number of different ways depending on the application-specific scenario. For example, for an $n \times d$ gene expression matrix, one should want to cluster the dimensions when one wants to find similar genes, and one should cluster the rows when one wants to find similar experimental conditions or patients in terms of gene expression.

In the deep learning domain, the popular approach is to use an autoencoder [Aljalbout et al. 2018; Goodfellow et al. 2016] to embed the individual rows (or columns) into a featured engineered space. [Gupta et al. 2015] proposed to use denoising autoencoders to perform this feature engineering and to cluster similar genes. A broader overview of some of the clustering methods for deep learning is provided in [Ching et al. 2018]. A denoising autoencoder is like a normal autoencoder, except that additional noise is added to the input of the autoencoder to train the autoencoder in the presence of corruption [Vincent et al. 2008]. By teaching the autoencoder how to remove the effects of possible data corruption, the results are often of higher quality. The general idea of using autoencoders is helpful in extracting the latent features that can be used for a variety of tasks [Way and Greene 2017; Titus et al. 2018a]. In particular, [Titus et al. 2018a; Titus et al. 2018b] apply the approach to DNA methylation data to extract the latent features of particular methylated genes that are relevant to breast cancer. The work uses a variational autoencoder [Kingma and Welling 2013], which can also be used for unsupervised clustering.

4.2 Regression with Gene Expression Data

A recent line of work predicts gene expression levels with the use of regression methods [Chen et al. 2016]. The idea is to reduce the cost of gene expression profiling by being able to infer a subset of them.

One can technically view this problem as that of having an $n \times d$ data matrix in which only a portion of the data matrix is fully specified. For simplicity, consider the case in which the first $d_1 < d$ genes are *landmark* genes for which all n expressions are fully specified, and the remaining $(d - d_1)$ expressions are fully specified in the training data, which corresponds to the first n_1 experiments/trials. These $(d-d_1)$ target gene expressions are missing in the test data. It is often the case that the value of d might be tens of times greater than that of d_1. Consider the case in which the gene expression matrix has entries denoted by x_{ij}, where i is the index of the experiment (which was collected using a DNA microarray chip such as expression ratio between a particular patient and a control) and j is the index of a particular gene feature. Therefore, the idea is to predict the target features x_{ij} (for $j > d_1$) from the landmark features x_{ij} (for $j \leq d_1$).

$$x_{ij} = f_j(x_{i1}, \ldots x_{i,d_1}) \quad \forall j \in \{d_1 + 1, \ldots, d\} \tag{4}$$

Here, $f_j(\cdot)$ is the jth function being modeled to predict the jth target gene. Note that we do not need models for $j \in \{1 \ldots d_1\}$ because these are landmark genes for which expression levels are already available.

In the simplest case, the function $f(\cdot)$ could be a simple linear function that uses a parameter vector $\overline{w}_j$ for modeling:

$$x_{ij} = \overline{w}_j \cdot [x_{i1} \ldots x_{i,d_1}] + b_j \quad \forall j > d_1 \tag{5}$$

The weights can be learned using the training data, and

predictions can be performed on the test data.

However, using a linear regression model is too simplistic in most cases. [Chen et al. 2016] use a feed-forward neural network for this type of profiling. In this approach the expression levels for only about 1000 *landmark* genes were profiled, and those of the remaining *target* genes were inferred, which corresponded to nearly 21,000 genes. Here, the key point is that since this is multi-task regression problem, an output needed to be included for each of the 21,000 output possibilities. Therefore, the hidden layer encoded latent features that were relevant to all of the predictions.

The effect of various other factors, such as *histone modification* on gene expression have also been studied [Singh et al. 2016]. Histone modification refers to the modifications occurring to histone proteins via processes like methylation, and phosphorylation. [Singh et al. 2016] propose a deep learning model for this task, and it is shown to outperform competing methods like support vector machines.

4.3 Classification with Gene Expression Data

It is noteworthy that even though machine learning and neural networks were invented in the eighties and nineties, the use of gene expression data for machine learning picked up only in the late nineties and at the turn of the century. Like neural networks, the use of gene expression data was also a relatively new technology, and it took a while for the two fields to come together. In the early years, the classification methods were not necessarily based on neural networks but were on simpler techniques, such as support vector machines [Brown et al. 2000; Guyon et al. 2002; Dudoit et al. 2002]. As pointed out in [Dudoit et al. 2002], a lot of the early work did not use neural network methods. This is not particularly surprising because the success of deep learning is a more recent phenomenon, and the larger successes in the area occurred after 2010. One of the earliest works that used artificial neural networks in the context of gene expression data was proposed in [Khan et al. 2001], and this work showed how to separate the tumors into different diagnostic categories. However, most of these earlier works did not yield particularly exciting results, which cannot be matched by existing machine learning methods.

Where deep learning methods really score over traditional machine learning techniques are cases where good feature engineering is a possibility. In this sense, the work in [Fakoor et al. 2013] uses sparse autoencoders [Bengio et al. 2007; Coates et al. 2011] to extract features from gene expression data. Note that this is an unsupervised feature extraction approach, and therefore, it can be used in cases where the amount of unlabeled data is significant, but there are few labeled data points. The use of an autoencoder was preceded by a phase of principal component analysis for better results. For actual classification, straightforward softmax regression was used in [Fakoor et al. 2013]. This general idea of using an autoencoder and then following it up with a classifier on the extracted features has been repeated in a few places. For example, [Danaee et al. 2017] also extracts features from microarray data for classification. However, it used a denoising autoencoder instead of a sparse autoencoder.

A number of recent researchers have also focused on supervised learning. For example, [Chen et al. 2015] demonstrate how supervised learning can be used in these models. This approach is more like a conventional neural network classifier rather than an unsupervised feature engineering method that uses an autoencoder. The technique in [Yousefi et al. 2017] shows how deep learning models can be used in order to predict clinical outcomes from gene expression profiles.

4.4 Inferring Gene Networks and Their Behavior

In the beginning of this survey, we discussed the connections between genes and proteins, as well as the capturing of interactions among proteins with protein-protein interaction networks. In this section, we discuss work in the field of *gene regulatory networks*, which is a more general concept. A gene regulatory network contains a set of DNA, RNA, proteins, or their complexes as nodes, and the interactions as the edges among them. Recall from our earlier discussion that proteins are constructed using DNA/RNA using the process of transcription or translation. In addition, proteins also serve the function of "turning on" genes, which results in the creation of more proteins. These created proteins might result in further interactions and so on. Therefore, the edges in the network could represent direct chemical interactions among genes, or they could correspond to processes by which genes affect each other. Clearly, gene networks could be potentially very complex and could involve loopy nonlinear interactions over time. As a result, the temporal dynamics of such networks are sometimes modeled with differential equations [Chen et al. 1999]. However, in practice, simplified models, such as Boolean models, are often used [Akutsu et al. 1999; Shmulevich et al. 2002].

Most of the time, the gene regulatory network is not directly available, and one has to convert the gene expression data into regulatory networks. This process of inferring gene regulatory networks from gene expression data and other types of data is also referred to as *reverse engineering*. [Rubiolo et al. 2015] discover a gene regulatory network from temporal expression profiles by using a pool of multiple neural networks with temporal delays at the inputs. Each neural network discovers the potential regulator of a target gene profile at the output. [Rubiolo et al. 2017] discuss how extreme learning machines can be used in to infer gene regulatory networks. A popular approach to reconstruction of gene regulatory networks is the use of bidirectional [Biswas and Acharyya 2018] or hierarchical [Kordmahalleh et al. 2017] recurrent neural networks. Both of these methods use time-delayed temporal dynamics, which is a natural candidate for neural network modeling. Broader reviews of neural models for gene regulatory network reconstruction and analysis are provided in [Biswas and Acharyya 2016; Delgado and Gómez-Vela 2018].

The inference of gene regulatory networks is closely related to the dynamics of the interactions between genes. Recurrent network approaches to model the dynamics of gene regulatory networks are discussed in [Hu et al. 2005; Maraziotis et al. 2007]. Note that recurrent networks present a natural approach for modeling temporal dynamics because of their ability to capture interactions over time. However, more recent studies [Smith et al. 2010] have suggested that recurrent neural networks do not necessarily outperform carefully designed multilayer neural networks. Therefore, it is still an open question as to whether recurrent neural networks are the tool of choice in this setting.

Another early line of work was to use Bayesian networks [Friedman et al. 2000] to analyze gene expression data. A Bayesian network can be viewed as a special case of a neural network

which is considered a *probabilistic graphical model*. A discussion of the modeling of gene expression networks with probabilistic graphical models is provided in [Friedman 2004]. In particular, a Bayesian network uses probabilistic computations across different nodes of the network. This approach shares some similarities with the principle of probabilistic Boolean networks [Shmulevich et al. 2002].

5. SUMMARY

This paper provided a survey of algorithms for protein and genomic analyses with the use of deep learning methods. Proteomics and genomics are closely related fields, given that the blueprints for proteins are contained in genomic sequences. DNA serves as the blueprint for genes, which are transcribed into messenger RNA. These messenger RNA then provide the data needed for the creation of proteins. Consequently, many of the problems that arise in the two fields are similar. For example, both genes and proteins can be arranged into network structures based on the interactions between individual components. In the case of proteins, these networks are referred to as protein-protein interaction networks; analyzing them provides insights about protein function.

In the case of genes, a number of models have been proposed for both supervised and unsupervised learning. In supervised learning, the key models relate to the use of predicting gene expression levels. In addition, gene expression data are used in the context of classification and clustering. Finally, the inference of gene networks and their behavior provides insights into the important properties of genes and their functions.

6. REFERENCES

Akutsu, T., Miyano, S., and Kuhara, S. 1999. Identification of genetic networks from a small number of gene expression patterns under the boolean network model. In *Biocomputing'99*. World Scientific, 17–28.

Alipanahi, B., Delong, A., Weirauch, M. T., and Frey, B. J. 2015. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology 33,* 8, 831.

Aljalbout, E., Golkov, V., Siddiqui, Y., and Cremers, D. 2018. Clustering with deep learning: Taxonomy and new methods. *arXiv preprint arXiv:1801.07648*.

Angermueller, C., Pärnamaa, T., Parts, L., and Stegle, O. 2016. Deep learning for computational biology. *Molecular systems biology 12,* 7, 878.

Asur, S., Ucar, D., and Parthasarathy, S. 2007. An ensemble framework for clustering protein–protein interaction networks. *Bioinformatics 23,* 13, i29–i40.

Bader, G. D. and Hogue, C. W. 2003. An automated method for finding molecular complexes in large protein interaction networks. *BMC bioinformatics 4,* 1, 2.

Baldi, P., Brunak, S., Frasconi, P., Soda, G., and Pollastri, G. 1999. Exploiting the past and the future in protein secondary structure prediction. *Bioinformatics 15,* 11, 937–946.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. 2007. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*. 153–160.

Bilgic, M. and Getoor, L. 2008. Effective label acquisition for collective classification. In *ACM KDD Conference*. ACM, 43–51.

Bishop, C. 1995. *Neural networks for pattern recognition*. Oxford university press.

Bishop, C. M. 2006. *Pattern Recognition and Machine Learning*. Springer.

Biswas, S. and Acharyya, S. 2016. Neural model of gene regulatory network: a survey on supportive metaheuristics. *Theory in Biosciences 135,* 1-2, 1–19.

Biswas, S. and Acharyya, S. 2018. A bi-objective rnn model to reconstruct gene regulatory network: A modified multi-objective simulated annealing approach. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB) 15,* 6, 2053–2059.

Bogdanov, P. and Singh, A. K. 2010. Molecular function prediction using neighborhood features. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB) 7,* 2, 208–217.

Brown, M. P., Grundy, W. N., Lin, D., Cristianini, N., Sugnet, C. W., Furey, T. S., Ares, M., and Haussler, D. 2000. Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proceedings of the National Academy of Sciences 97,* 1, 262–267.

Cao, R., Freitas, C., Chan, L., Sun, M., Jiang, H., and Chen, Z. 2017. Prolango: protein function prediction using neural machine translation based on a recurrent neural network. *Molecules 22,* 10, 1732.

Caragea, C. and Honavar, V. 2009. Machine learning in computational biology. In *Encyclopedia of Database Systems*. Springer, 1663–1667.

Chen, B., Fan, W., Liu, J., and Wu, F.-X. 2013. Identifying protein complexes and functional modules–from static ppi networks to dynamic ppi networks. *Briefings in bioinformatics 15,* 2, 177–194.

Chen, H., Zhao, H., Shen, J., Zhou, R., and Zhou, Q. 2015. Supervised machine learning model for high dimensional gene data in colon cancer detection. In *Big Data (BigData Congress), 2015 IEEE International Congress on*. IEEE, 134–141.

Chen, T., He, H. L., and Church, G. M. 1999. Modeling gene expression with differential equations. In *Biocomputing'99*. World Scientific, 29–40.

Chen, Y., Li, Y., Narayan, R., Subramanian, A., and Xie, X. 2016. Gene expression inference with deep learning. *Bioinformatics 32,* 12, 1832–1839.

Ching, T., Himmelstein, D. S., Beaulieu-Jones, B. K., Kalinin, A. A., Do, B. T., Way, G. P., Ferrero, E., Agapow, P.-M., Zietz, M., Hoffman, M. M., et al. 2018. Opportunities and obstacles for deep learning in biology and medicine. *Journal of The Royal Society Interface 15,* 141, 20170387.

Coates, A., Ng, A., and Lee, H. 2011. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 215–223.

Cohen, A. M. and Hersh, W. R. 2005. A survey of current work in biomedical text mining. *Briefings in bioinformatics 6,* 1, 57–71.

Danaee, P., Ghaeini, R., and Hendrix, D. A. 2017. A deep learning approach for cancer detection and relevant gene identification. In *PACIFIC SYMPOSIUM ON BIOCOMPUTING 2017*. World Scientific, 219–229.

Delgado, F. M. and Gómez-Vela, F. 2018. Computational methods for gene regulatory networks reconstruction and analysis: A review. *Artificial intelligence in medicine*.

Deng, M., Zhang, K., Mehta, S., Chen, T., and Sun, F. 2003. Prediction of protein function using protein–protein interaction data. *Journal of computational biology 10,* 6, 947–960.

Desrosiers, C. and Karypis, G. 2009. Within-network classification using local structure similarity. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 260–275.

Di Lena, P., Nagata, K., and Baldi, P. 2012. Deep architectures for protein contact map prediction. *Bioinformatics 28,* 19, 2449–2457.

Dill, K. A., Ozkan, S. B., Shell, M. S., and Weikl, T. R. 2008. The protein folding problem. *Annu. Rev. Biophys. 37*, 289–316.

Donaldson, I., Martin, J., De Bruijn, B., Wolting, C., Lay, V., Tuekam, B., Zhang, S., Baskin, B., Bader, G. D., Michalickova, K., et al. 2003. Prebind and textomy–mining the biomedical literature for protein-protein interactions using a support vector machine. *BMC bioinformatics 4,* 1, 11.

Dudoit, S., Fridlyand, J., and Speed, T. P. 2002. Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American statistical association 97,* 457, 77–87.

Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*. 2224–2232.

Elman, J. L. 1990. Finding structure in time. *Cognitive science 14,* 2, 179–211.

Evans et al., R. 2018. De novo structure prediction with deep-learning based scoring. *Thirteenth Critical Assessment of Techniques for Protein Structure Prediction (Abstracts)*.

Fakoor, R., Ladhak, F., Nazi, A., and Huber, M. 2013. Using deep learning to enhance cancer diagnosis and classification. In *Proceedings of the International Conference on Machine Learning*. Vol. 28. ACM New York, USA.

Fariselli, P., Pazos, F., Valencia, A., and Casadio, R. 2002. Prediction of protein–protein interaction sites in heterocomplexes with neural networks. *European Journal of Biochemistry 269,* 5, 1356–1361.

Fortunato, S. 2010. Community detection in graphs. *Physics reports 486,* 3-5, 75–174.

Friedman, N. 2004. Inferring cellular networks using probabilistic graphical models. *Science 303,* 5659, 799–805.

Friedman, N., Linial, M., Nachman, I., and Pe'er, D. 2000. Using bayesian networks to analyze expression data. *Journal of computational biology 7,* 3-4, 601–620.

Goodfellow, I., Bengio, Y., and Courville, A. 2016. *Deep Learning*. MIT Press.

Grover, A. and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *ACM KDD Conference*. ACM, 855–864.

Gupta, A., Wang, H., and Ganapathiraju, M. 2015. Learning structure in gene expression data using deep architectures, with an application to gene clustering. In *Bioinformatics and Biomedicine (BIBM), 2015 IEEE International Conference on*. IEEE, 1328–1335.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. 2002. Gene selection for cancer classification using support vector machines. *Machine learning 46,* 1-3, 389–422.

Hamilton, W., Ying, Z., and Leskovec, J. 2017a. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

Hamilton, W. L., Ying, R., and Leskovec, J. 2017b. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*.

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

Henaff, M., Bruna, J., and LeCun, Y. 2015. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*.

Hochreiter, S. and Schmidhuber, J. 1997. Long short-term memory. *Neural computation 9,* 8, 1735–1780.

Hsieh, Y.-L., Chang, Y.-C., Chang, N.-W., and Hsu, W.-L. 2017. Identifying protein-protein interactions in biomedical literature using recurrent neural networks with long short-term memory. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Vol. 2. 240–245.

Hu, X., Maglia, A. M., and Wunsch, D. C. 2005. A general recurrent neural network approach to model genetic regulatory networks.

Huang, X., Li, J., and Hu, X. 2017. Label informed attributed network embedding. In *WSDM*. 731–739.

Jiang, D., Tang, C., and Zhang, A. 2004. Cluster analysis for gene expression data: a survey. *IEEE Transactions on knowledge and data engineering 16,* 11, 1370–1386.

Jo, T., Hou, J., Eickholt, J., and Cheng, J. 2015. Improving protein fold recognition by deep learning networks. *Scientific reports 5*, 17573.

Khan, J., Wei, J. S., Ringner, M., Saal, L. H., Ladanyi, M., Westermann, F., Berthold, F., Schwab, M., Antonescu, C. R., Peterson, C., et al. 2001. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature medicine 7,* 6, 673.

Kingma, D. P. and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kipf, T. N. and Welling, M. 2016a. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kipf, T. N. and Welling, M. 2016b. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*.

Kordmahalleh, M. M., Sefidmazgi, M. G., Harrison, S. H., and Homaifar, A. 2017. Identifying time-delayed gene regulatory networks via an evolvable hierarchical recurrent neural network. *BioData mining 10,* 1, 29.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*. 1097–1105.

Kulmanov, M., Khan, M. A., and Hoehndorf, R. 2017. Deepgo: predicting protein functions from sequence and interactions using a deep ontology-aware classifier. *Bioinformatics 34,* 4, 660–668.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86,* 11, 2278–2324.

Lee, D., Redfern, O., and Orengo, C. 2007. Predicting protein function from sequence and structure. *Nature Reviews Molecular Cell Biology 8,* 12, 995.

Li, H. and Homer, N. 2010. A survey of sequence alignment algorithms for next-generation sequencing. *Briefings in bioinformatics 11,* 5, 473–483.

Liao, L. and Noble, W. S. 2003. Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships. *Journal of computational biology 10,* 6, 857–868.

Libbrecht, M. W. and Noble, W. S. 2015. Machine learning applications in genetics and genomics. *Nature Reviews Genetics 16,* 6, 321.

Liu, X. 2017. Deep recurrent neural network for protein function prediction from sequence. *arXiv preprint arXiv:1701.08318*.

London, B. and Getoor, L. 2014. Collective classification of network data. *Data Classification: Algorithms and Applications 399*.

Lü, L. and Zhou, T. 2011. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications 390,* 6, 1150–1170.

Maraziotis, I., Dragomir, A., and Bezerianos, A. 2007. Gene networks reconstruction and time-series prediction from microarray data using recurrent neural fuzzy networks. *IET Systems Biology 1,* 1, 41–50.

Martin, S., Roe, D., and Faulon, J.-L. 2004. Predicting protein–protein interactions using signature products. *Bioinformatics 21,* 2, 218–226.

Moult, J. 2005. A decade of casp: progress, bottlenecks and prognosis in protein structure prediction. *Current opinion in structural biology 15,* 3, 285–289.

Murvai, J., Vlahoviček, K., Szepesvári, C., and Pongor, S. 2001. Prediction of protein functional domains from sequences using artificial neural networks. *Genome research 11,* 8, 1410–1417.

Nepusz, T., Yu, H., and Paccanaro, A. 2012. Detecting overlapping protein complexes in protein-protein interaction networks. *Nature methods 9,* 5, 471.

Neville, J. and Jensen, D. 2003. Collective classification with relational dependency networks. In *Second International Workshop on Multi-Relational Data Mining*. 77–91.

Noble, W. S. et al. 2004. Support vector machine applications in computational biology. *Kernel methods in computational biology*, 71–92.

Peng, Y. and Lu, Z. 2017. Deep learning for extracting protein-protein interactions from biomedical literature. *arXiv preprint arXiv:1706.01556*.

Qi, Y., Bar-Joseph, Z., and Klein-Seetharaman, J. 2006. Evaluation of different biological data and computational classification methods for use in protein interaction prediction. *Proteins: Structure, Function, and Bioinformatics 63,* 3, 490–500.

Quang, D., Chen, Y., and Xie, X. 2014. Dann: a deep learning approach for annotating the pathogenicity of genetic variants. *Bioinformatics 31,* 5, 761–763.

Ravì, D., Wong, C., Deligianni, F., Berthelot, M., Andreu-Perez, J., Lo, B., and Yang, G.-Z. 2017. Deep learning for health informatics. *IEEE journal of biomedical and health informatics 21,* 1, 4–21.

Rubiolo, M., Milone, D. H., and Stegmayer, G. 2015. Mining gene regulatory networks by neural modeling of expression time-series. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB) 12,* 6, 1365–1373.

Rubiolo, M., Milone, D. H., and Stegmayer, G. 2017. Extreme learning machines for reverse engineering of gene regulatory networks from expression time series. *Bioinformatics 34,* 7, 1253–1260.

Schölkopf, B., Tsuda, K., and Vert, J.-P. 2004. *Kernel methods in computational biology*. MIT press.

Schwikowski, B., Uetz, P., and Fields, S. 2000. A network of protein–protein interactions in yeast. *Nature biotechnology 18,* 12, 1257.

Sharan, R., Ulitsky, I., and Shamir, R. 2007. Network-based prediction of protein function. *Molecular systems biology 3,* 1, 88.

Shmulevich, I., Dougherty, E. R., and Zhang, W. 2002. From boolean to probabilistic boolean networks as models of genetic regulatory networks. *Proceedings of the IEEE 90,* 11, 1778–1792.

Simpson, M. S. and Demner-Fushman, D. 2012. Biomedical text mining: a survey of recent progress. In *Mining text data*. Springer, 465–517.

Singh, R., Lanchantin, J., Robins, G., and Qi, Y. 2016. Deepchrome: deep-learning for predicting gene expression from histone modifications. *Bioinformatics 32,* 17, i639–i648.

Smith, M. R., Clement, M., Martinez, T., and Snell, Q. 2010. Time series gene expression prediction using neural networks with hidden layers.

Spencer, M., Eickholt, J., and Cheng, J. 2015. A deep learning network approach to ab initio protein secondary structure prediction. *IEEE/ACM transactions on computational biology and bioinformatics (TCBB) 12,* 1, 103–112.

Spirin, V. and Mirny, L. A. 2003. Protein complexes and functional modules in molecular networks. *Proceedings of the National Academy of Sciences 100,* 21, 12123–12128.

Szklarczyk, D., Franceschini, A., Wyder, S., Forslund, K., Heller, D., Huerta-Cepas, J., Simonovic, M., Roth, A., Santos, A., Tsafou, K. P., et al. 2014. String v10: protein–protein interaction networks, integrated over the tree of life. *Nucleic acids research 43,* D1, D447–D452.

Tang, H., Zhao, Y.-W., Zou, P., Zhang, C.-M., Chen, R., Huang, P., and Lin, H. 2018. Hbpred: a tool to identify growth hormone-binding proteins. *International Journal of Biological Sciences 14,* 8, 957–964.

Tian, F., Gao, B., Cui, Q., Chen, E., and Liu, T.-Y. 2014. Learning deep representations for graph clustering. In *AAAI*. 1293–1299.

Titus, A. J., Bobak, C. A., and Christensen, B. C. 2018a. A new dimension of breast cancer epigenetics. *Joint Conference on Biomedical Engineering Systems Technology*.

Titus, A. J., Wilkins, O. M., Bobak, C. A., and Christensen, B. C. 2018b. An unsupervised deep learning framework with variational autoencoders for genome-wide dna methylation analysis and biologic feature extraction applied to breast cancer. *bioRxiv*, 433763.

Vazquez, A., Flammini, A., Maritan, A., and Vespignani, A. 2003. Global protein function prediction from protein-protein interaction networks. *Nature biotechnology 21,* 6, 697.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*. ACM, 1096–1103.

Von Mering, C., Krause, R., Snel, B., Cornell, M., Oliver, S. G., Fields, S., and Bork, P. 2002. Comparative assessment of large-scale data sets of protein–protein interactions. *Nature 417,* 6887, 399.

Wang, J. T., Zaki, M. J., Toivonen, H. T., and Shasha, D. 2005. Introduction to data mining in bioinformatics. In *Data Mining in Bioinformatics*. Springer, 3–8.

Wang, S., Sun, S., Li, Z., Zhang, R., and Xu, J. 2017. Accurate de novo prediction of protein contact map by ultra-deep learning model. *PLoS computational biology 13,* 1, e1005324.

Watson, J. D., Laskowski, R. A., and Thornton, J. M. 2005. Predicting protein function from sequence and structural data. *Current opinion in structural biology 15,* 3, 275–284.

Way, G. P. and Greene, C. S. 2017. Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders. *bioRxiv*, 174474.

Webb, S. 2018. Deep learning for biology. *Nature 554,* 7693, 555–557.

Wu, Q., Ye, Y., Ho, S.-S., and Zhou, S. 2014. Semi-supervised multi-label collective classification ensemble for functional genomics. *BMC genomics 15,* 9, S17.

Xiong, W., Liu, H., Guan, J., and Zhou, S. 2013. Protein function prediction by collective classification with explicit and implicit edges in protein-protein interaction networks. *BMC bioinformatics 14,* 12, S4.

Yousefi, S., Amrollahi, F., Amgad, M., Dong, C., Lewis, J. E., Song, C., Gutman, D. A., Halani, S. H., Vega, J. E. V., Brat, D. J., et al. 2017. Predicting clinical outcomes from large scale cancer genomic profiles with deep survival models. *Scientific Reports 7,* 1, 11707.

Yue, T. and Wang, H. 2018. Deep learning for genomics: A concise overview. *arXiv preprint arXiv:1802.00810*.

Zeng, H., Wang, S., Zhou, T., Zhao, F., Li, X., Wu, Q., and Xu, J. 2018. Complexcontact: a web server for inter-protein contact prediction using deep learning. *Nucleic acids research*.

Zhou, T., Wang, S., and Xu, J. 2017. Deep learning reveals many more inter-protein residue-residue contacts than direct coupling analysis. *bioRxiv*, 240754.

Zhu, X., Ghahramani, Z., and Lafferty, J. D. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML Conference*. 912–919.