Deep Learning and Hierarchical Generative Models

Elchanan Mossel*
MIT
August 2, 2017

Abstract

In this paper we propose a new prism for studying deep learning motivated by connections between deep learning and evolution. Our main contributions are:

- We introduce a sequence of increasingly complex hierarchical generative models which interpolate between standard Markov models on trees (phylogenetic models) and deep learning models.
- Formal definitions of classes of algorithms that are not deep.
- Rigorous proofs showing that such classes are information theoretically much weaker than optimal “deep” learning algorithms. In our models, deep learning is performed efficiently and proven to classify correctly with high probability.

All of the models and results are in the semi-supervised setting. Some open problems and future directions are presented.

1 Introduction

We assume that the reader is familiar with the basic concepts and developments in deep learning. We do not attempt to summarize the big body of work studying neural networks and deep learning. We refer readers who are unfamiliar with the area to [9] and the references within.

The approach presented in this paper is motivated by connections between deep learning and evolution. Some of the most successful applications of deep learning are in labelling objects that are generated in an evolutionary fashion such as the identification of animals and breeds from images, see

*Supported by ONR grant N00014-16-1-2227 and NSF grants CCF-1665252 and DMS-1737944 elmos@mit.edu
e.g. [13] and the reference within. Languages also evolve and the success of deep learning in natural language processing tasks also highlights the connection between evolution and deep learning. A final obvious connection is that deep networks are models of neural networks which evolve.

Let us consider the problem of identifying species from images. One form of this problem was tackled by Darwin. In his *Evolution of Species*, Darwin used phylogenetic trees to summarize the evolution of species [4]. The evolutionary tree in turn helps in identifying observed species. The problem of species identification took a new twist in the DNA age, where morphological characters of species were replaced by DNA sequences as the data for inference of the relationship between species.

The problem of reconstruction of phylogenetic trees from DNA sequences is a central one in molecular biology, see e.g. [8, 21]. By now, there is a well developed theory of evolutionary models, as well as a comprehensive theory of how well such models can be reconstructed both information theoretically and algorithmically see e.g. [6, 15, 16] and follow up work.

Here we attempt to extend the phylogenetic theory from the problem of reconstructing trees based on DNA sequences to reconstructing relationships based on different types of representations with the ultimate goal of understanding “real” representations such as images, sentences etc.

In what follows we introduce a family of models. We start with the phylogenetic model. The phylogenetic model we use, the symmetric Markov model, is a classical model. However, we study it from a new perspective:

- First - in addition to a DNA sequence, each node of the tree is associated with a label where different nodes might have the same label. For example, a node with a specific DNA sequence might have the label,”dog’’ or ”mammal”.

- Second - we are interested in the semi-supervised learning problem, where the labels of a small subset of the data are known and the goal is to recover the labels of the remaining data.

We then consider variants of the phylogenetic model which have additional features:

- Change of representation. In phylogenetic models, the representation is given by the DNA sequences (or RNA, proteins etc.), while it seems like in many deep learning situations, there isn’t necessarily a canonical representation. We model this by introducing a permutation on the alphabet between every node and each of its descendants.
• Interaction between features. In classical phylogenetic models each letter evolves independently, while in most deep learning scenarios, interaction between features is key. We introduce a model that captures this property.

In order to establish the power of deep learning, we consider two types of limited algorithms (which are not "deep"):

• **Local algorithms.** Such algorithms have to determine the label of each data point based on the labelled data only. The notion of local algorithms is closely related to the notion of supervised learning. Note, however, that local algorithms do not output a classifier after observing the labelled data; instead for each sample of unlabelled data we run the local algorithm.

• **Shallow Algorithms.** These algorithms only use summary statistics for the labelled data. In other words, such algorithms are not allowed to utilize the correlations between different features of the labelled data.

In our main results we provide statistical lower bounds on the performance of local and shallow algorithms. We also provide efficient algorithms that are neither shallow nor local. Thus our results provide a formal interpretation of the power of deep learning. In the conclusion we discuss a number of research directions and open problems.

1.1 Related Work

Our work builds on work in theoretical phylogenetics with the aim of providing a new theoretical perspective on deep learning. A main novel feature of our generative models is that they include both representation and labels. In contrast, most of the work in the deep learning literature focuses on the encoding within the deep network. Much of the recent work in deep learning deals with the encoding of data from one level to the next. In our model we avoid this (important) aspect by considering only representations that are essentially 1 to 1 (if we exclude the effect of the noise, or “nuisance variables”). Thus our main focus is in obtaining rigorous results relating multi-level hierarchical models and classes of supervised algorithms whose goal is to label data generated from the models.

A main theme of research in the theory of deep networks is studying the expressive power of bounded width networks in terms of their depth, see e.g. [3, 5, 22]. In our model, we result of similar spirit by showing that if the
labelled data is compressed in a way that removes the correlation between features, then learning is information theoretically impossible.

We would also like to point out the positive results of [1], where it shown that certain random deep networks can be learned. Similarly to the current work and to previous work in phylogenetics, e.g., [15] [16], the algorithm for network reconstruction is recursive and performed level by level. But the two works are very different: our results introduce a new generative model which include labels and prove both lower and upper bounds on different families of algorithms, while [1] studies a random sparse (as graph and in representation) instance of a standard deep learning network and show it can be learned efficiently.

The idea of modeling the different layers of a deep learning network as a Markov chain is not new. For example, it is utilized in [23], where information theory arguments are used to analyze the behavior of deep networks in a setting where deeper layers have fewer nodes and using the standard deep learning encoding from one level to the next. This is in contrast to our model here, where all layers are of the same size and where mutations are part of the model.

Another tree Markov chain modeling approach to deep nets is given in a contemporary work [19] (which we learned about after the first version of this paper was posted). In [19] the authors introduce a natural tree Markov model of latent variables and show that EM optimization for this model results in a standard back propagation update algorithm for deep nets in convolutional networks.

1.2 Acknowledgements

We thank Ohad Shamir, Shai Shalev-Shwartz, Nadav Cohen and Moritz Hardt for interesting discussions on deep learning. Thanks to Vishesh Jain for comments on the first draft of the manuscript.

2 Hierarchical Generative Models

In this section we will define the generative models that will be discussed in the paper.

2.1 The space of all objects - a tree

All the models will be defined on a $d$-ary tree $T = (V, E)$ of $h$ levels, rooted at $v_0$. 


The assumption that the tree is regular is made for simplicity. Most of the results can be extended to many other models of trees, including random trees.

2.2 Representations

The representation is a function \( R : V \rightarrow [q]^{k} \). The representation of node \( v \in V \) is given by \( R(v) \).

In some examples, representations of nodes at different levels are of the same type. For example, if we think of \( T \) as a phylogenetic tree, then \( R(v) \) may represent the DNA sequence of \( v \). In other examples, \( R(v) \) has different meaning for nodes at different levels. For example, if \( T \) represents a corpus of images of animals, then \( R(v) \) for a deep node \( v \) may define the specie and the type of background, while \( R(v) \) at a lowest level may represent the pixels of an image of an animal.

Given a root to leaf path \( v_0, \ldots, v_\ell \), we will consider \( R(v_{\ell-1}) \) as a higher level representation of \( R(v_\ell) \). Similarly \( R(v_{\ell-2}) \) is a higher level representation of \( R(v_{\ell-1}) \) (and therefore of \( R(v_\ell) \)). Thus, each higher level representation has many descendant lower level representations. In particular, all of the representations considered are derived from \( R(v_0) \).

2.3 Labels

Each node \( v \in V \) the tree has a set of labels \( L(v) \). \( L(v) \) may be empty for some nodes \( v \). We require that if \( w \) is a descendant of \( v \) then \( L(v) \subseteq L(w) \) and that every two nodes \( v_1, v_2 \) that have the same label \( \ell \) have a common ancestor \( v_3 \) label \( \ell \). In other words, the set of nodes labelled by a certain label is a node in the tree and all nodes below that node.

For example, a possible value for \( L(v) \) is \{“dog”, “germanshepherd”\}.

2.4 The inference problem

Let \( L_T \) denote the set of leaves of \( T \). Let \( S \subset L_T \). The input to the inference problem consists of the set \( \{(R(v), L(v)) : v \in S\} \) which is the labelled data and the set \( \{R(v) : v \in L_T \setminus S\} \) which is the unlabelled data.

The desired output is \( L(v) \) for all \( v \in L_T \), i.e., the labels of all the leaves of the tree.
2.5 Generative Models

We consider a number of increasingly complex generative models. While all of the models are stylized, the more advanced ones capture more of the features of “deep learning” compared to the simpler models. All of the models will be Markov models on the tree $T$ (rooted at $v_0$). In other words, for each directed edge of the tree from a parent $v$ to child $w$, we have a transition matrix $M_{v,w}$ of size $[q]^k \times [q]^k$ that determines the transition probabilities from the representation $R(v)$ to the representation $R(w)$. We consider the following models:

2.5.1 The i.i.d. Model (IIDM)

We first consider one of the simplest and most classical phylogenetic models given by i.i.d. symmetric Markov models. Special cases of this model, for $q = 2$ or $q = 4$ are some of the most basic phylogenetic evolutionary model. These models called the CFN and Jukes-Cantor model respectively \cite{11, 18, 7, 2}. The model is defined as follows: If $w$ is the parent of $v$ then for each $1 \leq i \leq k$ independently, it holds that conditioned on $R(w)$:

$$P[R(v)_i = a] = \frac{1 - \lambda}{q} + \lambda \delta(R(w)_i = a).$$

In words, for each letter of $R(w)$ independently, the letter given by the parent is copied with probability $\lambda$ and is otherwise chosen uniformly at random.

2.5.2 The Varying Representation Model (VRM)

One of the reasons the model above is simpler than deep learning models is that the representation of nodes is canonical. For example, the model above is a classical model if $R(w)$ is the DNA sequence of node $w$ but is a poor model if we consider $R(w)$ to be the image of $w$, where we expect different levels of representation to have different “meanings”. In order to model the non-canonical nature of neural networks we will modify the above representation as follows. For each edge $e = (w, v)$ directed from the parent $w$ to the child $v$, we associate a permutation $\sigma_e \in S_{[q]}$, which encodes the relative representation between $w$ and $v$. Now, we still let different letters evolve independently but with different encodings for different edges. So we let:

$$P[R(v)_i = a] = \frac{1 - \lambda}{q} + \lambda \delta(R(w)_i = \sigma_e^{-1}(a)).$$
In words, each edge of the tree uses a different representation of the set $[q]$. We say the the collection $\sigma = (\sigma_e : v \in E)$ is adversarial if $\sigma$ is chosen by an adversary. We say that it is random if $\sigma_e$ are chosen i.i.d. uniform. We say that $\sigma = (\sigma_e : v \in E)$ are shared parameters if $\sigma_e$ is just a function of the level of the edge $e$.

2.5.3 The Feature Interaction Model (FIM)

The additional property we would like to introduce in our most complex model is that of an interaction between features. While the second model introduces some indirect interaction between features emanating from the shared representation, deep nets include stronger interaction. To model the interaction for each directed edge $e = (w, v)$, we let $\sigma_e \in S_{q^2}$. We can view $\sigma_e$ as a function from $[q]^2 \to [q]^2$ and it will be useful for us to represent it as a pair of functions $\sigma_e = (f_e, g_e)$ where $f_e : [q]^2 \to [q]$ and $g_e : [q]^2 \to [q]$. We also introduce permutations $\Sigma_1, \ldots, \Sigma_h \in S_k$ which correspond to rewiring between the different levels. We then let

$$P(\tilde{R}(v)_i = a) = \frac{1 - \lambda}{q} + \lambda \delta(R(w)_i = a),$$

and

$$R(v)_{2i} = f_e(\Sigma_{|v|}(\tilde{R}(w)(2i)), \Sigma_{|v|}(\tilde{R}(w)(2i + 1))$$

$$R(v)_{2i+1} = g_e(\Sigma_{|v|}(\tilde{R}(w)(2i)), \Sigma_{|v|}(\tilde{R}(w)(2i + 1)).$$

In words, two features at the parent mutate to generate two features at the child. The wiring between different features at different levels is given by some known permutation that are level dependent. This model resembles many of the convolutional network models and our model and results easily extend to other variants of interactions between features. For technical reasons we will require that for all $i, j$ it holds that

$$\{\Sigma_j(2i), \Sigma_j(2i + 1)\} \neq \{2i, 2i + 1\}. \quad (1)$$

In other words, the permutations actually permute the letters. It is easy to extend the model and our results to models that have more than two features interact.

2.6 The parameter sharing setup

While the traditional view of deep learning is in understanding one object - which is the deep net, our perspective is different as we consider the space
of all objects that can be encoded by the network and the relations between
them. The two point of view are consistent in some cases though. We
say that the VRM model is **fixed parametrization** if the permutation $\sigma_e$
are the same for all the edges at the same level. Similarly the FIM is
**parameter shared** if the functions $(f_e, g_e)$ depend on the level of the edge $e$
only. For an FIM model with a fixed parametrization, the deep network
that is associated with the model is just given by the permutation $\Sigma_i \in S_k$
and the permutations $(f^1, g^1), \ldots, (f^h, g^h)$ only. While our results and models
are stated more generally, the shared parametrization setup deserves special
attention:

- **Algorithmically:** The shared parametrization problem is obviously easier - in particular, one expects, that as in practice, after the parameter-
ization is learned, classification tasks per object should be performed
very efficiently.

- **Lower bounds:** our lower bounds hold also for the shared parametriza-
tion setup. However as stated in conjecture \ref{conj:nor}, we expect much
stronger lower bounds for the FIM model. We expect that such lower
bound hold even in the shared parametrization setup.

### 3 Shallow, Local and Deep Learning

We define “deep learning” indirectly by giving two definition of “shallow”
learning and of “local” learning. Deep learning will be defined implicitly as
learning that is neither local nor shallow.

A key feature that is observed in deep learning is the use of correlation
between features. We will call algorithms that do not use this correlation or
use the correlation in a limited fashion shallow algorithms.

Recall that the input to the inference problem $D$ is the union of the
labelled and unlabelled data

$$D := \{(R(v), L(v)) : v \in S\} \cup \{R(v) : v \in L_T \setminus S\}.$$  

**Definition 3.1.** Let $A = (A_1, \ldots, A_j)$ where $A_i \subset [k]$ for $1 \leq i \leq j$. The
compression of the data according to $A$, denoted $C_A(D)$, is

$$C_A(D) := \{R(v) : v \in L_T \setminus S\} \cup \left( n_D(A_i, x, \ell) : 1 \leq i \leq j, x \in [q]^{|A_i|}, \ell \text{ is a label} \right),$$

where for every possible label \( \ell, 1 \leq i \leq j \) and \( x \in [q]^{|A_i|} \), we define

$$n_D(A_i, x, \ell) := \# \{v \in S : L(v) = \ell, R(v)_{A_i} = x\}.$$
The canonical compression of the data, \( C_s(D) \), is given by \( C_{A_1,\ldots,A_k}(D) \), where \( A_i = \{ i \} \) for all \( i \).

In words, the canonical compression, gives for every label \( \ell \) and every \( 1 \leq i \leq k \), the histogram of the \( i \)’th letter (or column) of the representation among all labeled data with label \( \ell \). Note that the unlabelled data is still given uncompressed.

The more general definition of compression allows for histograms of joint distributions of multiple letters (columns). Note in particular that if \( A = (A_1) \) and \( A_1 = [k] \), then we may identify \( C_A(D) \) with \( D \) as no compression is taking place.

**Definition 3.2.** We say that an inference algorithm is \( s \)-shallow if the output of the algorithm as a function of the data depends only on \( C_A(D) \) where \( A = (A_1,\ldots,A_j) \) and each \( A_i \) is of size at most \( s \). We say that an inference algorithm is shallow if the output of the algorithm as a function of the data depends only on \( C_s(D) \).

In both cases, we allow the algorithm to be randomized, i.e., the output may depend on a source of randomness that is independent from the data.

We next define what local learning means.

**Definition 3.3.** Given the data \( D \), we say that an algorithm is local if for each \( R(w) \) for \( w \in L_T \setminus S \), the label of \( v \) is determined only by the representation of \( w \), \( R(w) \), and all labelled data \( \{(R(v), L(v)) : v \in S\} \).

Compared to the definition of shallow learning, here we do not compressed the labelled data. However, the algorithm has to identify the label of each unlabelled data point without access to the rest of the unlabelled data.

### 4 Main Results

Our main results include positive statements establishing that deep learning labels correctly in some regimes along with negative statements that establish that shallow or local learning do not. Combining both the positive and negative results establishes a large domain of the parameter space where deep learning is effective while shallow or local learning aren’t. We conjecture that the lower bounds in our paper can be further improved to yield a much stronger separation (see conjecture 4.7).
4.1 Main parameters

The results are stated are in terms of

- the branching rate of the tree \( d \),
- the noise level \( 1 - \lambda \),
- the alphabet size \([q]\) and
- The geometry of set \( S \) of the labelled data.

Another crucial parameter is \( k \), the length of representation. We will consider \( k \) to be between logarithmic and polynomial in \( n = d^h \).

We will also require the following definition.

**Definition 4.1.** Let \( \ell \) be a label. We say that \( \ell \) is well represented in \( S \) if the following holds. Let \( v \in V \) be the vertex closest to the root \( v_0 \) that is labelled by \( \ell \). Then there are two edge disjoint path from \( v \) to \( v_1 \in S, v_2 \in S \).

The following is immediate

**Proposition 4.2.** If the tree \( T \) is known and if \( \ell \) is well represented in \( S \) then all leaves whose label is \( \ell \) can be identified.

**Proof.** In general the set of leaves \( L_\ell \), labeled by \( \ell \) contains the leaves \( L' \) of the subtree rooted at the most common ancestor of the elements of \( S \) labelled by \( \ell \). If \( \ell \) is well represented, then \( L_\ell = L' \).

4.2 The power of deep learning

**Theorem 4.3.** Assume that \( d\lambda^2 > 1 \) or \( d\lambda > 1 + \varepsilon \) and \( q \geq q(\varepsilon) \) is sufficiently large. Assume further that \( k \geq C \log n \). Then the following holds for all three models (IIDM, VRM, FIM) with high probability:

- The tree \( T \) can be reconstructed. In other words, there exists an efficient (deep learning) algorithm that for any two representation \( R(u) \) and \( R(v) \), where \( u \) and \( v \) are leaves of the tree, computes their graph distance.

- For all labels \( \ell \) that are well represented in \( S \), all leaves labelled by \( \ell \) can be identified.
4.3 The weakness of shallow and local learning

We consider two families of lower bounds - for local algorithms and for shallow algorithms.

In both cases, we prove information theory lower bounds by defining distributions on instances and show that shallow/local algorithms do not perform well against these distributions.

4.3.1 The Instances and lower bounds

Let $h_0 < h_1 < h$. The instance is defined as follows.

**Definition 4.4.** The distribution over instances $I(h_0, h_1)$ is defined as follows. Initialize $S = \emptyset$.

- All nodes $v_1$ with $d(v_1, v_0) < h_0$ are not labelled.
- The nodes with $d(v_1, v_0) = h_0$ are labelled by a random permutation of "$1", "$2", "$3$, ... $d^{h_0}$".
- For each node $v_1$ with $d(v_1, v_0) = h_0$, pick two random descendants $v_2, v'_2$ with $d(v'_2, v_0) = d(v_2, v_0) = h_1$, such that the most common ancestor of $v_2, v'_2$ is $v_1$. Add to $S$ all leaves in $L_T$ that are descendants of $v_2$ and $v'_2$.

**Theorem 4.5.** Given an instance drawn from 4.4 and data generated from IIDM, VRM or FIM, the probability that a local algorithm labels a random leaf in $L_T \setminus S$ correctly is bounded by

$$d^{-h_0}(1 + O(k\lambda^{h-h_1}q)).$$

Note that given the distribution specified in the theorem, it is trivial to label a leaf correctly with probability $d^{-h_0}$ by assigning it any fixed label. As expected our bound is weaker for longer representations. A good choice for $h_1$ is $h_0 + 1$ (or $h_0 + 2$ if $d = 2$), while a good choice of $h_0$ is 1, where we get the bound

$$d^{-1}(1 + O(k\lambda^{h}q),$$

compared to $d^{-1}$ which can be achieved trivially.

**Theorem 4.6.** Consider a compression of the data $C_A(D)$, where $A = (A_1, \ldots, A_m)$ and let $s = \max_{j \leq m} |A_j|$. If $d\lambda^2 < 1$ and given an instance
drawn from $L_T \setminus S$, generated from IIDM, VRM or FIM, the probability that a shallow algorithm labels a random leaf in $L_T \setminus S$ correctly is at most

$$d^{-h_0} + C m d^{h_0} \exp(-c(h - h_1)),$$

where $c$ and $C$ are positive constants which depend on $\lambda$ and $s$.

Again, it is trivial to label nodes correctly with probability $d^{-h_0}$. For example if $h_0 = 1, h_1 = 2$ (or $= 3$ to allow for $d = 2$) and we look at the canonical compression $C_s(D)$, we obtain the bound $d^{-1} + O(k \exp(-ch)) = d^{-1} + O(k^{n^{\alpha}})$ for some $\alpha > 0$. Thus when $k$ is logarithmic of polylogarithmic in $n$, it is information theoretically impossible to label better than random.

4.4 Discussion

Theorem 4.5 establishes that local algorithms are inferior to deep algorithms if most of the data is unlabelled. Theorem 4.6 shows that in the regime where $\lambda^{-1} \in (\sqrt{d}, d)$ and for large enough $q$, shallow algorithms are inferior to deep algorithms. We conjecture that stronger lower bound and therefore stronger separation can be obtained for the VRM and FIM models. In particular:

**Conjecture 4.7.** In the setup of Theorem 4.6 and the VRM model, the results of the theorem extend to the regime $d \lambda^4 < 1$. In the case of the FIM model it extends to a regime where $\lambda < 1 - \phi(d, h)$, where $\phi$ decays exponentially in $h$.

5 The power of deep learning: proofs

The proof of all positive results is based on the following strategy:

- Using the representations $\{R(v) : v \in L_T\}$ reconstruct the tree $T$.

- For each label $\ell$, find the most common $w$ ancestor of $\{v : v \in L_T, R(v) \in S, L(v) = \ell\}$ and label all nodes in the subtree root at $w$ by $\ell$.

For labels that are well represented, it follows that if the tree constructed at the first step is the indeed the generative tree, then the identification procedure at the second step indeed identifies all labels accurately.

The reconstruction of the tree $T$ is based on the following simple iterative “deep” algorithm in which we iterate the following. Set $h' = h$. 

---

12
LS This step computes the Local Structure of the tree: For each \( w_1, w_2 \) with \( d(v_0, w_1) = d(v_0, w_2) = h' \), compute \( \min(d(w_1, w_2), 2r + 2) \). This identifies the structure of the tree in levels \( \min(h' - r, \ldots, h') \).

Cond If \( h' - r \leq 0 \) then EXIT, otherwise, set \( h' := h' - r \).

AR Ancestral Reconstruction. For each node \( w \) with \( d(v_0, w) = h' \), estimate the representation \( R(w) \) from all its descendants at level \( h' + r \).

This meta algorithm follows the main phylogenetic algorithm in [16]. We give more details on the implementation of the algorithm in the 3 setups.

5.1 IIDM

We begin with the easiest setup and explain the necessary modification for the more complicated ones later.

The analysis will use the following result from the theory of reconstruction on trees.

**Proposition 5.1.** Assume that \( d\lambda^2 > 1 \) or \( d\lambda > 1 + \varepsilon \) and \( q \geq q(\varepsilon) \) is sufficiently large. Then there exists \( \lambda_1 > 0 \) and \( r \) such that the following holds. Consider a variant of the IIDM model with \( r \) levels and \( k = 1 \). For each leaf \( v \), let \( R'(v) \sim \lambda_2 \delta_{R(v)} + (1 - \lambda_2)U \), where \( \lambda_2 > \lambda_1 \) and \( U \) is a uniform label. Then there exists an algorithm that given the tree \( T \), and \( (R'(v) : v \in L_T) \) returns \( R'(v_0) \) such that \( R'(v) \sim \lambda_3 \delta_{R(v)} + (1 - \lambda_3)U \) where \( \lambda_3 > \lambda_1 \).

**Proof.** For the case of \( d\lambda^2 > 1 \) this follows from [12, 17]. In the other case, this follows from [13].

Let \( \lambda(h') \) denote the quality of the reconstructed representations at level \( h' \). We will show by induction that \( \lambda(h') > \lambda_1 \) and that the distances between nodes are estimated accurately. The base case is easy as we can accurately estimate the distance of each node to itself and further take \( \lambda(h) = 1 \).

To estimate the distance between \( w_1 \) and \( w_2 \) we note that the expected hamming distance \( d_H(R(w_1), R(w_2)) \) between \( R(w_1) \) and \( R(w_2) \) is:

\[
\frac{q - 1}{q} \left( 1 - \lambda(h')^2 \lambda^{d(w_1, w_2)} \right)
\]

and moreover the Hamming distance is concentrated around the mean. Thus if \( k \geq C(\lambda', q, r) \log n \) then all distances up to \( 2r \) will be estimated accurately and moreover all other distances will be classified correctly as being larger
than $2r + 2$. This establishes that the step LS is accurate with high probability. We then apply Proposition 5.1 to recover $\hat{R}(v)$ for nodes at level $h'$. We conclude that indeed $\lambda_{h'} > \lambda_1$ for the new value of $h'$.

5.2 VRM

The basic algorithm for the VRM model is similar with the following two modifications:

- When estimating graph distance instead of the Hamming distance $d_H(R(w_1), R(w_2))$, we compute
  \[ d'_H(R(w_1), R(w_2)) = \min_{\sigma \in S_q} d_H(\sigma(R(w_1)), R(w_2)), \]  
  i.e., the minimal Hamming distance over all relative representations of $w_1$ and $w_2$. Again, using standard concentration results, we see that if $k \geq C(\lambda', q, r) \log n$ then all distances up to $2r$ will be estimated accurately. Moreover, for any two nodes $w_1, w_2$ of distance at most $2r$, the minimizer $\sigma$ in (2) is unique and equal to the relative permutation with high probability. We write $\sigma(w_1, w_2)$ for the permutation where the minimum is obtained.

- To perform ancestral reconstruction, we apply the same algorithm as before with the following modification: Given a node $v$ at level $h'$ and all of its descendants at level $r + h'$, $w_1, \ldots, w_{h'd}$. We apply the reconstruction algorithm in Proposition 5.1 to the sequences
  \[ R(w_1), \sigma(w_1, w_2)(R(w_2)), \ldots, \sigma(w_1, w_{h'd})(R(w_{h'd})), \]  
  where recall that $\sigma(w_1, w_j)$ is the permutation that minimizes the Hamming distance between $R(w_1)$ and $R(w_j)$. This will insure that the sequence $\hat{R}(v)$ has the right statistical properties. Note that additionally to the noise in the reconstruction process, it is also permuted by $\sigma(w_1, v)$.

5.3 FIM

The analysis of FIM is similar to VRM. The main difference is that while in VRM model, we reconstructed each sequence up to a permutation $\sigma \in S_q$, in the FIM model there are permutations over $S_{q^2}$ and different permutations do not compose as they apply to different pairs of positions. In order to overcome this problem, as we recover the tree structure, we also recover the
permutation \((f_v, g_v)\) up to a permutation \(\sigma \in S_q\) that is applied to each letter individually.

For simplicity of the arguments, we assume that \(d \geq 3\). Let \(w_1, w_2, w_3\) be three vertices that are identified as siblings in the tree and let \(w\) be their parent. We know that \(R(w_1), R(w_2)\) and \(R(w_3)\) are noisy versions of \(R(w)\), alas composed with permutations \(\tau_1, \tau_2, \tau_3\) on \(S_q^2\).

We next apply concentration arguments to learn more about \(\tau_1, \tau_2, \tau_3\). To do so, recall that \(k \geq C(\lambda) \log n\). Fix \(x = (x_1, x_2) \in [q]^2\), and consider all occurrences of \(\tau_1(x)\) in \(R(w_1)\). Such occurrences are correlated with \(\tau_2(x)\) in \(R(w_2)\) and \(\tau_3(x)\) in \(R(w_3)\). We now consider occurrences of \(\tau_2(x)\) in \(R(w_2)\) and \(\tau_3(x)\) in \(R(w_3)\). Again the most common co-occurrence in \(w_1\) is \(\tau_1(x)\). The following most likely occurrences values will be the \(2q - 1\) values \(y\) obtained as \(\tau_1(x_1, z_2)\), or \(\tau_1((z_1, x_2))\) where \(z_1 \neq x_1, z_2 \neq x_2\).

In other words, for each value \(\tau_1(x)\) we recover the set

\[
A(x) = B(x) \cup C(x),
\]

where

\[
B(x) = \{\tau_1(x_1, z_2) : z_2 \neq x_2\}, \quad C(x) = \{\tau_1(z_1, x_2) : z_1 \neq x_1\}.
\]

Note that if \(y^1, y^2 \in B(x)\) then \(y^2 \in A(y_1)\) but this is not true if \(y^1 \in B(x)\) and \(y^2 \in C(x)\). We can thus recover for every value \(\tau_1(x)\) not only the set \(A(x)\) but also its partition into \(B(x)\) and \(C(x)\) (without knowing which one is which).

Our next goal is to recover

\[
\{(x, B(x)) : x \in [q]^2\}, \quad \{(x, C(x)) : x \in [q]^2\}
\]

up to a possible global flip of \(B\) and \(C\). In order to do so, note that

\[
\{x\} \cup B(x) \cup_{y \in C(x)} B(y) = [q]^2,
\]

and if any of the \(B(y)\) is replaced by a \(C(y)\), this is no longer true. Thus once we have identified \(B(y)\) for one \(y \in C(x)\), we can identify \(B(y)\) for all \(y \in C(x)\). Repeating this for different values of \(x\), recovers the desired \(B\) and \(C\).

We next want to refine this information even further. WLOG let \(x = \tau_1(0, 0)\) and let \(y = \tau_1(a, 0) \in C(x)\). and \(z = (0, b) \in B(x)\). And note that \(C(y) \cap B(z)\) contains a single element, i.e., \(\tau_1(a, b)\). We have thus recovered \(\tau_1\) up to a permutation of \(S\) as needed.

After recovering \(\tau_1, \tau_2, \tau_3\) etc. we may recover the ancestral state at their parent \(w\) up to the following degrees of freedom (and noise)
• A permutation of $S_q$ applied to each letter individually.
• A global flip of the sets $B$ and $C$.

In general the second degree of freedom cannot be recovered. However, if $v_2$ is a sister of $v_1$ (with the same degrees of freedom), then only the correct choice of the $B/C$ flips will minimize the distance defined by taking a minimum over permutations in $S_q^2$. Thus by a standard concentration argument we may again recover the global $B/C$ flip and continue recursively. Note that this argument is using condition (1).

6 The limited power of limited algorithms

6.1 The Limited Power of Local Algorithms

To prove lower bounds it suffices to prove them for the IIDM model as is a special case of the more general models. We first prove Theorem 4.5.

Proof. Let $R(w)$ be an unlabelled leaf representation. Let $M = d^{h_0}$. Let $u_1, \ldots, u_M$ denote the nodes level $h_0$ and denote their labels by $\ell_1, \ldots, \ell_M$. Let $v_i, v'_i$ denote the nodes below $u_i$ at level $h_1$ with the property that the leaves of the tree rooted at $v_i$ are the elements of $S$ with label $\ell_i$.

Let $u_i$ be the root of the tree $w$ belongs to and let $x_i$ be the lowest intersection between the path from $w$ to $u_i$ and the path between $v_i$ and $v'_i$. We write $h'$ for $d(w, x_i)$. Note that $h' \geq h - h_1$. For $j \neq i$ let $x_j$ be the node on the path between $v_j$ and $v'_j$ such that $d(v_j, x_j) = d(v_i, x_i)$. We assume that in addition to the labelled data we are also given $h'$ and $D' = (\ell_1, R(x_1)), \ldots, (\ell_M, R(x_m))$.

Note that we are not given the index $i$.

Of course having more information reduces the probability of error in labelling $R(w)$. However, note that $R(w)$ is independent of $\{(R(v), L(v)) : v \in S\}$ conditioned on $D'$ and $h'$. It thus suffices to upper bound the probability of labelling $R(w)$ correctly given $D'$. By Bayes:

$$P[L(w) = \ell_i | D', h'] = \frac{P[R(w) | D', L(w) = \ell_i, h']}{\sum_{j=1}^{M} P[R(w) | D', L(w) = \ell_j, h']} = \frac{P[R(w) | R(x_i), h']}{\sum_{j=1}^{M} P[R(w) | R(x_j), h']}$$

We note that

$$\left( \frac{(1 - \lambda h')/q}{(\lambda h' + (1 - \lambda h')/q)} \right)^k \leq \frac{P[R(w) | R(x_i), h']}{P[R(w) | R(x_j), h']} \leq \left( \frac{(\lambda h' + (1 - \lambda h')/q)}{(1 - \lambda h')/q} \right)^k$$

16
So the ratio is

\[ 1 + O(k\lambda^{h'}q) = 1 + O(k\lambda^{h-h_1}q) \]

and therefore the probability of correct labelling is bounded by

\[ \frac{1}{M}(1 + O(k\lambda^{h-h_1}q)) \]

as needed.

6.2 On count reconstruction

We require the following preliminary result in order to bound the power of local algorithms.

**Lemma 6.1.** Consider the IIDM with \( d\lambda^2 < 1 \) and assume that all the data is labelled and that is compressed as \( C_A(D) \), where \( A = (A_1) \) and \( A_1 = [k] \), i.e., we are given the counts of the data. Let \( P_{x,h} \) denote the distribution of \( C_A(D) \) conditional on \( R(v_0) = x \). There there exists a distribution \( Q = Q^h \) such that for all \( x \), it holds and

\[ P_{x,h} = (1 - \eta)Q + \eta P'_{x,h} \quad \eta \leq C \exp(-ch), \quad (3) \]

where \( Q \) is independent of \( x \) and \( c, C \) are two positive constant which depend on \( \lambda \) and \( k \) (but not on \( h \)).

Our proof builds on the a special case of the result for \( k = 1 \), where \[17\] show that the “count reconstruction problem is not solvable” which implies the existence of \( \eta(h) \) which satisfies \( \eta(h) \to 0 \) as \( h \to \infty \). The statement above generalizes the result to all \( k \). Moreover, we obtain an exponential bound on \( \eta \) in terms of \( h \).

**Proof.** Assume first that \( k = 1 \). The proof that the threshold for “count reconstruction is determined by the second eigenvalue” \[17\] implies the statement of the lemma with a value \( \eta = \eta(h) \) which decays to 0 as \( h \to \infty \). Our goal in the lemma above is to obtain a more explicit bound showing an exponential decay in \( h \). Such exponential decay follows from \[10\] for a different problem of robust reconstruction. Robust reconstruction is a variation of the reconstruction problem, where the tree structure is known but the value of each leaf is observed with probability \( \delta > 0 \), independently for each leaf. \[10\] proved that if \( d\lambda^2 < 1 \) and if \( \delta(d, \lambda) > 0 \) is small enough then
the distributions $S_x$ of the partially observed leaves given $R(v_0) = x$ satisfy

$$S_x = (1 - \eta)S + \eta S_{X}^\prime, \quad \eta \leq C \exp(-ch).$$

(4)

From the fact that the census reconstruction problem is not solvable [17], it follows that there exists a fixed $h' = h'(\delta)$ such that for the reconstruction problem with $h'$ levels, the distribution of the counts at the leaves can be coupled for all root values except with probability $\delta$. We can now generate $P_{x,h+h'}$ as follows: we first generate $Q_{x,h}$ which is the representations at level $h$. Then each node at level $h$ is marked as coupled with probability $1 - \delta$ and uncoupled with probability $\delta$ independently. To generate the census $P_{x,h+h'}$ from $Q_{x,h}$, as follows: for each coupled node at level $h$, we generate the census of the leaves below it at level $h+h'$ conditioned on the coupling being successful (note that this census is independent of the representation of the node at level $h$). For uncoupled nodes, we generate the census, conditioned on the coupling being unsuccessful. From the description it is clear that $P_{x,h+h'}$ can be generated from $Q_{x,h}$. Since $Q_{x,h}$ has the representation (4), it now follows that $P_{x,h+h'}$ has the desired representation (3) as needed.

6.3 The limited power of shallow algorithms

We now prove Theorem 4.6.

Proof. The idea of the proof is to utilize Lemma 6.1 to show that the compressed labelled data is essentially independent of the unlabelled data. Write $M = d^{h_0}$. For each permutation $\sigma$ of the $M$ labels $1, \ldots, M$, we write $P_\sigma$ for the induced distribution on the compressed labelled data $C_A(D)$. Our goal is to show we can write

$$P_\sigma = (1 - \eta)P + \eta P'_\sigma$$

where $\eta$ is small. Note that (5) implies that the probability of labelling a unlabelled leaf accurately is at most $M^{-1} + \eta$. Indeed we may consider a problem where in addition to the sample from $P_\sigma$ we are also told if it is coming from $P$ or from $P'_\sigma$. If it is coming from $P$, we know it is generated independently of the labels and therefore we cannot predict better than random (i.e. $M^{-1}$).

Let $R(v_1(1)), R(v_2(1)), \ldots, R(v_1(M)), R(v_2(M))$ denote the representations at the roots of the subtrees of the labelled data. Let $I$ denote all of the
representations and let $P_I$ denote the distribution of $C_A(D)$ conditioned on these representations. By convexity, to prove the desired coupling it suffices to prove

$$P_I = (1 - \eta)P + \eta P'_I$$

By applying Lemma 6.1 to each of the trees rooted at $v_1(1), v_2(1), \ldots, v_1(M), v_2(M)$ and to each of the sets $A_i$, we obtain the desired results.

\[\Box\]

7 Future directions

Future research directions include:

7.1 Stronger Lower Bounds

We expect that stronger lower bound should be achievable for the more complex models of networks such as VRM and FIM, see for example Conjecture 4.7.

7.2 Random Trees

The assumption that the generative trees are regular was made for the ease of expositions and proofs. A natural follow up step is to extend out results to the much more realistic setup of randomly generated trees.

7.3 Better models

Other than random trees, better models should include the following:

- The VRM and FIM both allow for the change of representation and for feature interaction to vary arbitrarily between different edges. More realistic models should penalize variation in these parameters. This should make the learning task easier.

- The FIM model allows interaction only between fixed nodes in one level to the next. This is similar to convolutional networks. However, for many other applications it makes sense to allow interaction with a small but varying number of nodes with preference towards certain localities. It is interesting to extend the models and results in such fashion.
• It is interesting to consider non-tree generating networks. In many applications involving vision and language it makes sense to allow to “concatenate” two or more representations. We leave such models for future work.

• As mentioned earlier, our work circumvents auto encoders and issues of overfitting by using compact, almost 1-1 dense representations. It is interesting to combine our “global” framework with “local” auto encoders.

7.4 More Robust Algorithms

The combinatorial algorithms presented in the paper assume that the data is generated accurately from the model. It is interesting to develop robust algorithm that work for data that is approximately generated from the model. In particular, it is very interesting to study if the standard optimization algorithms that are used in deep learning are as efficient in recovering the models presented here. We note that for the phylogenetic reconstruction problem, even showing that the Maximum Likelihood tree is the correct one is a highly non-trivial task and we still do not have a proof that standard algorithms for finding the tree, actually find one, see [20].

References

[1] S. Arora, A. Bhaskara, R. Ge, and T. Ma. Provable bounds for learning some deep representations. In International Conference on Machine Learning, pages 584–592, 2014.

[2] J. A. Cavender. Taxonomy with confidence. Math. Biosci., 40(3-4), 1978.

[3] N. Cohen, O. Sharir, and A. Shashua. On the expressive power of deep learning: A tensor analysis. In V. Feldman, A. Rakhlin, and O. Shamir, editors, 29th Annual Conference on Learning Theory, volume 49 of Proceedings of Machine Learning Research, pages 698–728, Columbia University, New York, New York, USA, 23–26 Jun 2016. PMLR.

[4] C. Darwin. On the origin of species. 1859.

[5] R. Eldan and O. Shamir. The power of depth for feedforward neural networks. In Conference on Learning Theory, pages 907–940, 2016.
[6] P. L. Erdős, M. A. Steel, L. A. Székely, and T. A. Warnow. A few logs suffice to build (almost) all trees (part 1). Random Structures Algorithms, 14(2):153–184, 1999.

[7] J. S. Farris. A probability model for inferring evolutionary trees. Syst. Zool., 22(4):250–256, 1973.

[8] J. Felsenstein. Inferring Phylogenies. Sinauer, New York, New York, 2004.

[9] I. Goodfellow, Y. Bengio, and A. Courville. Deep learning. MIT Press, 2016.

[10] S. Janson and E. Mossel. Robust reconstruction on trees is determined by the second eigenvalue. Ann. Probab., 32:2630–2649, 2004.

[11] T. H. Jukes and C. Cantor. Mammalian protein metabolism. In H. N. Munro, editor, Evolution of protein molecules, pages 21–132. Academic Press, 1969.

[12] H. Kesten and B. P. Stigum. Additional limit theorems for indecomposable multidimensional Galton-Watson processes. Ann. Math. Statist., 37:1463–1481, 1966.

[13] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. Nature, 521(7553):436–444, 2015.

[14] E. Mossel. Reconstruction on trees: beating the second eigenvalue. Ann. Appl. Probab., 11(1):285–300, 2001.

[15] E. Mossel. On the impossibility of reconstructing ancestral data and phylogenies. Jour. Comput. Bio., 10(5):669–678, 2003.

[16] E. Mossel. Phase transitions in phylogeny. Trans. Amer. Math. Soc., 356(6):2379–2404 (electronic), 2004.

[17] E. Mossel and Y. Peres. Information flow on trees. Ann. Appl. Probab., 13(3):817–844, 2003.

[18] J. Neyman. Molecular studies of evolution: a source of novel statistical problems. In S. S. Gupta and J. Yackel, editors, Statistical desicion theory and related topics, pages 1–27. 1971.
[19] A. B. Patel, M. T. Nguyen, and R. Baraniuk. A probabilistic framework for deep learning. In Advances in Neural Information Processing Systems, pages 2558–2566, 2016.

[20] S. Roch and A. Sly. Phase transition in the sample complexity of likelihood-based phylogeny inference. ArXiv e-prints, 2015.

[21] C. Semple and M. Steel. Phylogenetics, volume 22 of Mathematics and its Applications series. Oxford University Press, 2003.

[22] M. Telgarsky. Benefits of depth in neural networks. arXiv preprint arXiv:1602.04485, 2016.

[23] N. Tishby and N. Zaslavsky. Deep learning and the information bottleneck principle. In Information Theory Workshop (ITW), 2015 IEEE, pages 1–5. IEEE, 2015.