

Shallow-to-Deep Chaining and Hierarchical Learning beyond Constant-Depth Models

Yunwei Ren
Princeton University

Based on joint work with

Yatin Dandi



Zihao Wang



Florent Krzakala



Jason D. Lee



Motivation

- ▶ **Empirical heuristics.** Deep learning works because deep networks can **hierarchically learn** complex functions.

Motivation

- ▶ **Empirical heuristics.** Deep learning works because deep networks can **hierarchically learn** complex functions.
- ▶ **Deep learning theory.** Most optimization results concern **two-layer** networks.
 - ▶ (Relatively) easier to analyze.
 - ▶ Very different from practice.

Motivation

- ▶ **Empirical heuristics.** Deep learning works because deep networks can **hierarchically learn** complex functions.
- ▶ **Deep learning theory.** Most optimization results concern **two-layer** networks.
 - ▶ (Relatively) easier to analyze.
 - ▶ Very different from practice.

Q. Prove **optimization/hierarchical learning** guarantees for $\omega(1)$ -depth networks?

Motivation

- ▶ **Empirical heuristics.** Deep learning works because deep networks can **hierarchically learn** complex functions.
- ▶ **Deep learning theory.** Most optimization results concern **two-layer** networks.
 - ▶ (Relatively) easier to analyze.
 - ▶ Very different from practice.

Q. Prove **optimization/hierarchical learning** guarantees for $\omega(1)$ -depth networks?

Shallow-to-deep chaining principle

Correlation + clean signal + weak identifiability $\Rightarrow$ hierarchical learning

Plan

1. The shallow-to-deep chaining principle.
2. Applications:
 - ▶ Hierarchical staircase functions.
Algorithmic Separation between Constant-Depth and Logarithmic-Depth Neural Networks.
[R, Wang, Lee (2026)].
 - ▶ Certain context-free grammars.¹
Provable Learning of Random Hierarchy Models and Hierarchical Shallow-to-Deep Chaining.
[R, Dandi, Krzakala, Lee (2026)]
3. Summary and future directions

¹Check the appendix slides if you are interested.

Shallow-to-Deep Chaining

Informal Principle (Shallow-to-deep chaining)

Suppose that the target function is hierarchical. If

- (i) the output is “correlated” with the lower-level parts;
- (ii) the signal received by the lower-level parts of the learner is “clean;”
- (iii) the lower-level features are “(weakly) identifiable,”

⇒ can learn it by **layerwise training** a suitable deep network.

Shallow-to-Deep Chaining

Informal Principle (Shallow-to-deep chaining)

Suppose that the target function is hierarchical. If

- (i) the output is “correlated” with the lower-level parts;
- (ii) the signal received by the lower-level parts of the learner is “clean;”
- (iii) the lower-level features are “(weakly) identifiable,”

⇒ can learn it by **layerwise training** a suitable deep network.

- **Condition (i):** Rules out k -parity-like functions ($k \gg 1$).

Shallow-to-Deep Chaining

Informal Principle (Shallow-to-deep chaining)

Suppose that the target function is hierarchical. If

- (i) the output is “correlated” with the lower-level parts;
 - (ii) the signal received by the lower-level parts of the learner is “clean;”
 - (iii) the lower-level features are “(weakly) identifiable,”
- ⇒ can learn it by **layerwise training** a suitable deep network.

- ▶ **Condition (i):** Rules out k -parity-like functions ($k \gg 1$).
- ▶ **Condition (ii):** Lower part of the learner won't overfit the higher part of the target.
⇒ enables layerwise training.

Shallow-to-Deep Chaining

Informal Principle (Shallow-to-deep chaining)

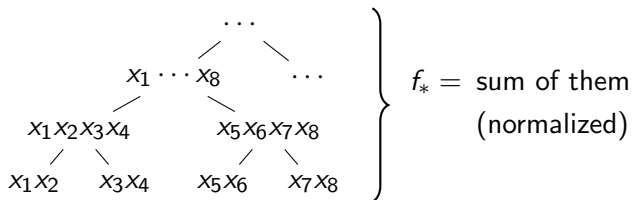
Suppose that the target function is hierarchical. If

- (i) the output is “correlated” with the lower-level parts;
 - (ii) the signal received by the lower-level parts of the learner is “clean;”
 - (iii) the lower-level features are “(weakly) identifiable,”
- ⇒ can learn it by **layerwise training** a suitable deep network.

- ▶ **Condition (i):** Rules out k -parity-like functions ($k \gg 1$).
- ▶ **Condition (ii):** Lower part of the learner won't overfit the higher part of the target.
⇒ enables layerwise training.
- ▶ **Condition (iii):** Fitting the lower portion of the target in value
⇒ learning some useful lower-level features.

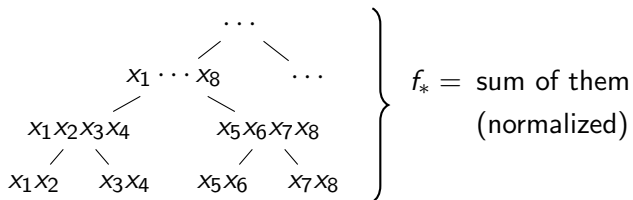
Application I: Hierarchical staircase function [R, Wang, Lee, 2026]

Running example. (base) deep quadratic function: $f_* : \{\pm 1\}^d \rightarrow \mathbb{R}$ with $d = 2^L$



Application I: Hierarchical staircase function [R, Wang, Lee, 2026]

Running example. (base) deep quadratic function: $f_* : \{\pm 1\}^d \rightarrow \mathbb{R}$ with $d = 2^L$

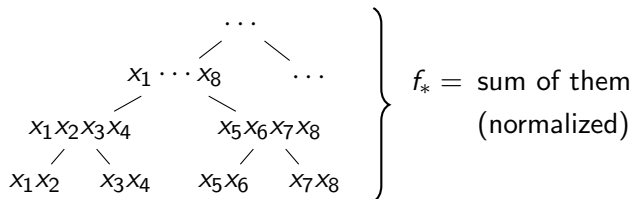


$$\begin{aligned}
 f_*(\mathbf{x})L^{1/2} = & 2^{-(L-1)/2} (x_1x_2 + x_3x_4 + \cdots + x_{d-1}x_d) \\
 & + 2^{-(L-2)/2} (x_1x_2x_3x_4 + \cdots + x_{d-3}x_{d-2}x_{d-1}x_d) \\
 & + 2^{-(L-3)/2} (x_1 \cdots x_8 + \cdots + x_{d-7} \cdots x_d) \\
 & + \cdots + 2^0 (x_1 \cdots x_d).
 \end{aligned}$$

(Coefficients chosen so that each level has the same L^2 mass)

(General deep quad. function = base + permutation in every layer)

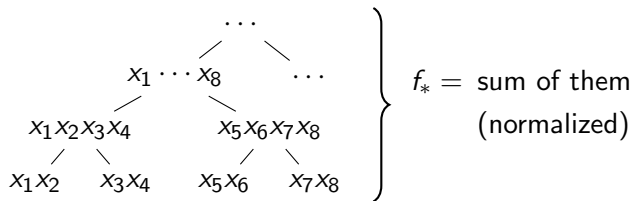
Application I: Hierarchical staircase function [R, Wang, Lee, 2026]



Theorem ([RWL'26], lower bound)

No constant-depth FC network (with sufficiently regular activations and spectrally bounded weights) can approximate f_ to $o(1)$ -accuracy.*

Application I: Hierarchical staircase function [R, Wang, Lee, 2026]



► **Fact.** Set $\chi_S(\mathbf{x}) := \prod_{i \in S} x_i$. $\{\chi_S\}_S$ form an **orthonormal** basis of $\mathcal{L}^2(\text{Unif}\{\pm 1\}^d)$.

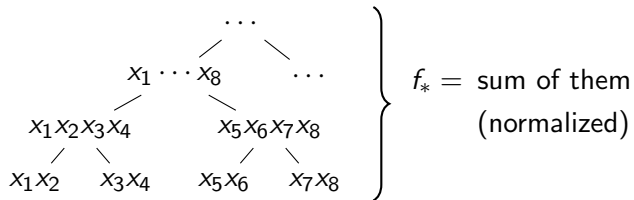
Shallow-to-deep chaining?

Correlation?

Clean signal?

Weak identifiability?

Application I: Hierarchical staircase function [R, Wang, Lee, 2026]



► **Fact.** Set $\chi_S(\mathbf{x}) := \prod_{i \in S} x_i$. $\{\chi_S\}_S$ form an **orthonormal** basis of $\mathcal{L}^2(\text{Unif}\{\pm 1\}^d)$.

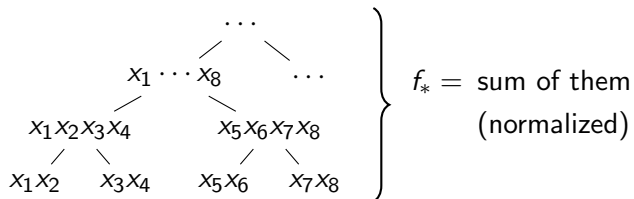
Shallow-to-deep chaining?

Correlation? $L^{1/2} \mathbb{E}[f_*(\mathbf{x})x_1x_2] = 2^{-(L-1)/2}$, $L^{1/2} \mathbb{E}[f_*(\mathbf{x})x_1 \cdots x_4] = 2^{-(L-2)/2}$, ...

Clean signal?

Weak identifiability?

Application I: Hierarchical staircase function [R, Wang, Lee, 2026]



► **Fact.** Set $\chi_S(\mathbf{x}) := \prod_{i \in S} x_i$. $\{\chi_S\}_S$ form an **orthonormal** basis of $\mathcal{L}^2(\text{Unif}\{\pm 1\}^d)$.

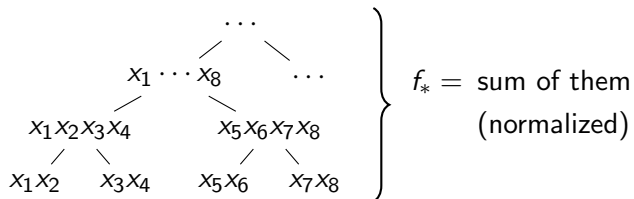
Shallow-to-deep chaining?

Correlation? $L^{1/2} \mathbb{E}[f_*(\mathbf{x})x_1x_2] = 2^{-(L-1)/2}$, $L^{1/2} \mathbb{E}[f_*(\mathbf{x})x_1 \cdots x_4] = 2^{-(L-2)/2}$, ...

Clean signal? Make first ℓ layers of the learner a degree- 2^ℓ polynomial
 $\implies$ can't overfit the higher part of the target.

Weak identifiability?

Application I: Hierarchical staircase function [R, Wang, Lee, 2026]



► **Fact.** Set $\chi_S(\mathbf{x}) := \prod_{i \in S} x_i$. $\{\chi_S\}_S$ form an **orthonormal** basis of $\mathcal{L}^2(\text{Unif}\{\pm 1\}^d)$.

Shallow-to-deep chaining?

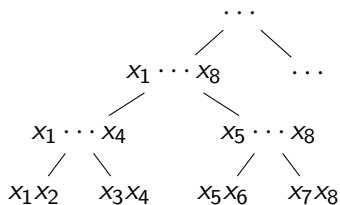
Correlation? $L^{1/2} \mathbb{E}[f_*(\mathbf{x})x_1x_2] = 2^{-(L-1)/2}$, $L^{1/2} \mathbb{E}[f_*(\mathbf{x})x_1 \cdots x_4] = 2^{-(L-2)/2}$, ...

Clean signal? Make first ℓ layers of the learner a degree- 2^ℓ polynomial
 $\implies$ can't overfit the higher part of the target.

Weak identifiability? Use explicit polynomial feature mappings.

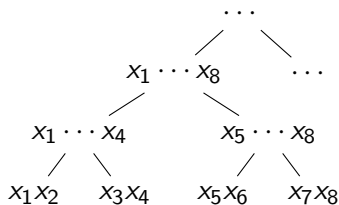
Learning deep quadratic functions — the meta algorithm

- **Meta layerwise algorithm.**



Learning deep quadratic functions — the meta algorithm

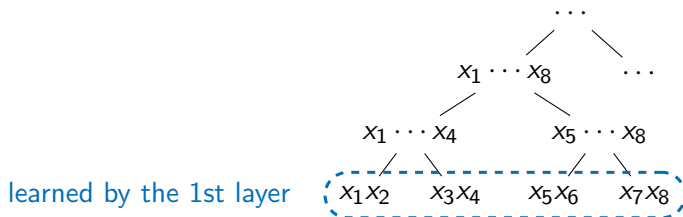
- **Meta layerwise algorithm.** $\mathcal{S}_* :=$ Fourier spectrum of f_*



Learning deep quadratic functions — the meta algorithm

► Meta layerwise algorithm.

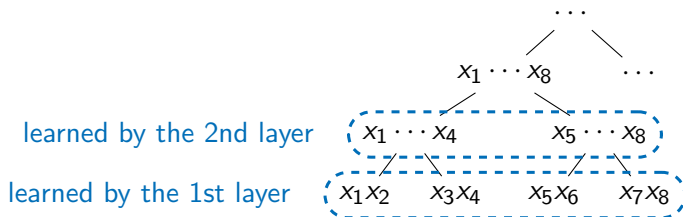
1. For each $i \neq j$, test if $\{i, j\} \in \mathcal{S}_*$ by estimating $\mathbb{E}[f_*(\mathbf{x})x_i x_j]$. Keep $\{i, j\}$ iff $\{i, j\} \in \mathcal{S}_*$.



Learning deep quadratic functions — the meta algorithm

► Meta layerwise algorithm.

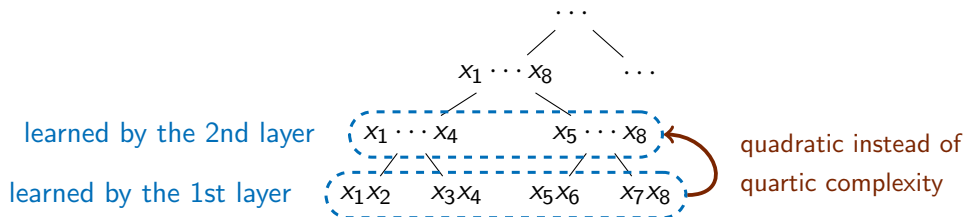
1. For each $i \neq j$, test if $\{i, j\} \in \mathcal{S}_*$ by estimating $\mathbb{E}[f_*(\mathbf{x})x_i x_j]$. **Keep $\{i, j\}$ iff $\{i, j\} \in \mathcal{S}_*$.**
2. For every **kept** $\{i_1, j_1\}, \{i_2, j_2\}$, test if $\{i_1, i_2, j_1, j_2\} \in \mathcal{S}_*$.
Keep $\{i_1, i_2, j_1, j_2\}$ iff $\{i_1, i_2, j_1, j_2\} \in \mathcal{S}_*$.
3. ...



Learning deep quadratic functions — the meta algorithm

► Meta layerwise algorithm.

1. For each $i \neq j$, test if $\{i, j\} \in \mathcal{S}_*$ by estimating $\mathbb{E}[f_*(\mathbf{x})x_i x_j]$. **Keep $\{i, j\}$ iff $\{i, j\} \in \mathcal{S}_*$.**
2. For every **kept** $\{i_1, j_1\}, \{i_2, j_2\}$, test if $\{i_1, i_2, j_1, j_2\} \in \mathcal{S}_*$.
Keep $\{i_1, i_2, j_1, j_2\}$ iff $\{i_1, i_2, j_1, j_2\} \in \mathcal{S}_*$.
3. ...



Learning hierarchical staircase functions

Informal Definition (Hierarchical staircase functions)

Boolean functions whose Fourier spectra $\mathcal{S}_*$ are hierarchical:

- ▶ High-degree terms in $\mathcal{S}_*$ can be constructed from a constant number of low-degree terms in $\mathcal{S}_*$, and/or $O(1)$ -degree terms.
- ▶ Overlapping and reusing are allowed.

$${}^2\mathcal{K}^{\oplus D} := \{K_1 \oplus \dots \oplus K_D : K_1, \dots, K_D \in \mathcal{K}\}.$$

Learning hierarchical staircase functions

Informal Definition (Hierarchical staircase functions)

Boolean functions whose Fourier spectra $\mathcal{S}_*$ are hierarchical:

- ▶ High-degree terms in $\mathcal{S}_*$ can be constructed from a constant number of low-degree terms in $\mathcal{S}_*$, and/or $O(1)$ -degree terms.
- ▶ Overlapping and reusing are allowed.

Definition (Hierarchical staircase functions)

f_* is a hierarchical staircase function of **depth** L and **local degree** D if $\mathcal{S}_* \subset \mathcal{K}_L$ where

$$\mathcal{K}_1 := \{\emptyset, \{1\}, \dots, \{d\}\}, \quad \mathcal{K}_{l+1} = \mathcal{K}_l \cup \left(\mathcal{K}_l^{\oplus D} \cap \mathcal{S} \right).^2 \quad (1)$$

² $\mathcal{K}^{\oplus D} := \{K_1 \oplus \dots \oplus K_D : K_1, \dots, K_D \in \mathcal{K}\}.$

Learning hierarchical staircase functions — main results

Theorem (Informal version of Theorem 3.1 of [RWL'26])

- ▶ Suppose local degree D is $O(1)$, $|\mathcal{S}_*| = \text{poly}(d)$, signal strength $= 1/\text{poly}(d)$.
- ▶ We can *train a depth- L network* using layerwise empirical “coordinate descent” to learn every hierarchical staircase function with depth $\leq L$, local degree $\leq D$.
- ▶ Model size, sample complexity, and running time are all $\text{poly}(d)$.

Learning hierarchical staircase functions — main results

Theorem (Informal version of Theorem 3.1 of [RWL'26])

- ▶ Suppose local degree D is $O(1)$, $|\mathcal{S}_*| = \text{poly}(d)$, signal strength $= 1/\text{poly}(d)$.
- ▶ We can *train a depth- L network* using layerwise empirical “coordinate descent” to learn every hierarchical staircase function with depth $\leq L$, local degree $\leq D$.
- ▶ Model size, sample complexity, and running time are all $\text{poly}(d)$.

Corollary (Algorithmic depth separation)

- ▶ *No* (regular) constant-depth networks can *approximate* deep quadratic functions;
- ▶ *Trained* $\log(d)$ -depth networks can *learn* deep quadratic functions.

Application II: Random Hierarchy Model (RHM)

- ▶ Previous example (hierarchical staircase functions):

Orthogonality of boolean monomials $\Rightarrow$ clean signals for the lower layers.

Application II: Random Hierarchy Model (RHM)

- ▶ Previous example (hierarchical staircase functions):

Orthogonality of boolean monomials $\Rightarrow$ clean signals for the lower layers.

- ▶ **Q:** Other mechanisms?

Application II: Random Hierarchy Model (RHM)

- ▶ Previous example (hierarchical staircase functions):

Orthogonality of boolean monomials $\Rightarrow$ clean signals for the lower layers.

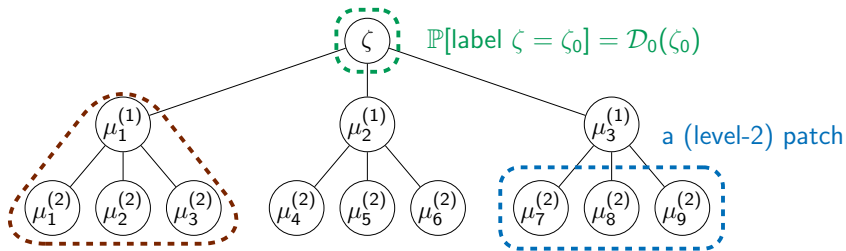
- ▶ **Q:** Other mechanisms? **A:** Tree/Markovian structure $\Rightarrow$ clean signal.

Application II: Random Hierarchy Model (RHM)

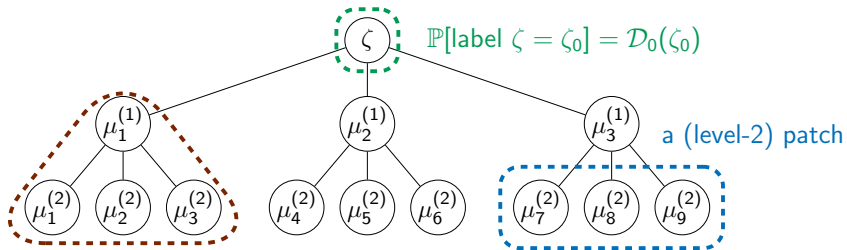
- Previous example (hierarchical staircase functions):

Orthogonality of boolean monomials $\Rightarrow$ clean signals for the lower layers.

- **Q:** Other mechanisms? **A:** Tree/Markovian structure $\Rightarrow$ clean signal.



$$\mathbb{P} \left[\left(\mu_1^{(l+1)}, \mu_2^{(l+1)}, \mu_3^{(l+1)} \right) = \boldsymbol{\mu} \mid \mu_1^{(l)} = \nu \right] = \mathcal{Q}_l(\nu, \boldsymbol{\mu}) \quad \leftarrow \text{translation invariant}$$

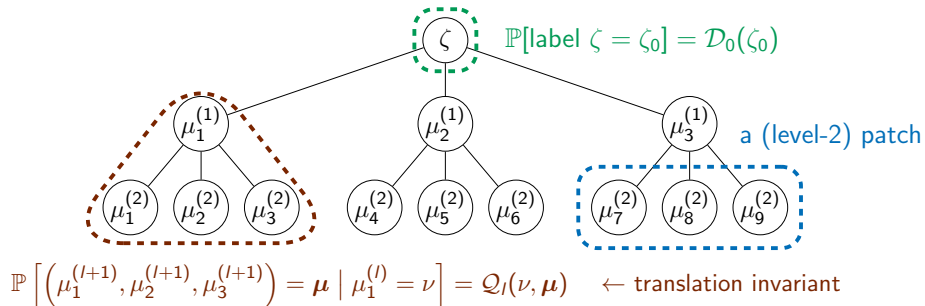


$$\mathbb{P} \left[\left(\mu_1^{(l+1)}, \mu_2^{(l+1)}, \mu_3^{(l+1)} \right) = \boldsymbol{\mu} \mid \mu_1^{(l)} = \nu \right] = \mathcal{Q}_l(\nu, \boldsymbol{\mu}) \quad \leftarrow \text{translation invariant}$$

► Assume **non-ambiguity**.

- Different level- $(l - 1)$ symbols cannot generate the same level- l patch.

Clean signal and weak identifiability



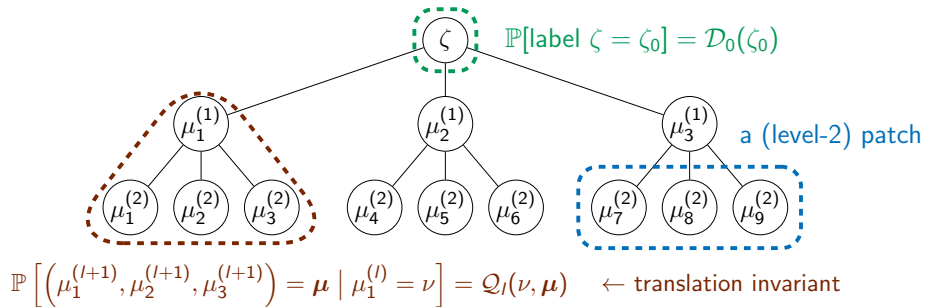
► Assume **non-ambiguity**.

- Different level- $(l - 1)$ symbols cannot generate the same level- l patch.

Clean signal and weak identifiability

- Train the model to predict the label using **only the first patch**

$$\Rightarrow \mathbf{q}_\mu^{(l)} := (\mathbb{P}[\text{label is } \zeta \mid \text{first level-}l \text{ patch is } \mu])_{\zeta \in \mathcal{V}_0}.$$



► Assume **non-ambiguity**.

- Different level- $(l-1)$ symbols cannot generate the same level- l patch.

Clean signal and weak identifiability

- Train the model to predict the label using **only the first patch**

$$\implies \mathbf{q}_{\mu}^{(l)} := (\mathbb{P}[\text{label is } \zeta \mid \text{first level-}l \text{ patch is } \mu])_{\zeta \in \mathcal{V}_0}.$$

- $\mathbf{q}_{\mu}^{(l)} = \mathbf{q}_{\mu'}^{(l)}$ iff μ, μ' generated by the same ν .

Summary and future directions

Shallow-to-deep chaining principle

Correlation + clean signal + weak identifiability $\Rightarrow$ hierarchical learning

- ▶ **Mechanisms:** orthogonality, tree structure, ...

Summary and future directions

Shallow-to-deep chaining principle

Correlation + clean signal + weak identifiability $\Rightarrow$ hierarchical learning

- **Mechanisms:** orthogonality, tree structure, ... **Q.** Other mechanisms?

Summary and future directions

Shallow-to-deep chaining principle

Correlation + clean signal + weak identifiability $\Rightarrow$ hierarchical learning

- ▶ **Mechanisms:** orthogonality, tree structure, ... **Q.** Other mechanisms?
- ▶ **Limitation.** The clean signal condition and identifiability are **rigid**.

Summary and future directions

Shallow-to-deep chaining principle

Correlation + clean signal + weak identifiability $\Rightarrow$ hierarchical learning

- ▶ **Mechanisms:** orthogonality, tree structure, ... **Q.** Other mechanisms?
- ▶ **Limitation.** The clean signal condition and identifiability are **rigid**.
 - ▶ **Ideal.** Mild overfitting should be allowed, as long as fixable after we fit later layers.
(backward feature correction [\[Allen-Zhu, Li \(2020\)\]](#)).

Summary and future directions

Shallow-to-deep chaining principle

Correlation + clean signal + weak identifiability $\Rightarrow$ hierarchical learning

- ▶ **Mechanisms:** orthogonality, tree structure, ... **Q.** Other mechanisms?
- ▶ **Limitation.** The clean signal condition and identifiability are **rigid**.
 - ▶ **Ideal.** Mild overfitting should be allowed, as long as fixable after we fit later layers. (backward feature correction [Allen-Zhu, Li (2020)]).
 - ▶ **Future direction.** Incorporate generic backward feature correction into this framework.

Summary and future directions

Shallow-to-deep chaining principle

Correlation + clean signal + weak identifiability $\Rightarrow$ hierarchical learning

- ▶ **Mechanisms:** orthogonality, tree structure, ... **Q.** Other mechanisms?
- ▶ **Limitation.** The clean signal condition and identifiability are **rigid**.
 - ▶ **Ideal.** Mild overfitting should be allowed, as long as fixable after we fit later layers. (backward feature correction [Allen-Zhu, Li (2020)]).
 - ▶ **Future direction.** Incorporate generic backward feature correction into this framework.

Takeaway message

Summary and future directions

Shallow-to-deep chaining principle

Correlation + clean signal + weak identifiability $\Rightarrow$ hierarchical learning

- ▶ **Mechanisms:** orthogonality, tree structure, ... **Q.** Other mechanisms?
- ▶ **Limitation.** The clean signal condition and identifiability are **rigid**.
 - ▶ **Ideal.** Mild overfitting should be allowed, as long as fixable after we fit later layers. (backward feature correction [Allen-Zhu, Li (2020)]).
 - ▶ **Future direction.** Incorporate generic backward feature correction into this framework.

Takeaway message

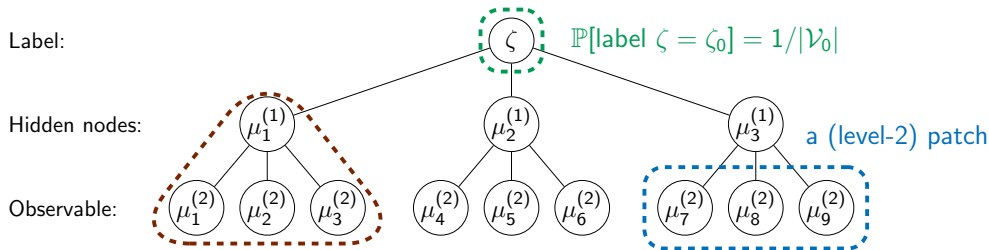
Optimization guarantees beyond shallow models are not as hard as you might think.

Appendix. Learning Random Hierarchy Models

Definition (Random Hierarchy Model [Cagnetta et al., 2024])

An **unambiguous L -level RHM** with **branching factor s** is a PCFG with $L + 1$ levels:

- **Production rules:** Specifying what level- l patches μ can a level- $(l - 1)$ symbol ν generate;
 - Example: $\nu_1 \rightarrow \mu_1\mu_1\mu_1 \mid \mu_1\mu_2\mu_3, \quad \nu_2 \rightarrow \mu_3\mu_2\mu_1 \mid \mu_2\mu_2\mu_2.$
- **Non-ambiguity:** Different symbols can not generate the same patch.

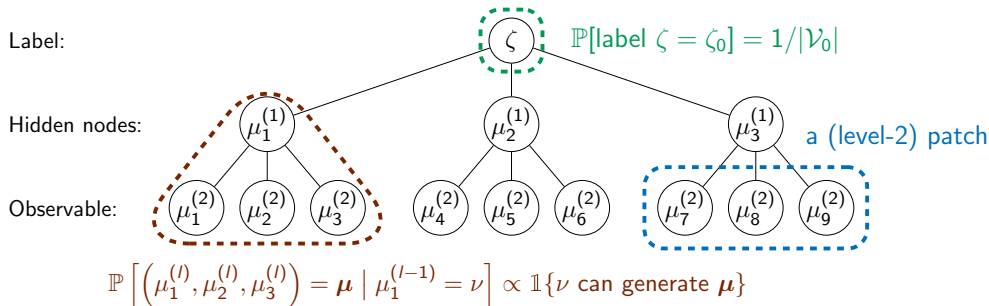


$$\mathbb{P} \left[\left(\mu_1^{(l)}, \mu_2^{(l)}, \mu_3^{(l)} \right) = \mu \mid \mu_1^{(l-1)} = \nu \right] \propto \mathbb{1} \{ \nu \text{ can generate } \mu \}$$

Definition (Random Hierarchy Model [Cagnetta et al., 2024])

An **unambiguous L -level** RHM with **branching factor s** is a PCFG with $L + 1$ levels:

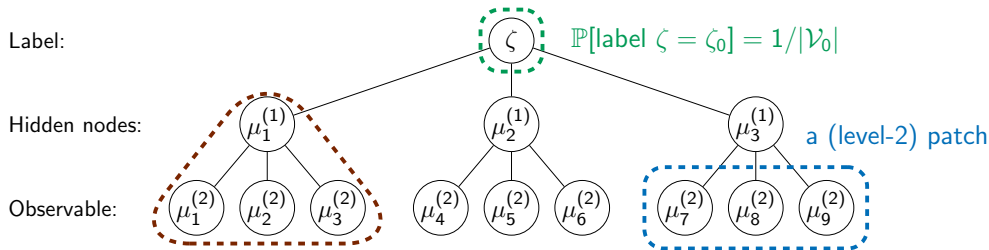
- **Production rules:** Specifying what level- l patches μ can a level- $(l - 1)$ symbol ν generate;
 - Example: $\nu_1 \rightarrow \mu_1\mu_1\mu_1 \mid \mu_1\mu_2\mu_3, \quad \nu_2 \rightarrow \mu_3\mu_2\mu_1 \mid \mu_2\mu_2\mu_2.$
- **Non-ambiguity:** Different symbols can not generate the same patch.



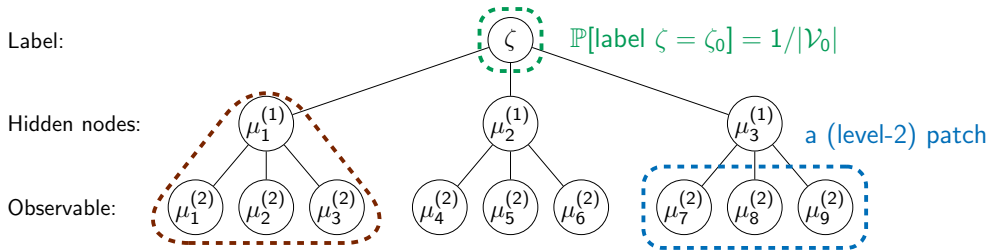
Definition (Random Hierarchy Model [Cagnetta et al., 2024])

An **unambiguous L -level** RHM with **branching factor s** is a PCFG with $L + 1$ levels:

- **Production rules:** Specifying what level- l patches μ can a level- $(l - 1)$ symbol ν generate;
 - Example: $\nu_1 \rightarrow \mu_1\mu_1\mu_1 \mid \mu_1\mu_2\mu_3, \quad \nu_2 \rightarrow \mu_3\mu_2\mu_1 \mid \mu_2\mu_2\mu_2.$
- **Non-ambiguity:** Different symbols can not generate the same patch.



$$\mathbb{P} \left[\left(\mu_1^{(l)}, \mu_2^{(l)}, \mu_3^{(l)} \right) = \mu \mid \mu_1^{(l-1)} = \nu \right] \propto \mathbb{1} \{ \nu \text{ can generate } \mu \}$$



$$\mathbb{P} \left[\left(\mu_1^{(l)}, \mu_2^{(l)}, \mu_3^{(l)} \right) = \boldsymbol{\mu} \mid \mu_1^{(l-1)} = \nu \right] \propto \mathbb{1} \{ \nu \text{ can generate } \boldsymbol{\mu} \}$$

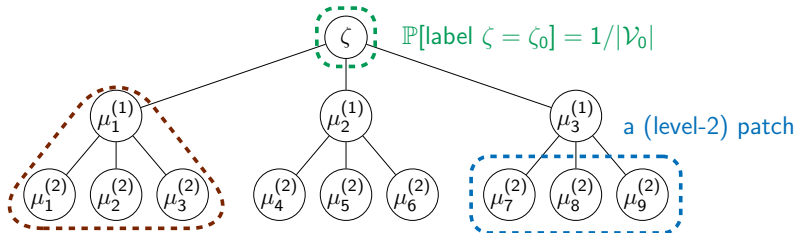
Learning task

- ▶ Learner can only see the leaves and root label (No intermediate supervision)
- ▶ **Goal.** Learn the function that maps the leaves to the label.
 - ▶ **Subgoal.** Infer the internal production rules (up to permutations).

Label:

Hidden nodes:

Observable:



$$\mathbb{P} \left[\left(\mu_1^{(l)}, \mu_2^{(l)}, \mu_3^{(l)} \right) = \boldsymbol{\mu} \mid \mu_1^{(l-1)} = \nu \right] \propto \mathbb{1} \{ \nu \text{ can generate } \boldsymbol{\mu} \}$$

Observations

- ▶ Train the model to predict the label using **only the first patch**

$$\Rightarrow \mathbf{q}_{\boldsymbol{\mu}}^{(l)} := (\mathbb{P} [\text{label is } \zeta \mid \text{first level-}l \text{ patch is } \boldsymbol{\mu}])_{\zeta \in \mathcal{V}_0}.$$

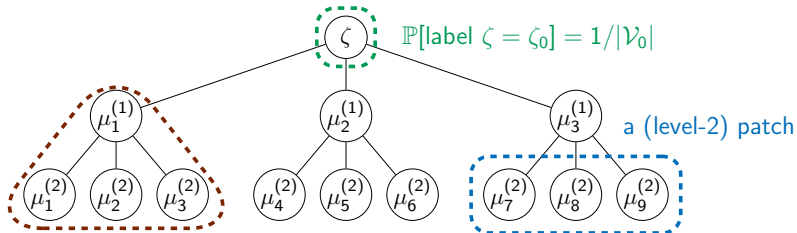
- ▶ Under some regularity conditions on the RHM instance,

$$\mathbf{q}_{\boldsymbol{\mu}}^{(l)} = \mathbf{q}_{\boldsymbol{\mu}'}^{(l)} \iff \boldsymbol{\mu}, \boldsymbol{\mu}' \text{ generated by the same } \nu.$$

Label:

Hidden nodes:

Observable:



$$\mathbb{P} \left[\left(\mu_1^{(l)}, \mu_2^{(l)}, \mu_3^{(l)} \right) = \boldsymbol{\mu} \mid \mu_1^{(l-1)} = \nu \right] \propto \mathbb{1} \{ \nu \text{ can generate } \boldsymbol{\mu} \}$$

Observations

- Train the model to predict the label using **only the first patch**

$$\Rightarrow \mathbf{q}_{\boldsymbol{\mu}}^{(l)} := (\mathbb{P} [\text{label is } \zeta \mid \text{first level-}l \text{ patch is } \boldsymbol{\mu}])_{\zeta \in \mathcal{V}_0}.$$

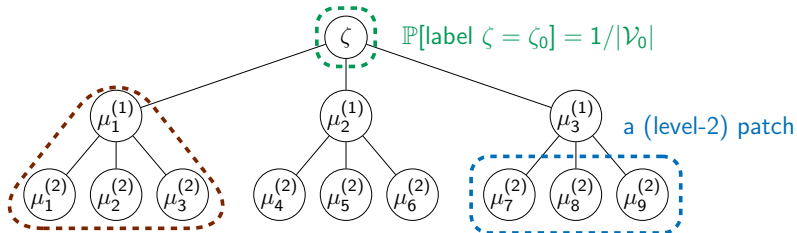
- Under some regularity conditions on the RBM instance,

$$\mathbf{q}_{\boldsymbol{\mu}}^{(l)} = \mathbf{q}_{\boldsymbol{\mu}'}^{(l)} \iff \boldsymbol{\mu}, \boldsymbol{\mu}' \text{ generated by the same } \nu.$$

Label:

Hidden nodes:

Observable:



$$\mathbb{P} \left[\left(\mu_1^{(l)}, \mu_2^{(l)}, \mu_3^{(l)} \right) = \boldsymbol{\mu} \mid \mu_1^{(l-1)} = \nu \right] \propto \mathbb{1} \{ \nu \text{ can generate } \boldsymbol{\mu} \}$$

Observations

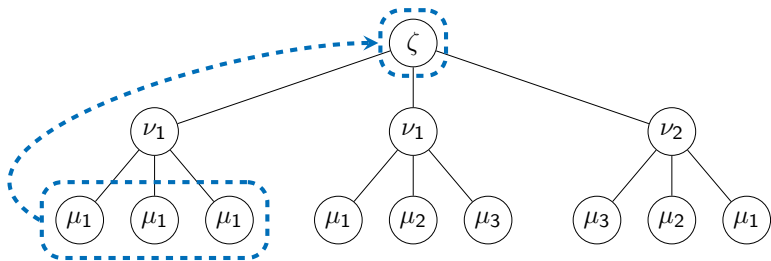
- Train the model to predict the label using **only the first patch**

$$\Rightarrow \mathbf{q}_{\boldsymbol{\mu}}^{(l)} := (\mathbb{P} [\text{label is } \zeta \mid \text{first level-}l \text{ patch is } \boldsymbol{\mu}])_{\zeta \in \mathcal{V}_0}.$$

- Under some regularity conditions on the RHM instance,

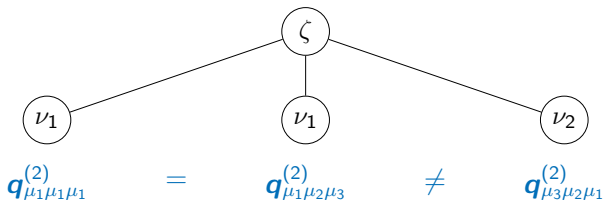
$$\mathbf{q}_{\boldsymbol{\mu}}^{(l)} = \mathbf{q}_{\boldsymbol{\mu}'}^{(l)} \iff \boldsymbol{\mu}, \boldsymbol{\mu}' \text{ generated by the same } \nu.$$

► Production rules: $\nu_1 \rightarrow \mu_1\mu_1\mu_1 \mid \mu_1\mu_2\mu_3$, $\nu_2 \rightarrow \mu_3\mu_2\mu_1 \mid \dots$, $\dots$



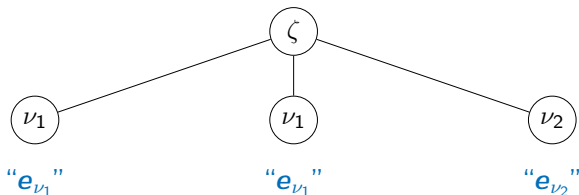
1. Train the model to predict the label using the first level- l patch
 $\implies$ approximately learn $\mu \mapsto \mathbf{q}_\mu^{(l)}$ (using the l -th level of the learner network).
2. Translation invariance $\implies$ can apply the same $\mu \mapsto \mathbf{q}_\mu^{(l)}$ to all patches.
3. Use the level- l outputs as the embedding of the level- $(l - 1)$ tokens.
4. Use an RBF RF layer to convert $\{\mathbf{q}_\mu^{(l)}\}_\mu$ to (near-)orthonormal embeddings of $\{\nu\}_\nu$
5. Repeat for level $l + 1$.

► Production rules: $\nu_1 \rightarrow \mu_1\mu_1\mu_1 \mid \mu_1\mu_2\mu_3, \quad \nu_2 \rightarrow \mu_3\mu_2\mu_1 \mid \dots, \quad \dots$



1. Train the model to predict the label using the first level- l patch
 $\implies$ approximately learn $\mu \mapsto \mathbf{q}_\mu^{(l)}$ (using the l -th level of the learner network).
2. Translation invariance $\implies$ can apply the same $\mu \mapsto \mathbf{q}_\mu^{(l)}$ to all patches.
3. Use the level- l outputs as the embedding of the level- $(l - 1)$ tokens.
4. Use an RBF RF layer to convert $\{\mathbf{q}_\mu^{(l)}\}_\mu$ to (near-)orthonormal embeddings of $\{\nu\}_\nu$
5. Repeat for level $l + 1$.

► Production rules: $\nu_1 \rightarrow \mu_1\mu_1\mu_1 \mid \mu_1\mu_2\mu_3, \quad \nu_2 \rightarrow \mu_3\mu_2\mu_1 \mid \dots, \quad \dots$



1. Train the model to predict the label using the first level- l patch
 $\implies$ approximately learn $\mu \mapsto \mathbf{q}_\mu^{(l)}$ (using the l -th level of the learner network).
2. Translation invariance $\implies$ can apply the same $\mu \mapsto \mathbf{q}_\mu^{(l)}$ to all patches.
3. Use the level- l outputs as the embedding of the level- $(l - 1)$ tokens.
4. Use an RBF RF layer to convert $\{\mathbf{q}_\mu^{(l)}\}_\mu$ to (near-)orthonormal embeddings of $\{\nu\}_\nu$
5. Repeat for level $l + 1$.

► Production rules: $\nu_1 \rightarrow \mu_1\mu_1\mu_1 \mid \mu_1\mu_2\mu_3, \quad \nu_2 \rightarrow \mu_3\mu_2\mu_1 \mid \dots, \quad \dots$

1. Train the model to predict the label using the first level- l patch
 $\implies$ approximately learn $\mu \mapsto \mathbf{q}_\mu^{(l)}$ (using the l -th level of the learner network).
2. Translation invariance $\Rightarrow$ can apply the same $\mu \mapsto \mathbf{q}_\mu^{(l)}$ to all patches.
3. Use the level- l outputs as the embedding of the level- $(l - 1)$ tokens.
4. Use an RBF RF layer to convert $\{\mathbf{q}_\mu^{(l)}\}_\mu$ to (near-)orthonormal embeddings of $\{\nu\}_\nu$
5. Repeat for level $l + 1$.

Learning Random Hierarchy Models — main results

Theorem (Informal version of Theorem 4.1 of [RDKL'26])

- ▶ For any non-degenerate, L -level, unambiguous RHM with branching factor s and m production rules per symbol,
- ▶ a polynomial-width L -layer convolutional network,
- ▶ trained with *layerwise empirical gradient descent*,
- ▶ can efficiently learn the input-to-label function using $O(m^{(1+o(1))L})$ *samples* and *sample-polynomial gradient steps*.

Learning Random Hierarchy Models — main results

Theorem (Informal version of Theorem 4.1 of [RDKL'26])

- ▶ For any non-degenerate, L -level, unambiguous RHM with branching factor s and m production rules per symbol,
- ▶ a polynomial-width L -layer convolutional network,
- ▶ trained with layerwise empirical gradient descent,
- ▶ can efficiently learn the input-to-label function using $O(m^{(1+o(1))L})$ samples and sample-polynomial gradient steps.

Conjecture/Empirical observation [Cagnetta et al., 2024]. Shallow networks require m^{s^L} samples.

(Take $s = m = \log \log d$ and $L = \log d / \log \log d$ to get an exponential separation.)