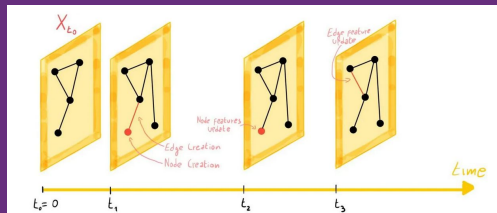
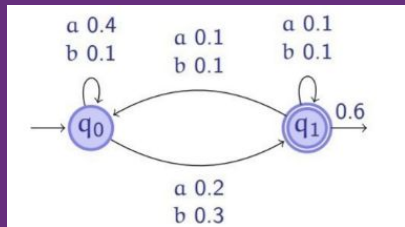


Second-order RNNs, Tensor Decomposition, Weighted Automata & Transformers



Guillaume Rabusseau, DIRO / Mila
RIKEN - AIP

October 2024

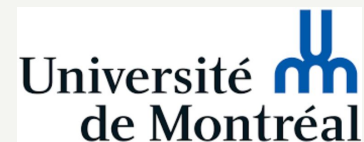
About me



PhD
2013-2016



Postdoc
2016-2018



Professor
2018-

Research Group



Farzaneh Heidari (PhD)
Tensor Networks for Interpretability



Beheshteh Rakhshan (PhD)
Randomized Methods and Tensor Networks



Marawan Gamal (PhD)
Tensor Networks for Efficient Language Models



Andy Huang (PhD)
Learning with Dynamic Graphs



Alireza Dizaji (MSc)
Temporal Structures in Dynamic Graphs



Jun Dai (postdoc)
Quantum Computing and Machine Learning



Michael Rizvi-Martel (PhD)
Formal Analysis of Reasoning in Sequence Models



Maude Lizaire (PhD)
Formal Languages and Recurrent Neural Networks



Pascal Jr. Tikeng Notsawo (PhD)
Generalization of Neural Networks Beyond the Overfitting Regime



Soroush Omranpour (MSc)
Multi-Modal Attention with Tensor Networks

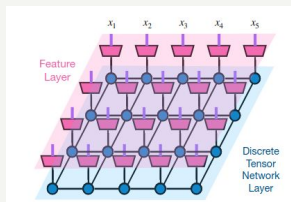
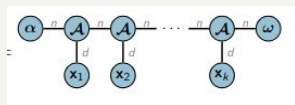


Omar Chikhar (MSc)
Kernel Quantum Machine Learning

Research Interests

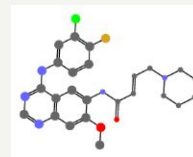
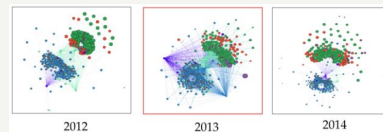
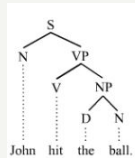
Tensor Networks for ML

- TNs for efficient ML models (PEFT, inference, compression, ...)
- TNs for interpretable ML models
- TNs for very high-dimensional data
- TNs for multi-modal data



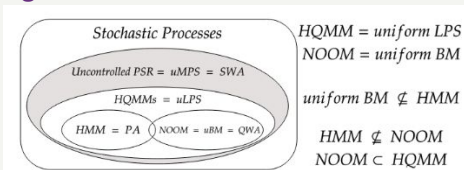
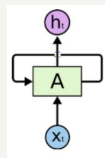
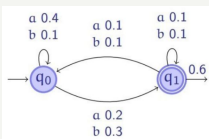
Learning with Structured Data

- Temporal Graphs Models & Benchmarks
- Learning and designing models for sequences, trees, graphs (with TNs but not only)
- Spectral Learning of Weighted Automata



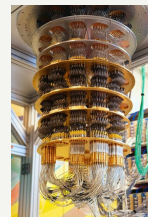
Formal Methods and ML

- Formal methods to analyze / probe reasoning abilities of ML models
- Formal analysis of ML models (benefits of depth, separation results)
- Understanding links between NN and formal models



Quantum Computing & ML

- Inductive biases in QML.
- Which quantum circuits can't be simulated by Tensor Networks? How are they relevant to ML?
- QML theory: Generalization bounds, benefits of depth...
- Learning Quantum Circuit Designs.

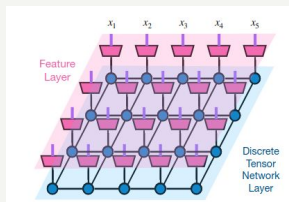
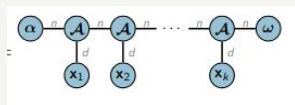


$$\mathbb{P}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4) = \left(\alpha \overset{R}{\text{---}} \underset{\underset{\mathbf{x}_1}{\uparrow}^d}{\mathcal{A}} \overset{R}{\text{---}} \underset{\underset{\mathbf{x}_2}{\uparrow}^d}{\mathcal{A}} \overset{R}{\text{---}} \underset{\underset{\mathbf{x}_3}{\uparrow}^d}{\mathcal{A}} \overset{R}{\text{---}} \underset{\underset{\mathbf{x}_4}{\uparrow}^d}{\mathcal{A}} \overset{R}{\text{---}} \omega \right)^2$$

Research Interests

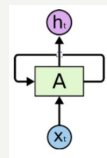
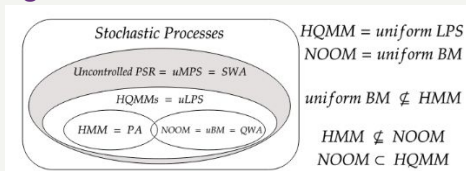
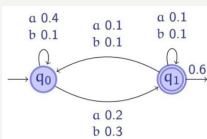
Tensor Networks for ML

- TNs for efficient ML models (PEFT, inference, compression, ...)
- TNs for interpretable ML models
- TNs for very high-dimensional data
- TNs for multi-modal data



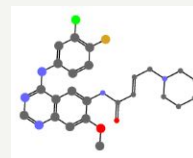
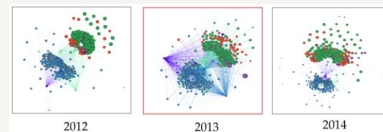
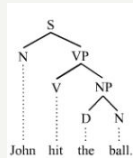
Formal Methods and ML

- Formal methods to analyze / probe reasoning abilities of ML models
- Formal analysis of ML models (benefits of depth, separation results)
- Understanding links between NN and formal models



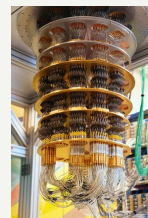
Learning with Structured Data

- Temporal Graphs Models & Benchmarks
- Learning and designing models for sequences, trees, graphs (with TNs but not only)
- Spectral Learning of Weighted Automata



Quantum Computing & ML

- Inductive biases in QML.
- Which quantum circuits can't be simulated by Tensor Networks? How are they relevant to ML?
- QML theory: Generalization bounds, benefits of depth...
- Learning Quantum Circuit Designs.

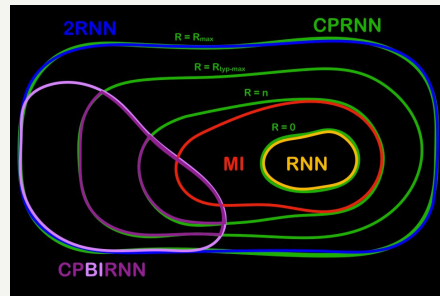


$$\mathbb{P}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4) = \left(\begin{array}{c} \alpha \quad A \quad A \quad A \quad A \quad \omega \\ \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\ x_1 \quad x_2 \quad x_3 \quad x_4 \end{array} \right)^2$$

Today's talk

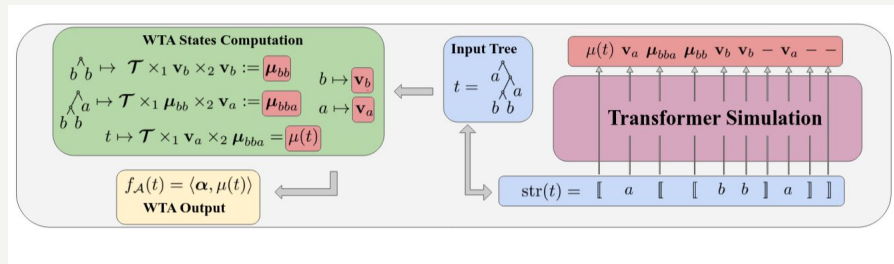
A Tensor Decomposition Perspective on 2nd Order RNNs, ICML 2024

M. Lizaire, M. Rizvi-Martel, M. Gamal & GR



Simulating Weighted Automata with Transformers, AISTATS 2024

M. Rizvi-Martel, M. Lizaire, C. Lacroce & GR



A Tensor Decomposition Perspective on Second-order RNNs

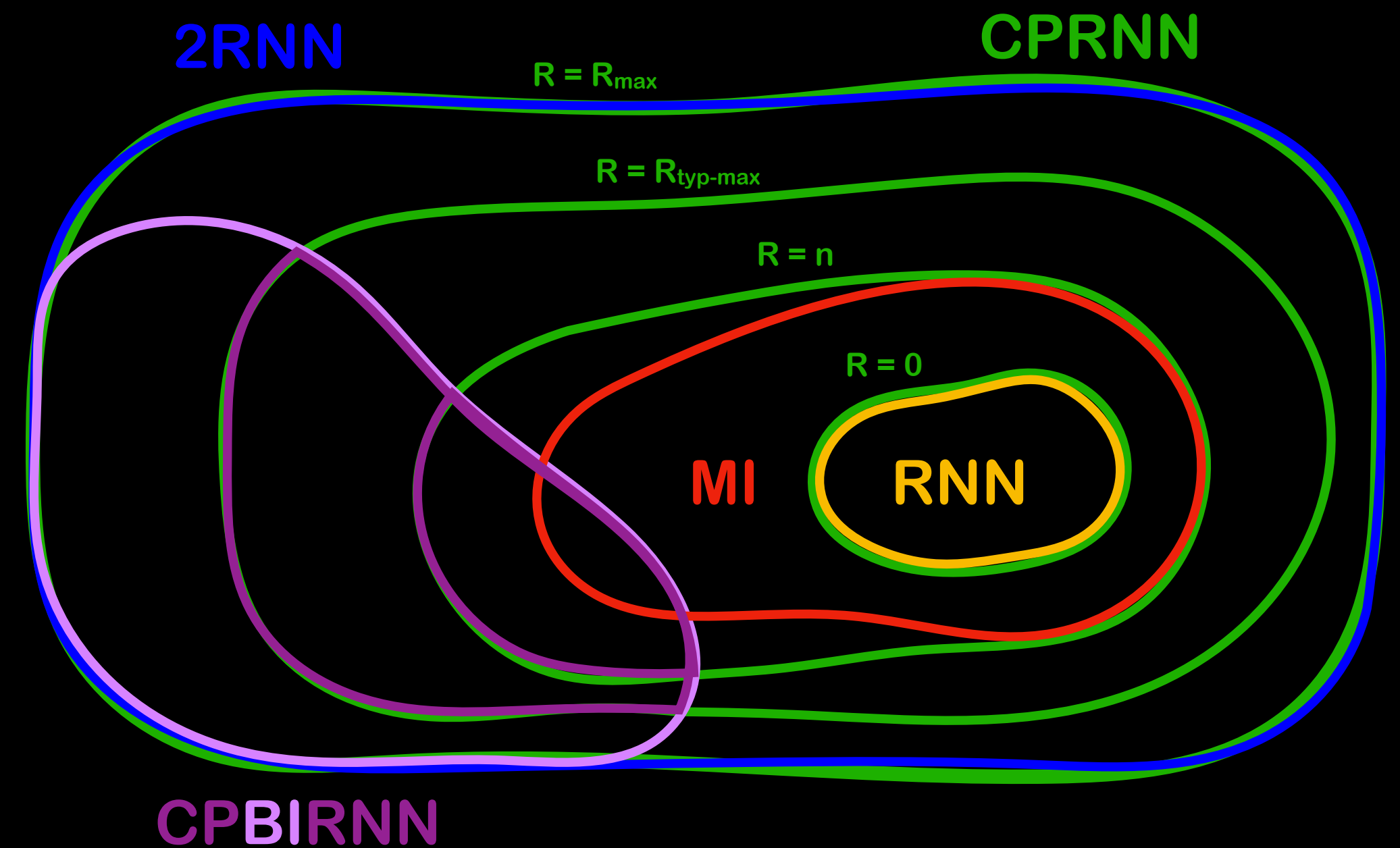
~CPRNNs~

Maude Lizaire, Michael Rizvi-Martel, Marawan Gamal, Guillaume Rabusseau

RIKEN-AIP - October 2024

Tensor Decomposition

as a tool to
characterize the
expressivity of RNNs
with **second-order**
interactions.



Outline

Introduction

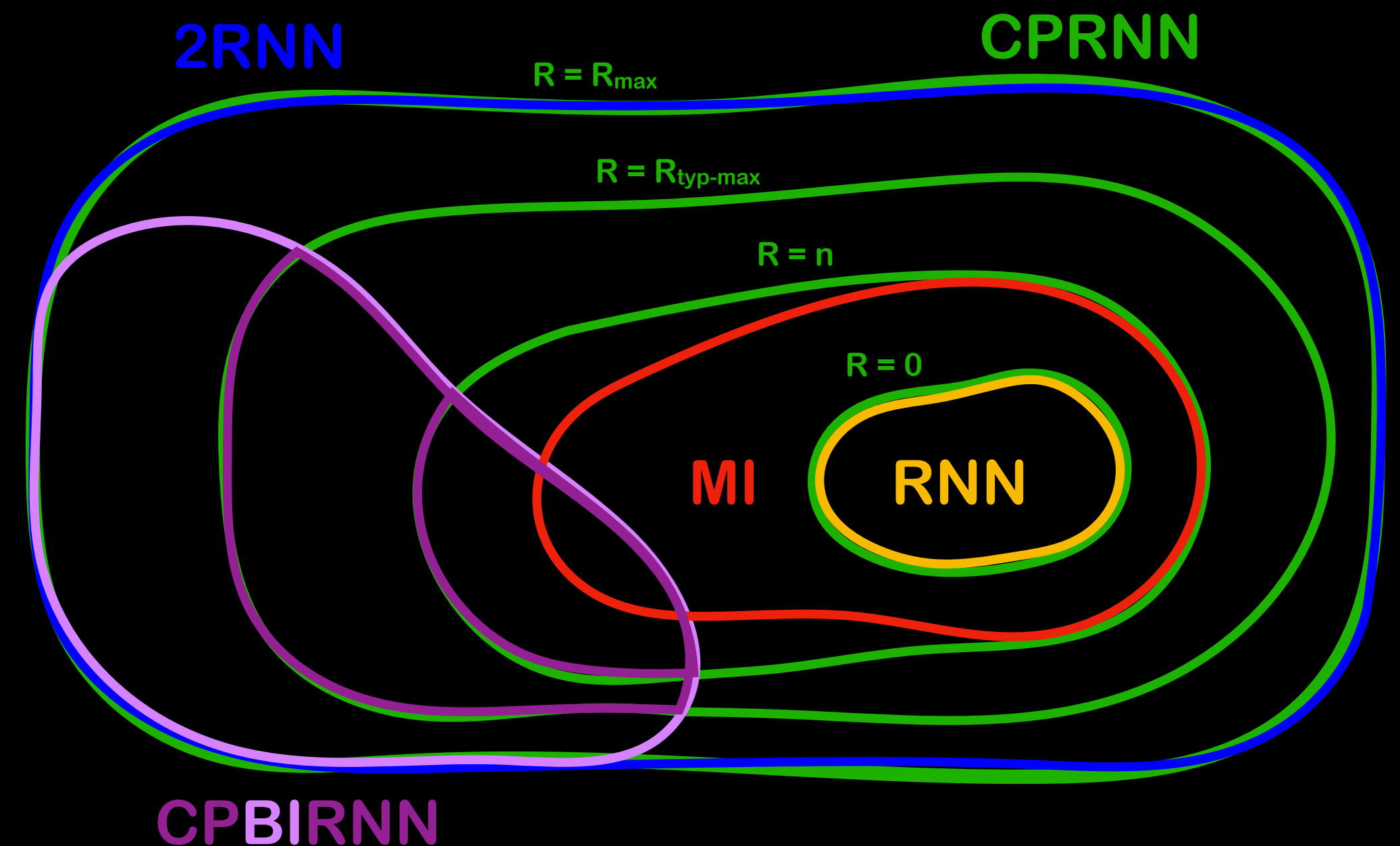
- What are 2RNNs? Why 2RNNs?
- CP Decomposition & CPRNNs
- Models
- Questions

Theoretical Results

- Thm1: Function Space and Tensor Space
- Thm 2: Interplay Rank and Hidden Size

Experimental Results

Conclusion

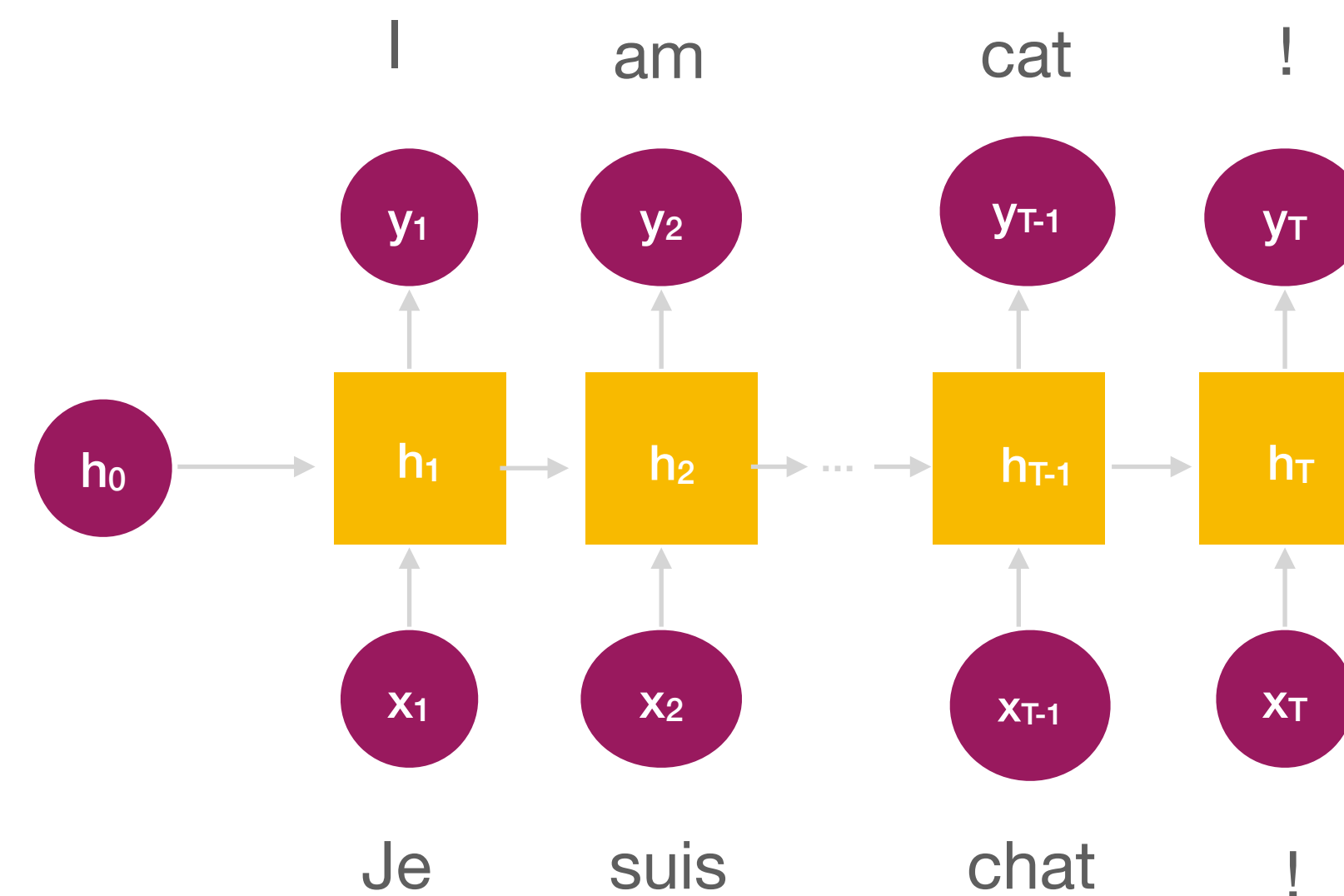


What are 2RNNs?

What are 2RNNs?

RNN

$$\sigma(\mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$$



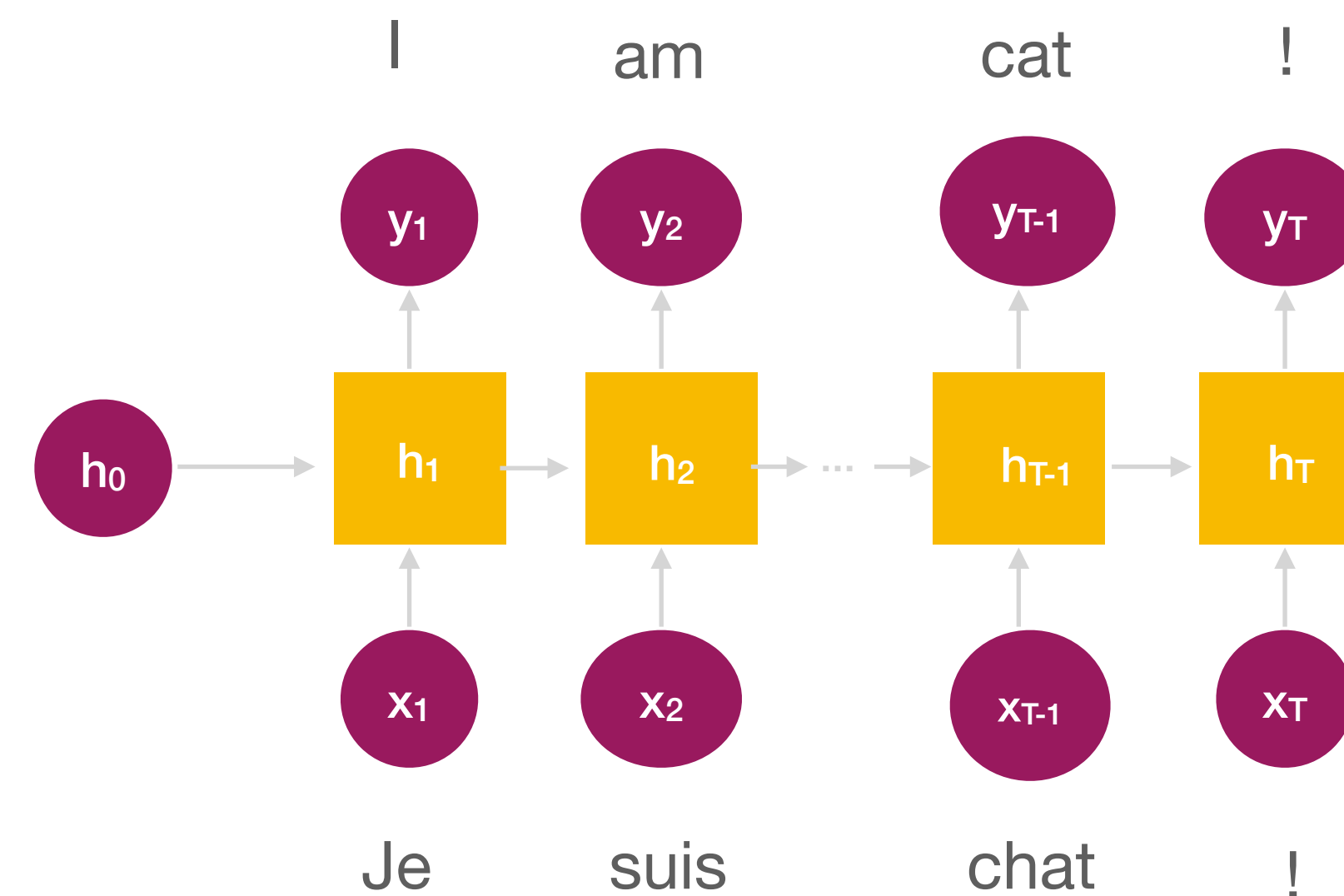
What are 2RNNs?

RNN

$$\sigma(\mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$$

2RNN

$$\sigma(\underbrace{\mathcal{A} \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t}_{\text{tensor product}} + \underbrace{\mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b}}_{\text{linear combination}})$$



What are 2RNNs?

RNN

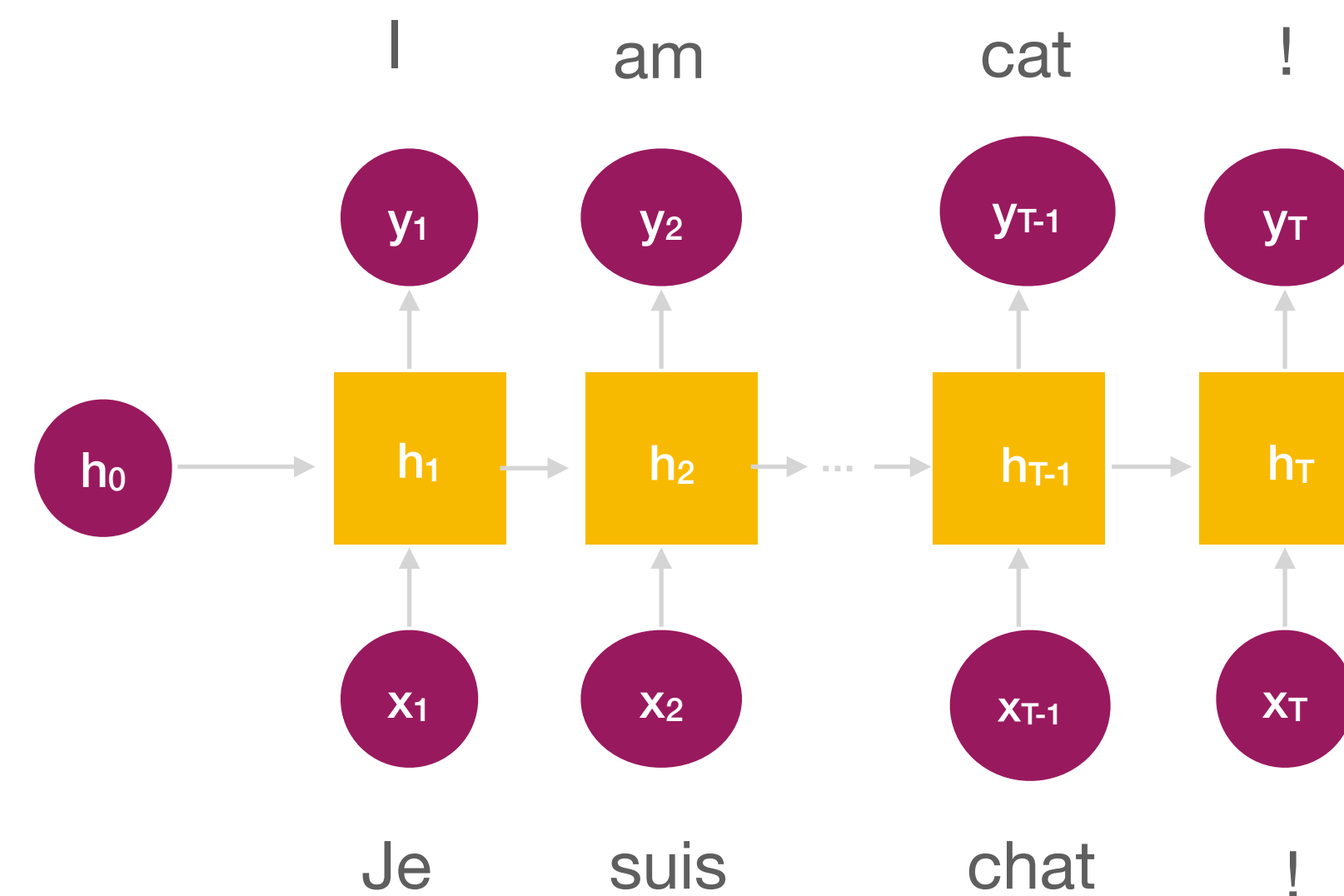
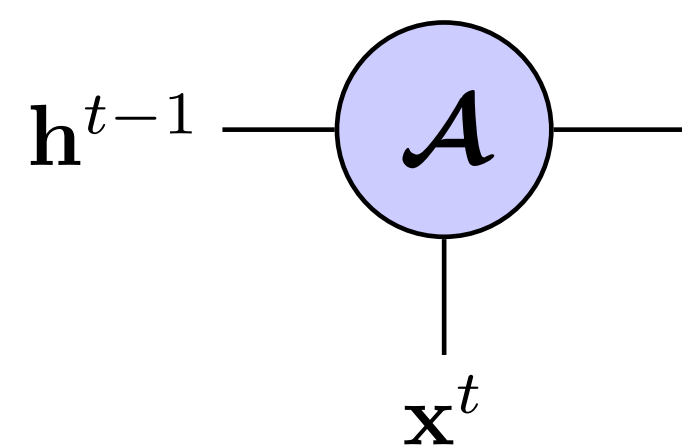
$$\sigma(\mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$$

2RNN

$$\sigma(\underbrace{\mathcal{A} \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t}_{\text{2nd order term}} + \underbrace{\mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b}}_{\text{RNN}})$$

2nd order term

RNN



Why 2RNNs?

RNN

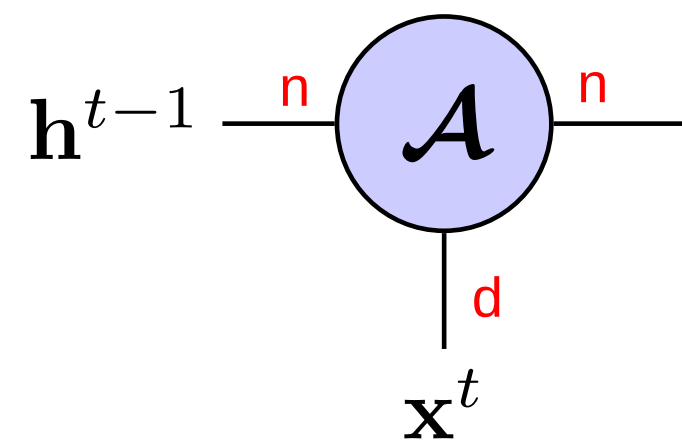
$$\sigma(\mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$$

2RNN

$$\sigma(\underbrace{\mathcal{A} \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t}_{\text{2nd order term}} + \underbrace{\mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b}}_{\text{RNN}})$$

2nd order term

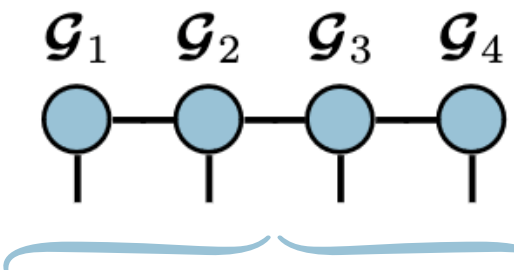
RNN



But, 2nd order parameter tensor very large!

Because,

- Strictly more expressive than RNNs
- Language Modelling : Learning more complex dependencies (e.g. compositional semantics).



- Connection with Tensor Train and Weighted Finite Automata (WFA)

$$f(x_1 x_2 \cdots x_k) = \alpha \quad \mathbf{A}^{x_1} \quad \mathbf{A}^{x_2} \quad \cdots \quad \mathbf{A}^{x_{k-1}} \quad \mathbf{A}^{x_k} \quad \omega$$

Why 2RNNs?

RNN

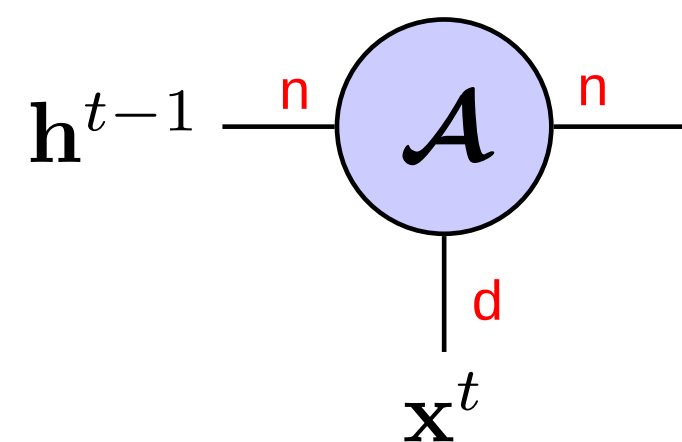
$$\sigma(\mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$$

2RNN

$$\sigma(\underbrace{\mathcal{A} \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t}_{\text{2nd order term}} + \underbrace{\mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b}}_{\text{RNN}})$$

2nd order term

RNN



Multiplicative Integration RNN

Only component-wise
(via Hadamard product)

MIRNN

2nd order term

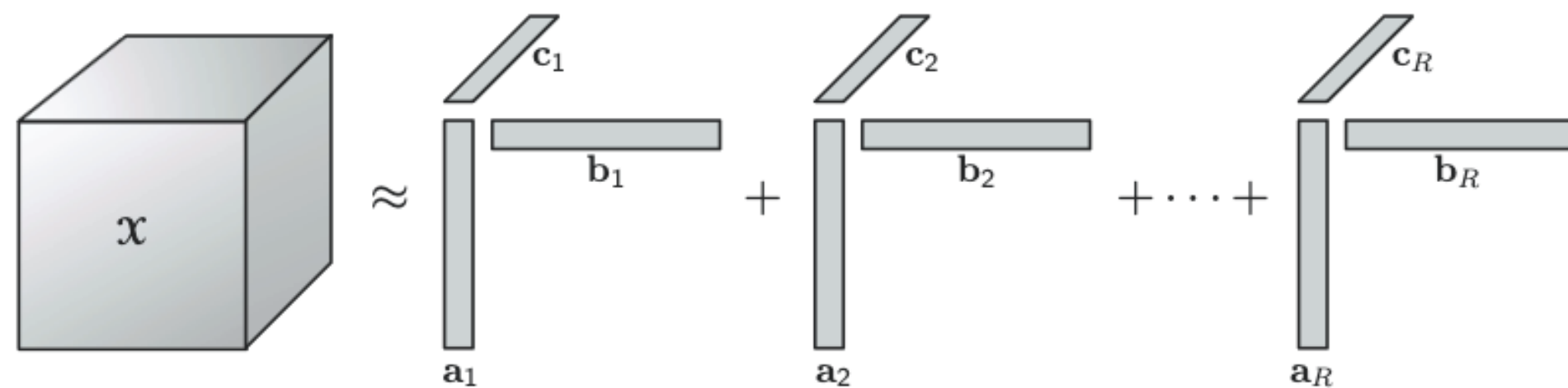
$$\sigma(\underbrace{\alpha \odot \mathbf{V}\mathbf{h}^{t-1} \odot \mathbf{U}\mathbf{x}^t}_{\text{2nd order term}} + \underbrace{\beta_1 \odot \mathbf{V}\mathbf{h}^{t-1} + \beta_2 \odot \mathbf{U}\mathbf{x}^t + \mathbf{b}}_{\text{RNN}})$$

RNN

But, 2nd order parameter tensor very large!

CP Decomposition & CPRNN

$$\mathcal{X} \approx [\mathbf{A}, \mathbf{B}, \mathbf{C}] \equiv \sum_{r=1}^R \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$$

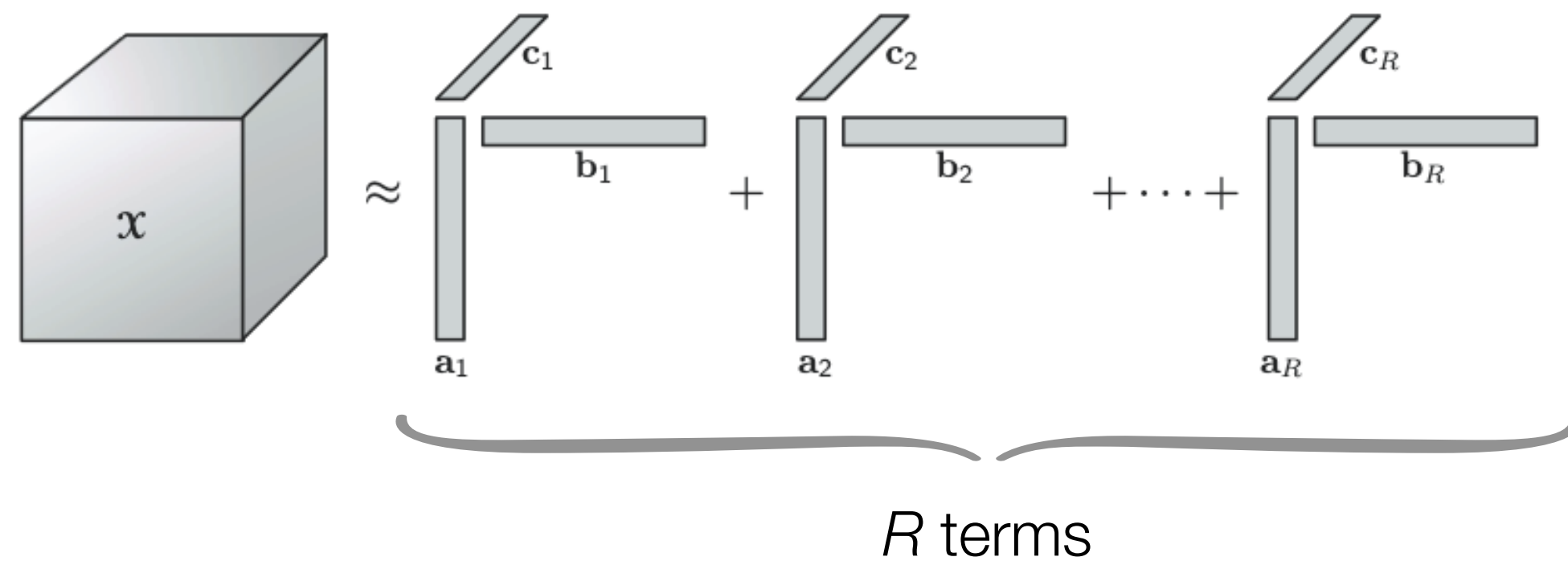


T. Kolda and B. Bader, Tensor Decompositions and applications. SIAM REVIEW (2009)

CP Decomposition & CPRNN

$$\mathcal{X} \approx [\mathbf{A}, \mathbf{B}, \mathbf{C}] \equiv \sum_{r=1}^R \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$$

Rank



T. Kolda and B. Bader, Tensor Decompositions and applications. SIAM REVIEW (2009)

Reduces the number of parameters:

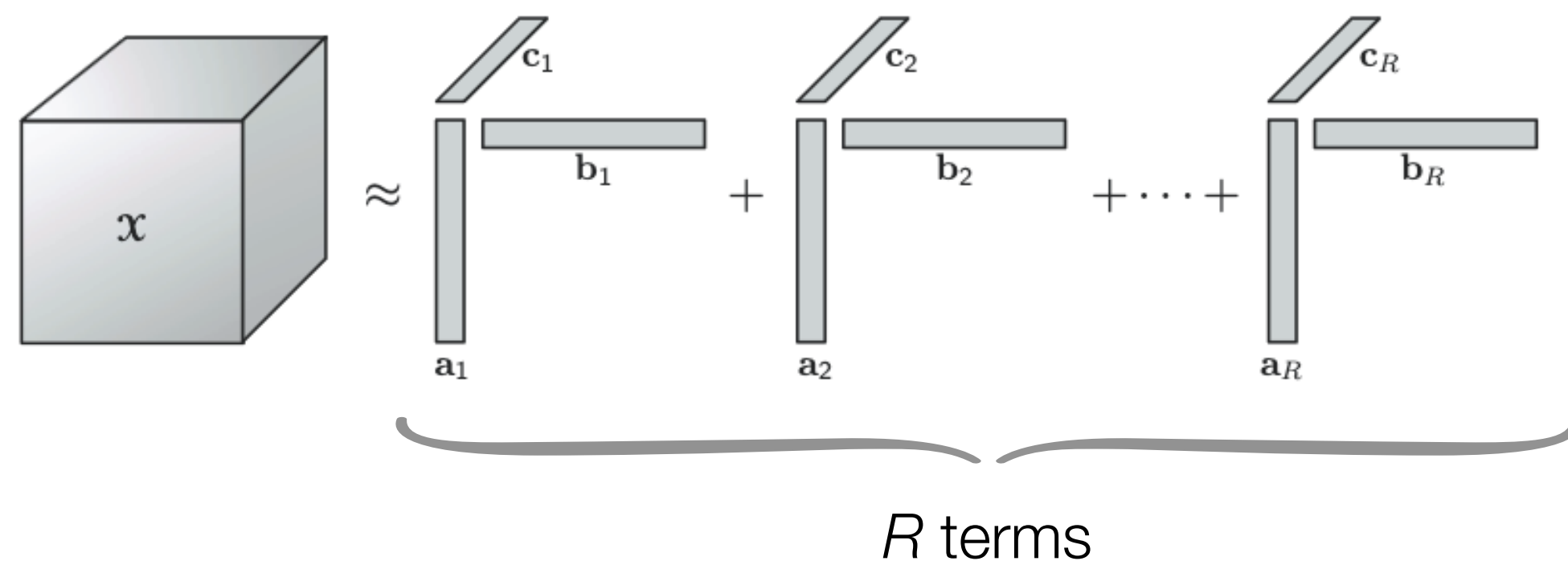
$$\mathcal{O}(d^3) \longrightarrow \mathcal{O}(Rd)$$

$d = \max\{d_1, d_2, d_3\}$

CP Decomposition & CPRNN

$$\mathcal{X} \approx [\mathbf{A}, \mathbf{B}, \mathbf{C}] \equiv \sum_{r=1}^R \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$$

Rank



T. Kolda and B. Bader, Tensor Decompositions and applications. SIAM REVIEW (2009)

Reduces the number of parameters:

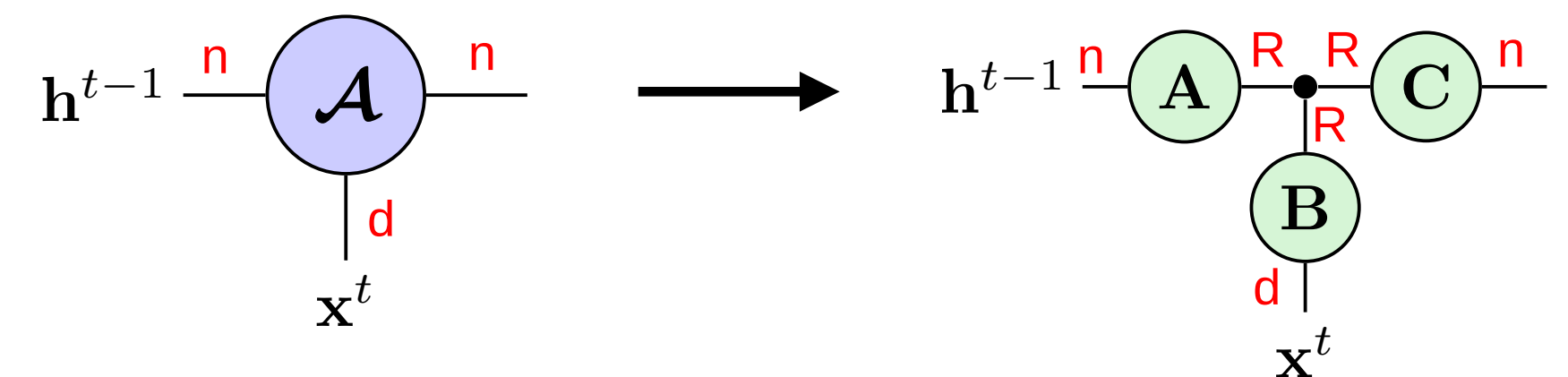
$$\mathcal{O}(d^3) \longrightarrow \mathcal{O}(Rd)$$

$$d = \max\{d_1, d_2, d_3\}$$

CPRNN

$$\sigma([\mathbf{A}, \mathbf{B}, \mathbf{C}] \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t + \mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$$

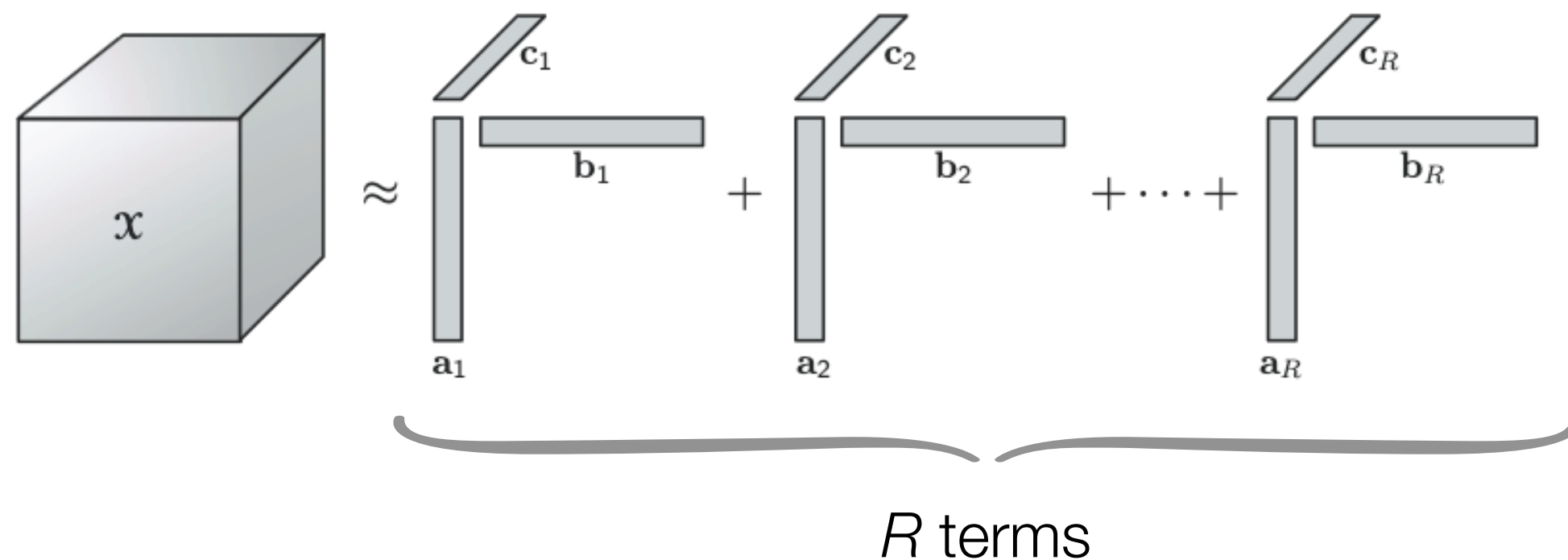
- 2nd-order term parameterized by CP decomposition



CP Decomposition & CPRNN

$$\mathcal{X} \approx [\mathbf{A}, \mathbf{B}, \mathbf{C}] \equiv \sum_{r=1}^R \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$$

Rank



T. Kolda and B. Bader, Tensor Decompositions and applications. SIAM REVIEW (2009)

Reduces the number of parameters:

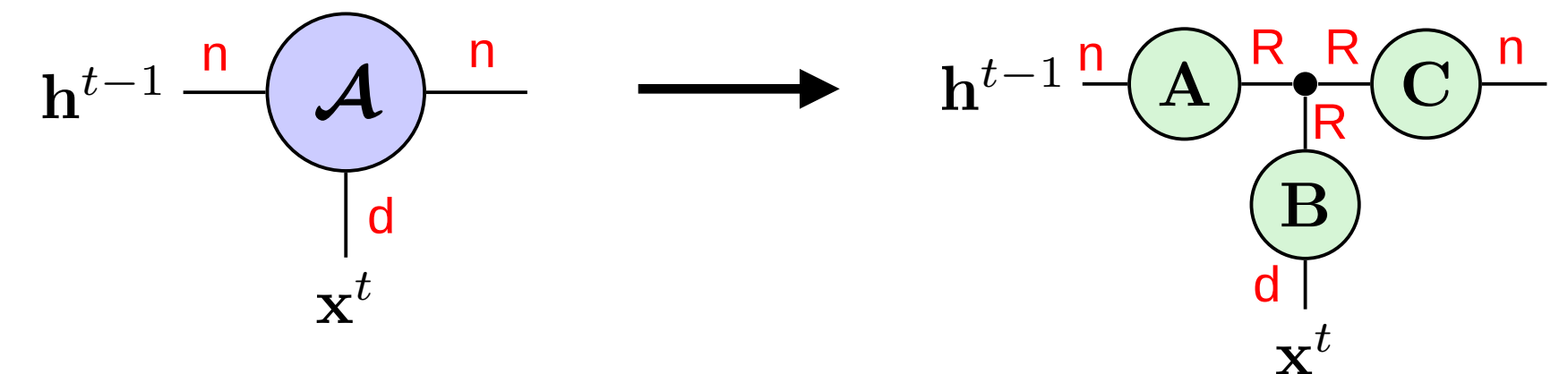
$$\mathcal{O}(d^3) \longrightarrow \mathcal{O}(Rd)$$

$$d = \max\{d_1, d_2, d_3\}$$

CPRNN

$$\sigma([\mathbf{A}, \mathbf{B}, \mathbf{C}] \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t + \mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$$

- 2nd-order term parameterized by CP decomposition

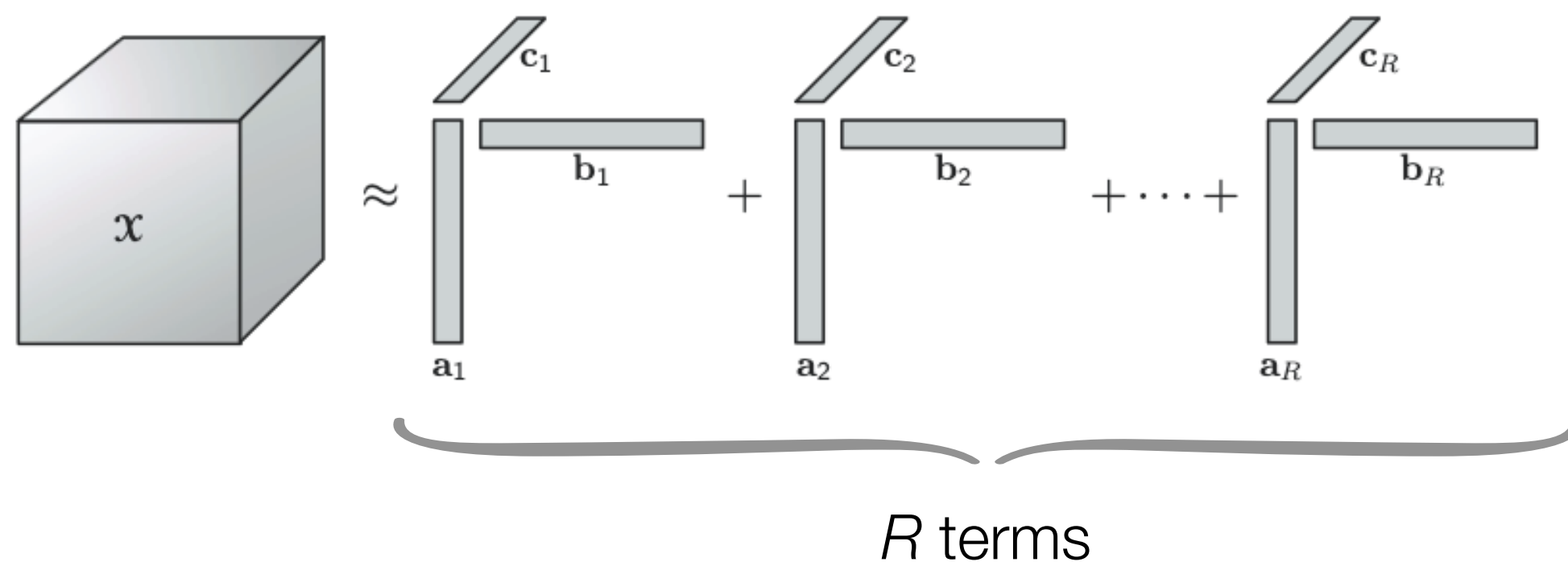


- Rank of CPRNN = Model hyper-parameter

CP Decomposition & CPRNN

$$\mathcal{X} \approx [\mathbf{A}, \mathbf{B}, \mathbf{C}] \equiv \sum_{r=1}^R \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$$

Rank



T. Kolda and B. Bader, Tensor Decompositions and applications. SIAM REVIEW (2009)

Reduces the number of parameters:

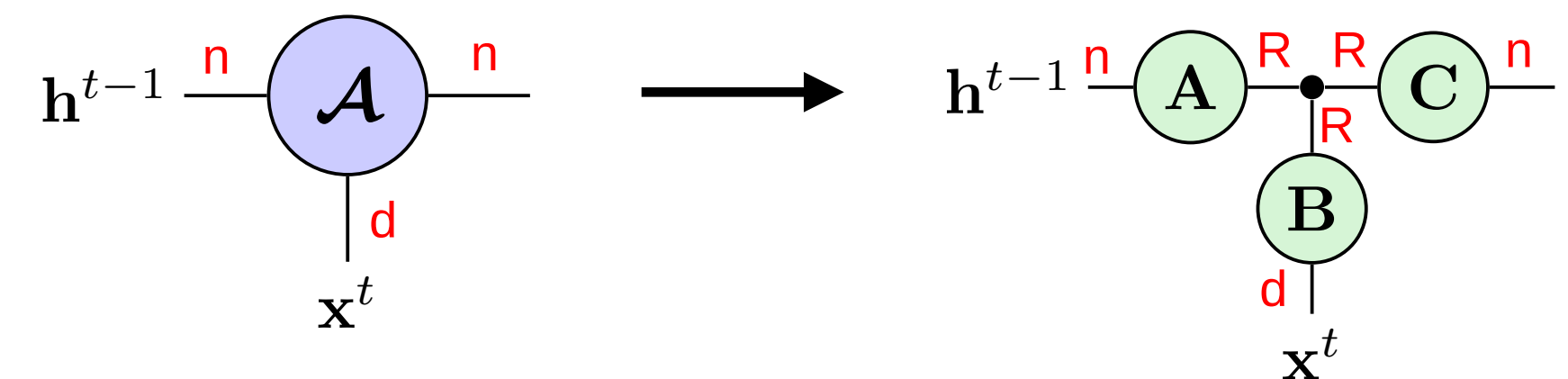
$$\mathcal{O}(d^3) \longrightarrow \mathcal{O}(Rd)$$

$$d = \max\{d_1, d_2, d_3\}$$

CPRNN

$$\sigma([\mathbf{A}, \mathbf{B}, \mathbf{C}] \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t + \mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$$

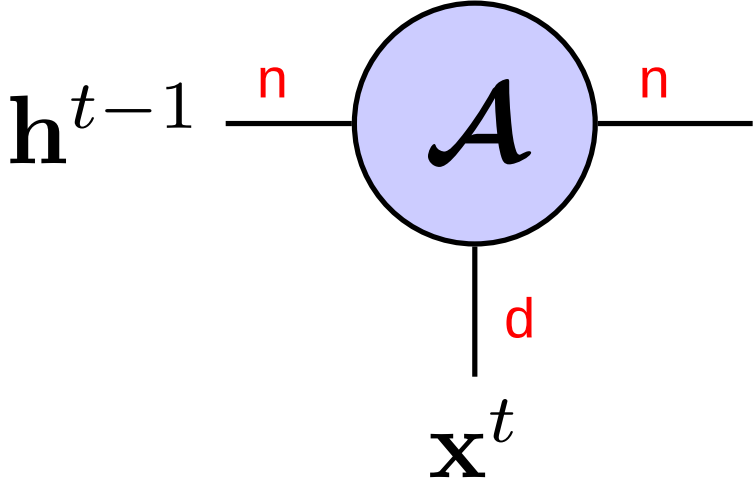
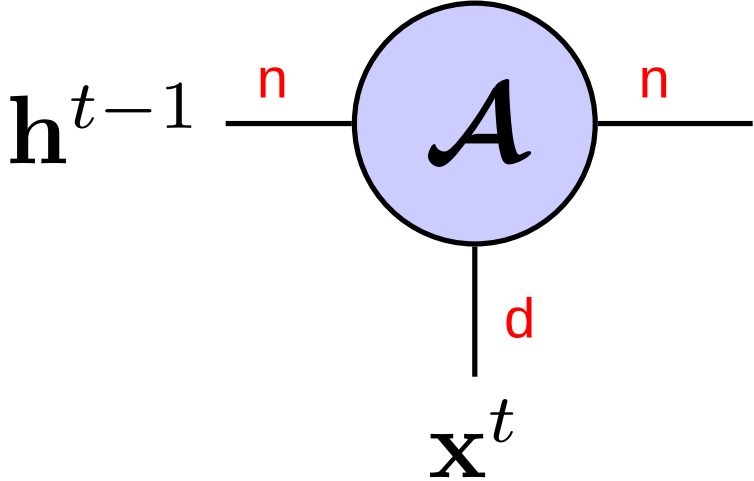
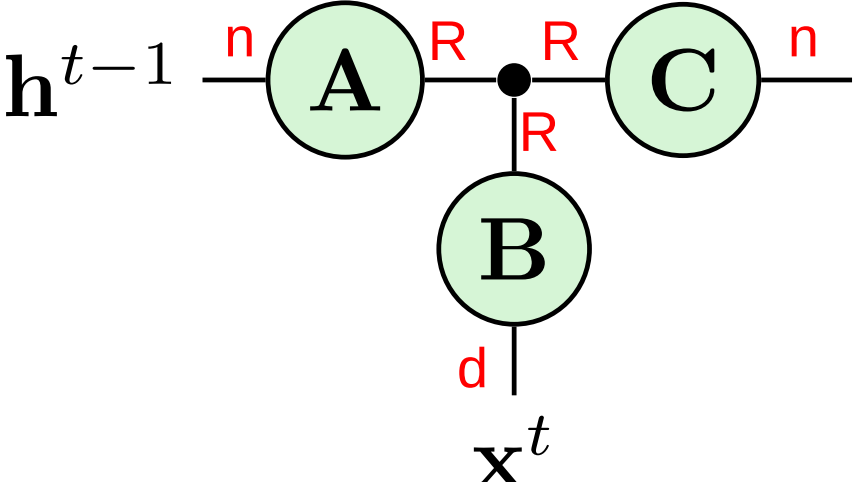
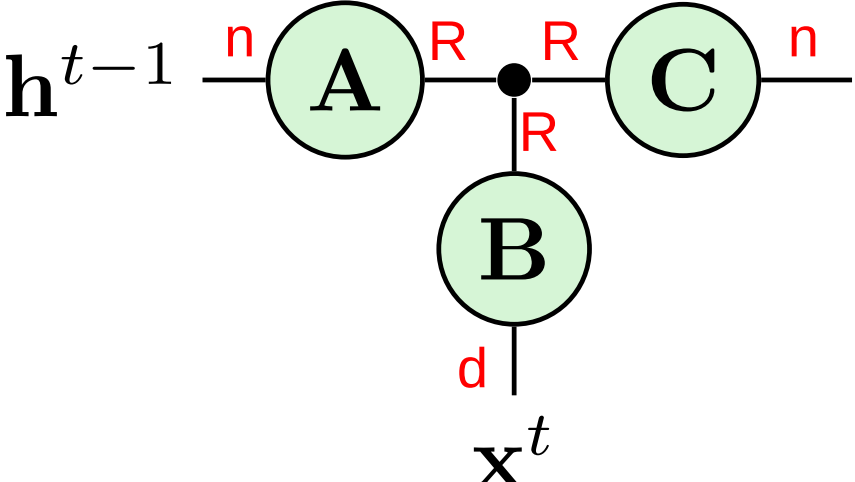
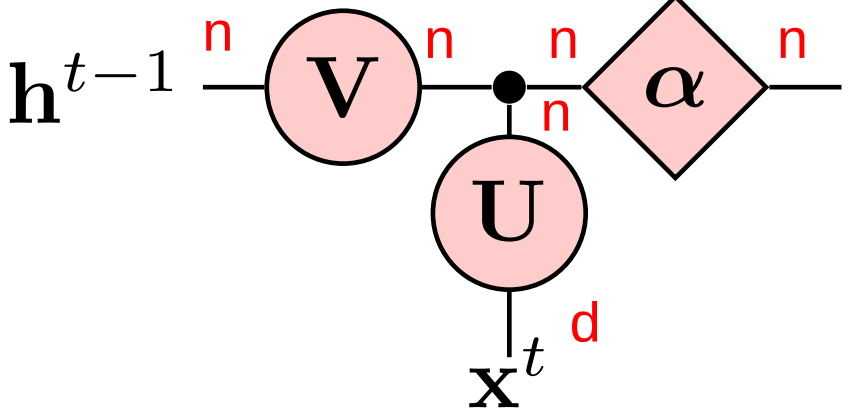
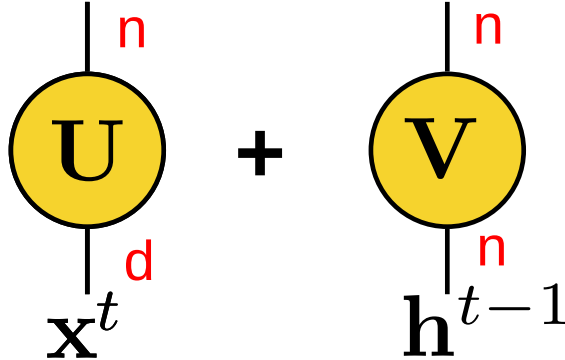
- 2nd-order term parameterized by CP decomposition



- Rank of CPRNN = Model hyper-parameter
- Empirically successful, but no theoretical analysis

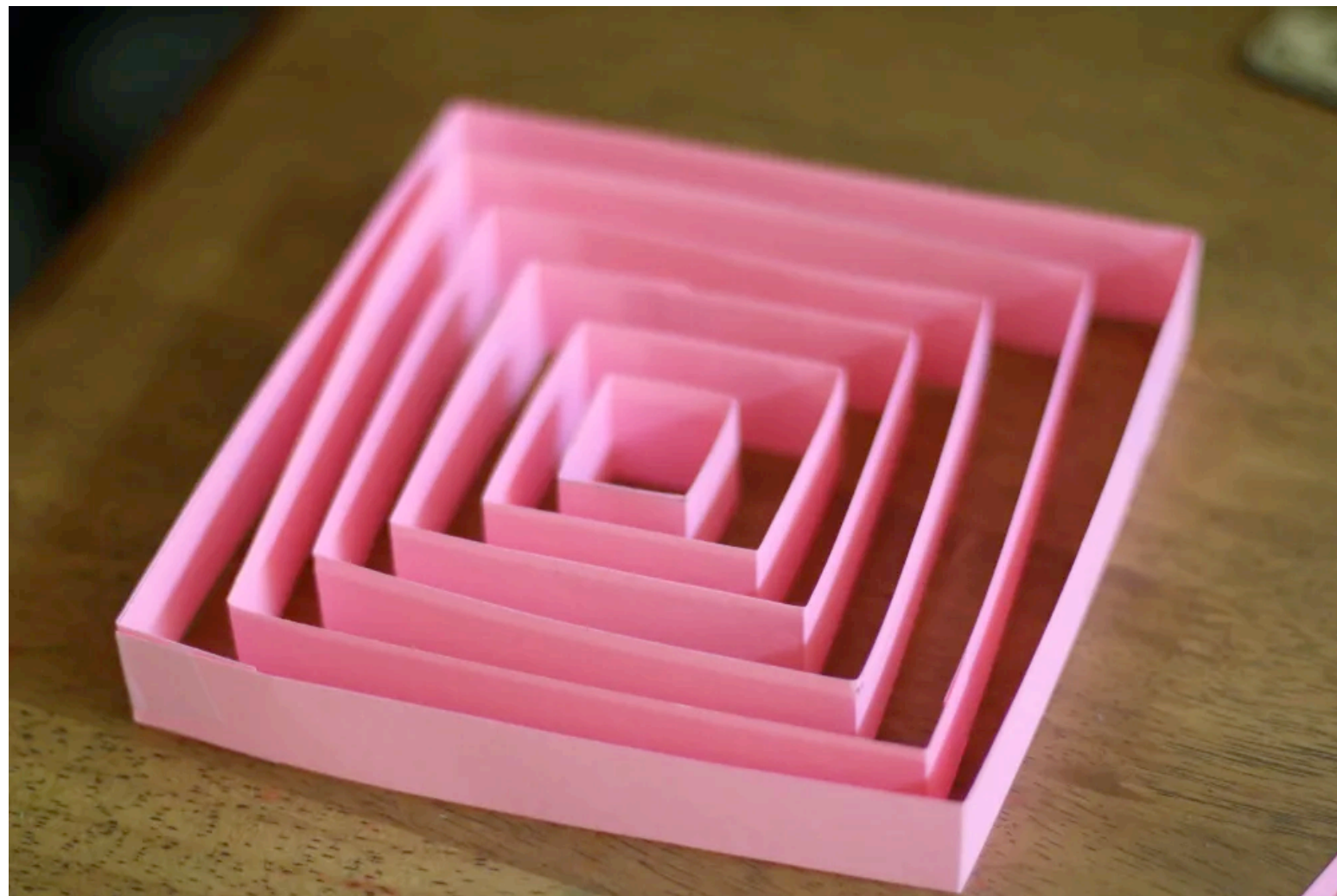
(I. Sutskever et al., Generating text with recurrent neural networks. ICML (2011))

Models

Models	Hidden state	Tensor Network (2nd order terms)
2RNN	$\sigma(\mathcal{A} \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t + \mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$	
BIRNN	$\sigma(\mathcal{A} \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t)$	
CPRNN	$\sigma([\mathbf{A}, \mathbf{B}, \mathbf{C}] \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t + \mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$	
CPBIRNN	$\sigma([\mathbf{A}, \mathbf{B}, \mathbf{C}] \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t)$	
MIRNN	$\sigma(\alpha \odot \mathbf{V}\mathbf{h}^{t-1} \odot \mathbf{U}\mathbf{x}^t + \beta_1 \odot \mathbf{V}\mathbf{h}^{t-1} + \beta_2 \odot \mathbf{U}\mathbf{x}^t + \mathbf{b})$	
RNN	$\sigma(\mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$	

Questions

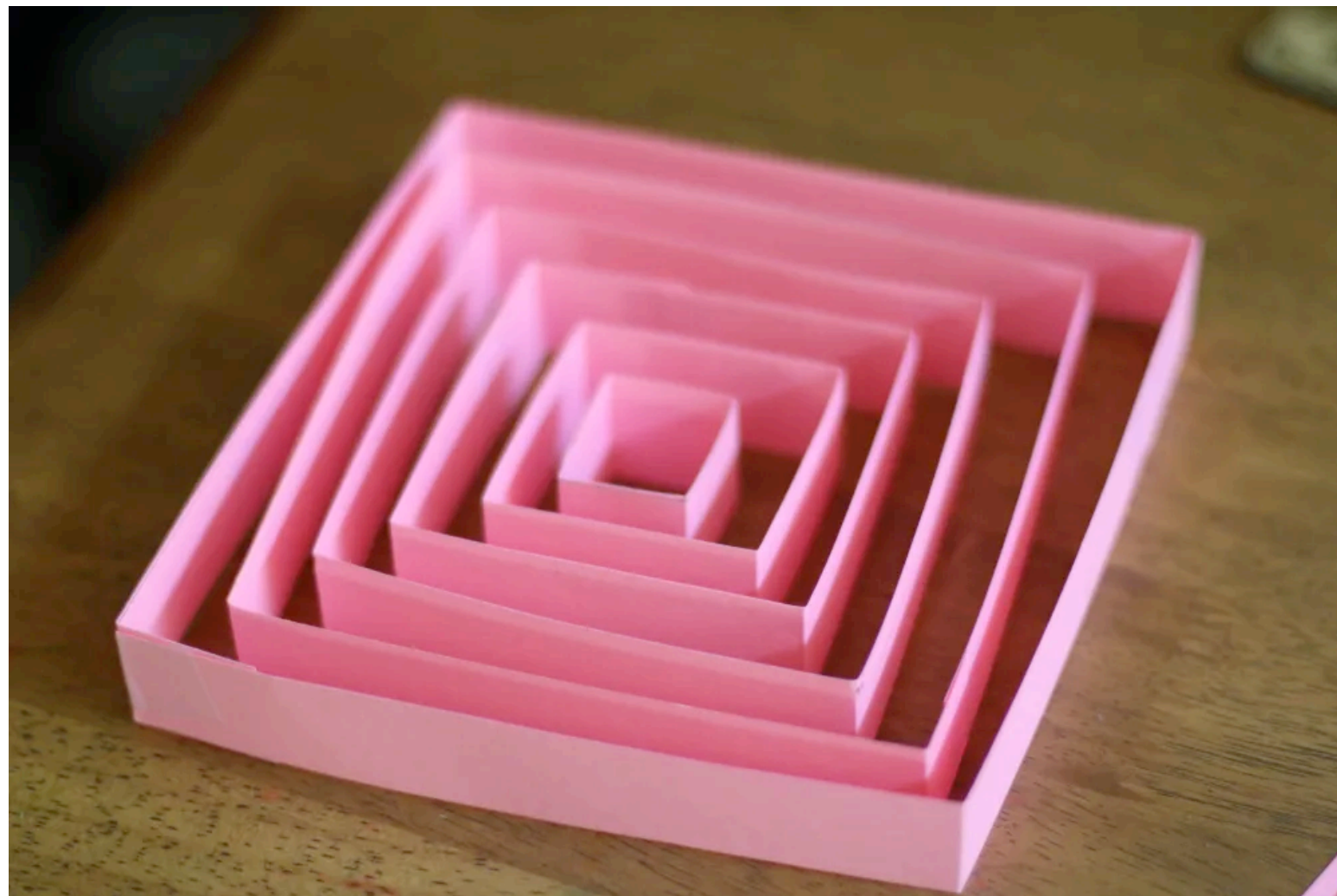
Nested family in tensor space



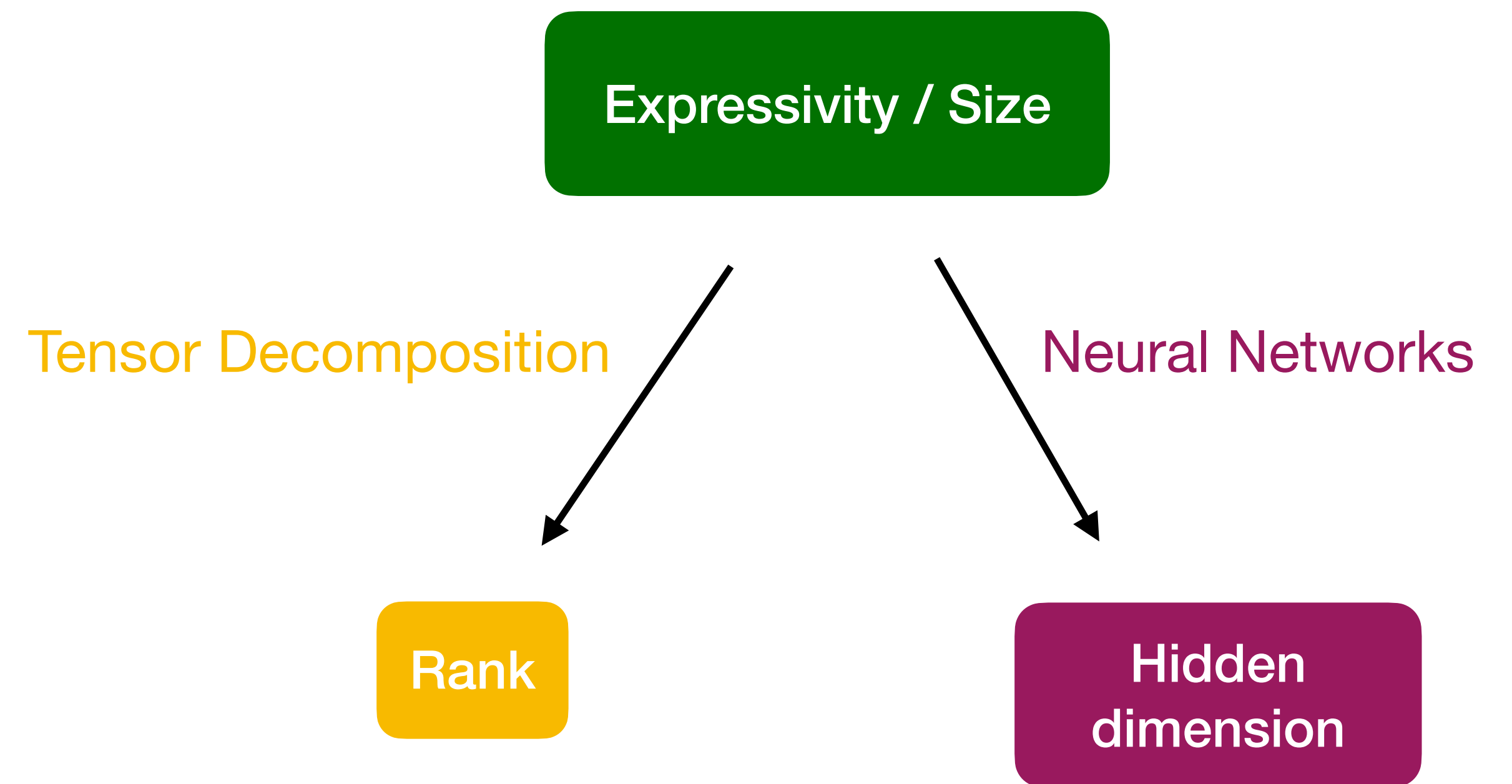
Increasing rank increases tensor
capacity up to saturation

Questions

Nested family in tensor space

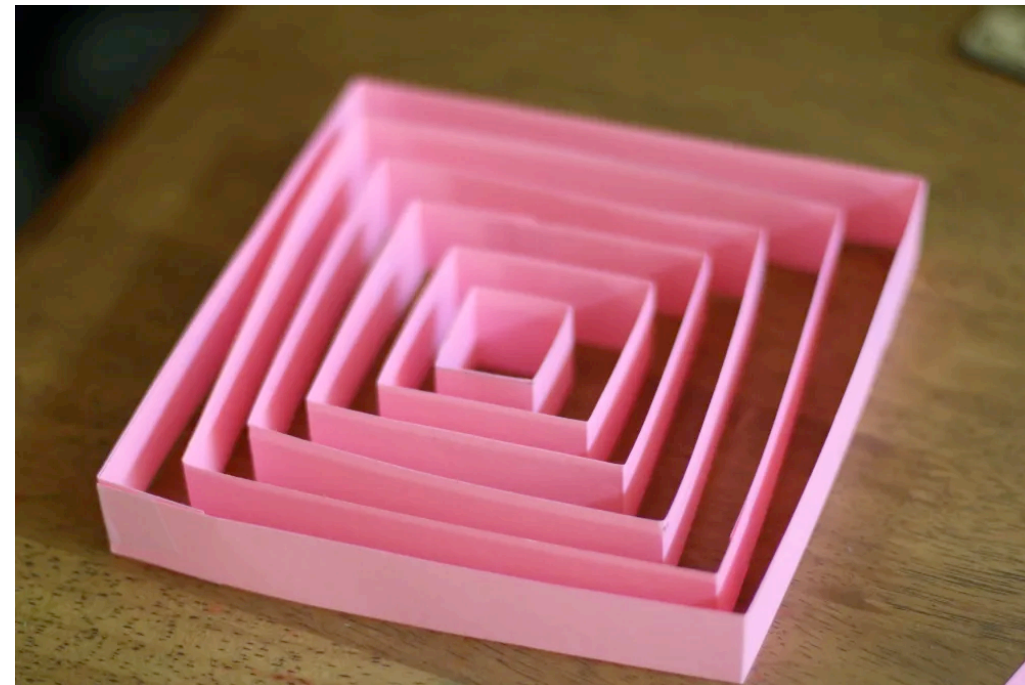


Increasing rank increases tensor capacity up to saturation

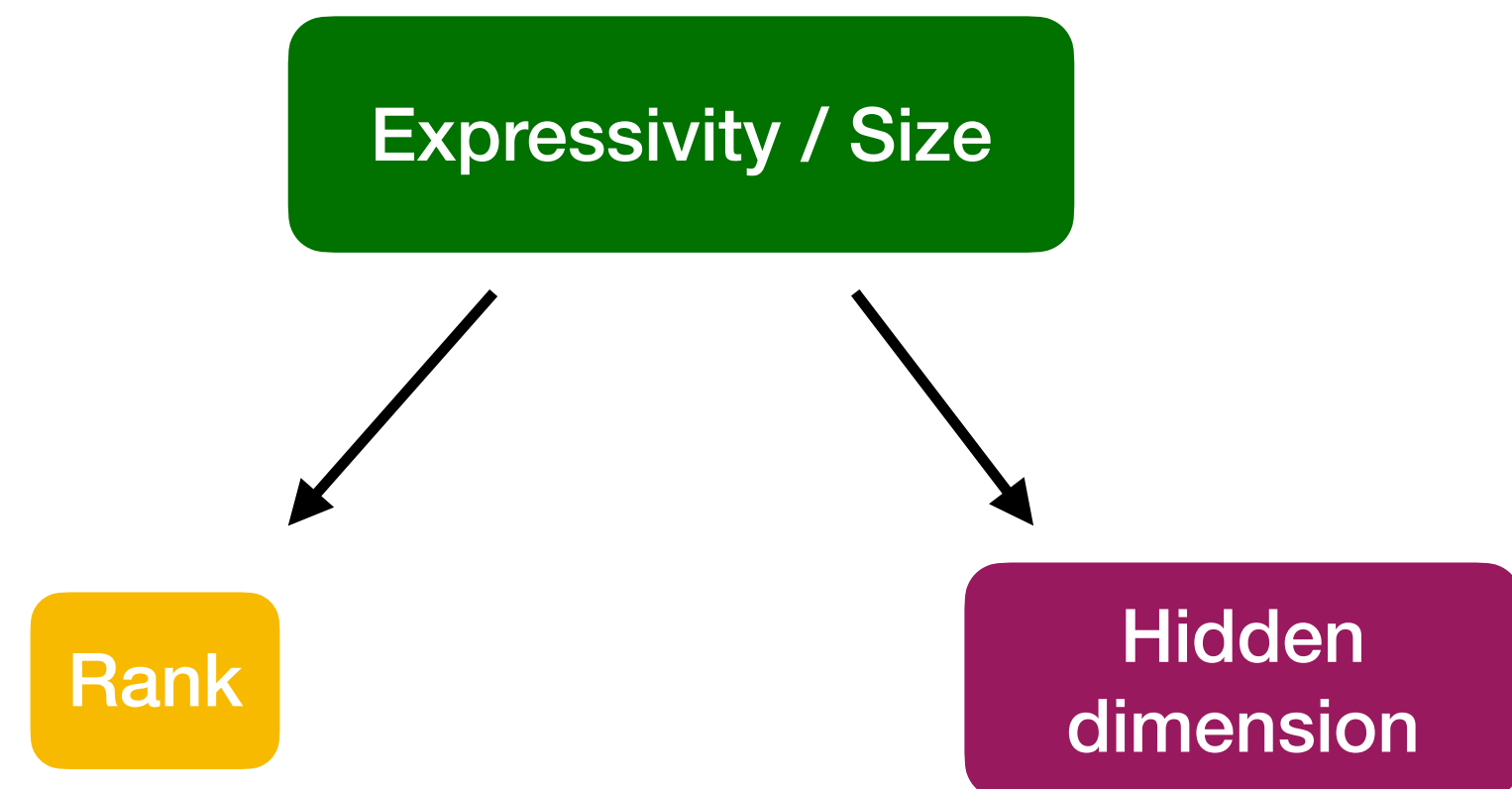


Questions

Nested family in tensor space

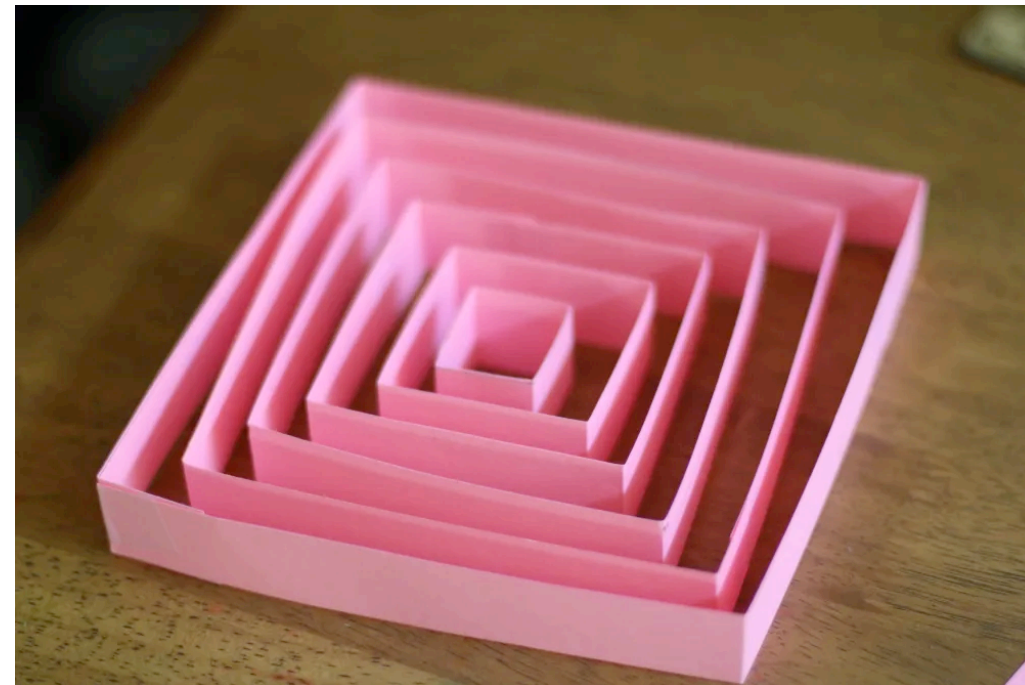


How does increasing the capacity of the tensor parameter relate to the **expressivity** of the CPRNN?



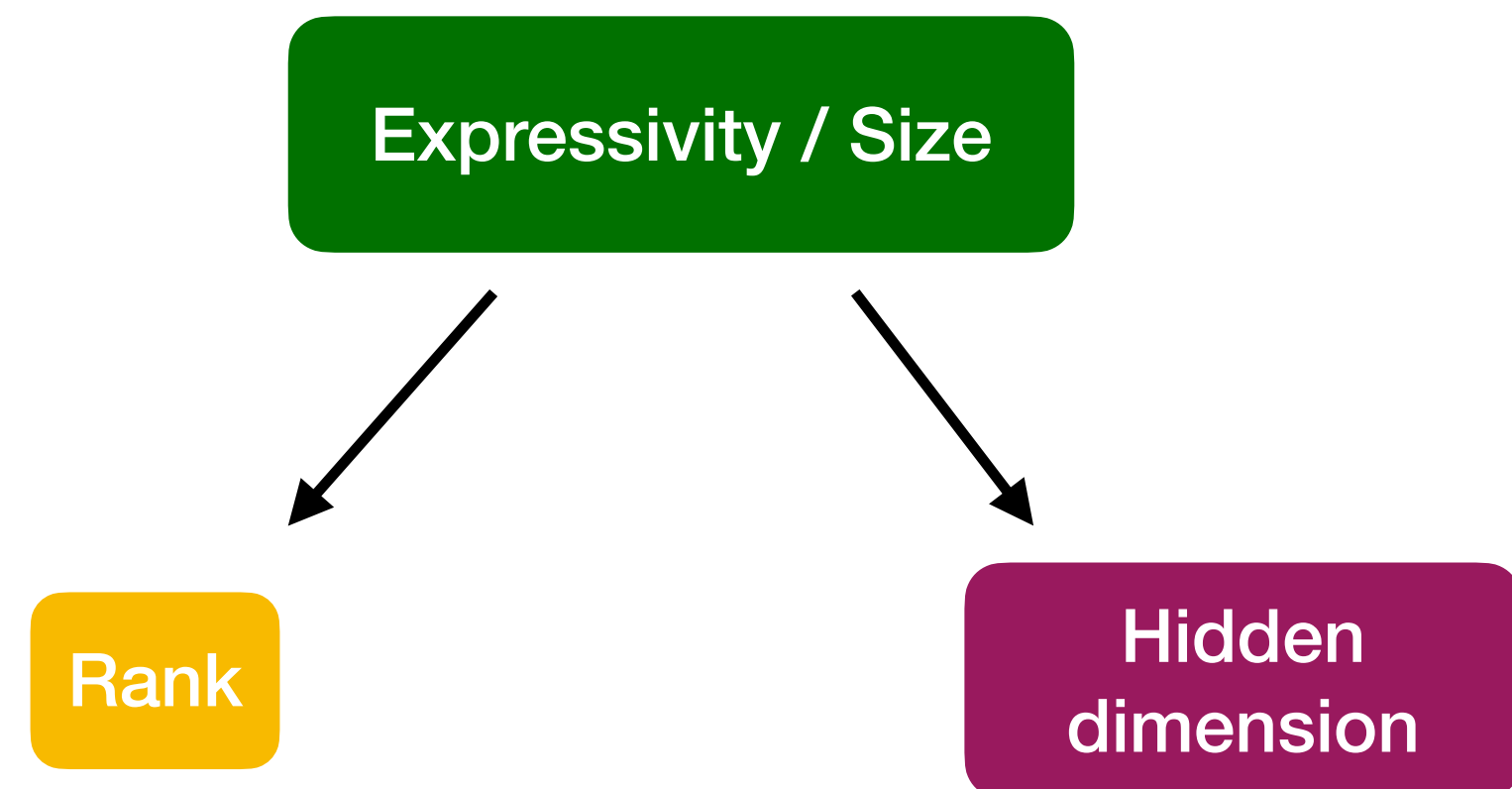
Questions

Nested family in tensor space



How does increasing the capacity of the tensor parameter relate to the **expressivity** of the CPRNN?

How does the **point of saturation** in **tensor space** translate in **function space**?



Questions

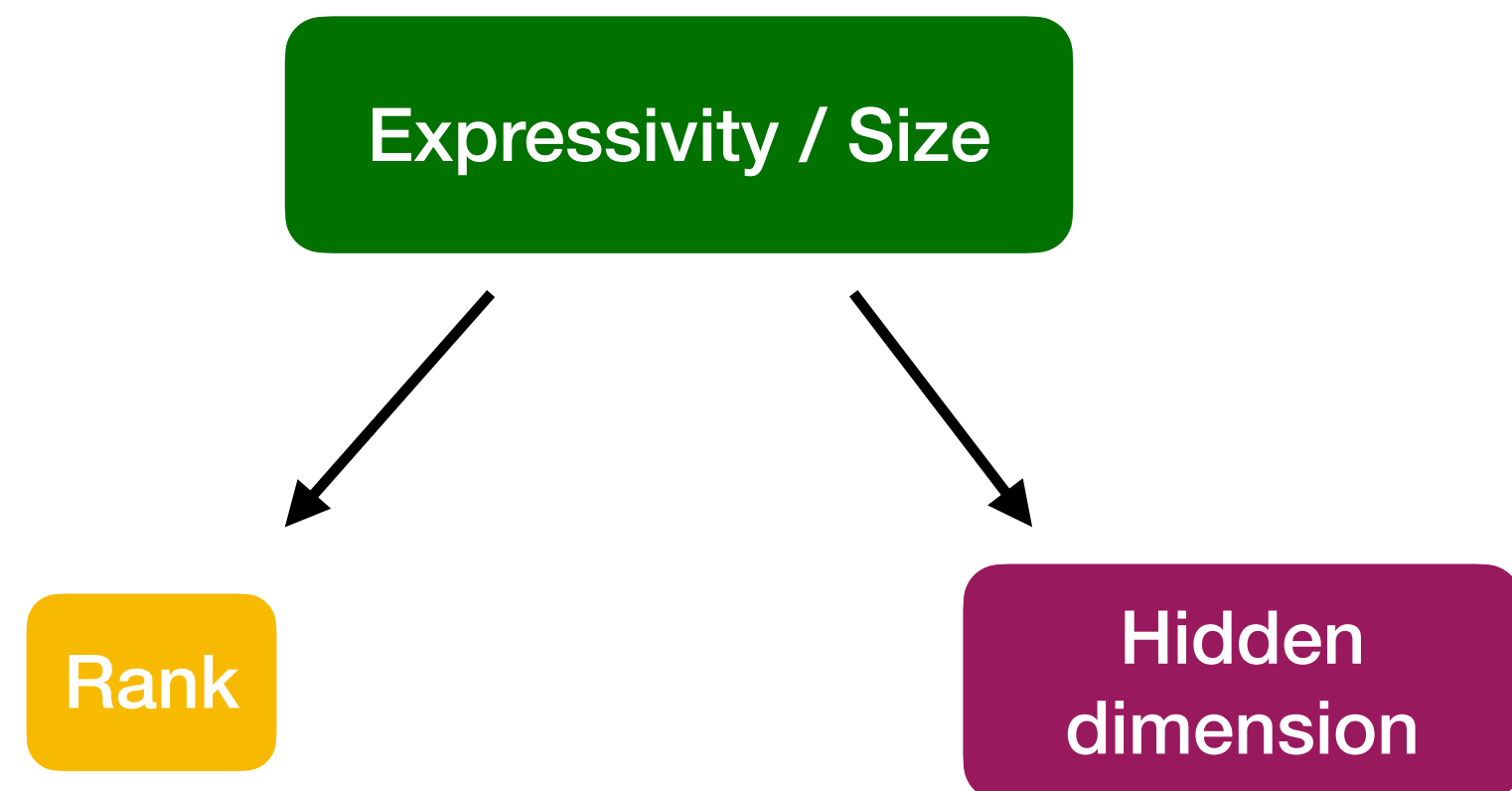
Nested family in tensor space



How does increasing the capacity of the tensor parameter relate to the **expressivity** of the CPRNN?

How does the **point of saturation** in **tensor space** translate in **function space**?

How do the **rank** and **hidden size** **interplay** in controlling the CPRNNs capacity?



Relating Tensor Space to Function Space through Rank

Relating Tensor Space to Function Space through Rank

Definition : $\mathcal{H}_{\text{CPRNN}}(R, n)$

Set of functions h mapping
input sequences to hidden state sequences
computed by CPRNNs of
rank R and hidden state n

Relating Tensor Space to Function Space through Rank

Definition : $\mathcal{H}_{\text{CPRNN}}(R, n)$

Set of functions h mapping

 computed by CPRNNs of
 rank R and hidden state n

$$h : (\mathbf{x}^1, \dots, \mathbf{x}^T) \mapsto (\mathbf{h}^1, \dots, \mathbf{h}^T)$$

Relating Tensor Space to Function Space through Rank

Definition : $\mathcal{H}_{\text{CPRNN}}(R, n)$

Set of functions h mapping
input sequences to hidden state sequences
computed by CPRNNs of
rank R and hidden state n

$$h : (\mathbf{x}^1, \dots, \mathbf{x}^T) \mapsto (\mathbf{h}^1, \dots, \mathbf{h}^T)$$

$$\mathbf{h}^t = \sigma(\llbracket \mathbf{A}, \mathbf{B}, \mathbf{C} \rrbracket \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t + \mathbf{V} \mathbf{h}^{t-1} + \mathbf{U} \mathbf{x}^t + \mathbf{b})$$

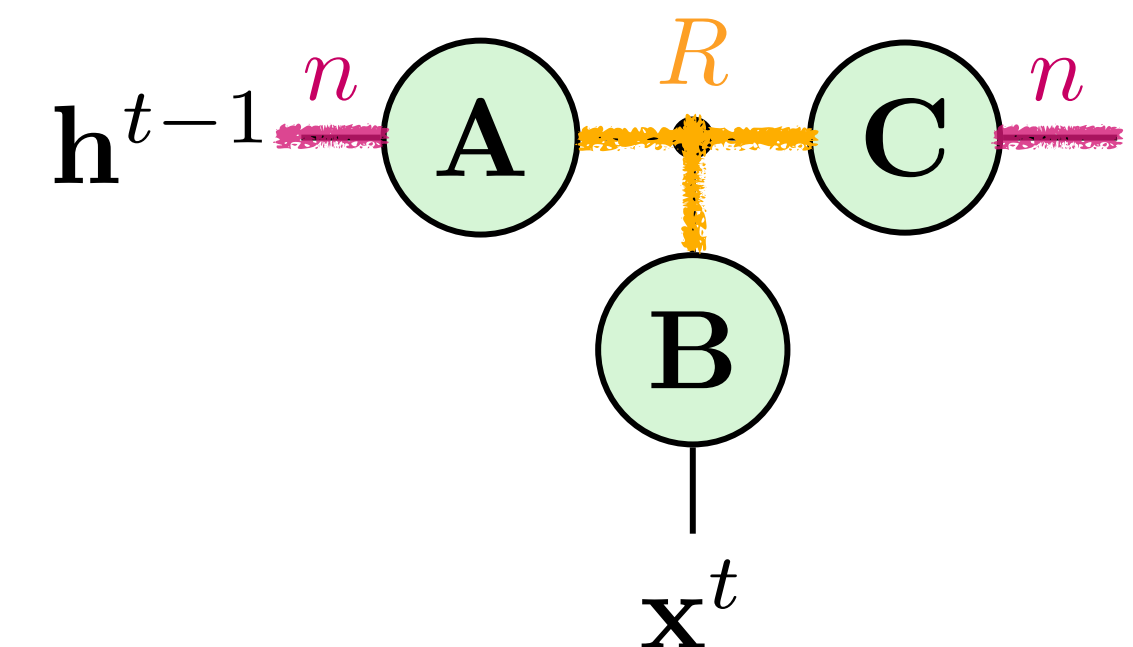
Relating Tensor Space to Function Space through Rank

Definition : $\mathcal{H}_{\text{CPRNN}}(R, n)$

Set of functions h mapping
input sequences to hidden state sequences
computed by CPRNNs of
rank R and hidden state n

$$h : (\mathbf{x}^1, \dots, \mathbf{x}^T) \mapsto (\mathbf{h}^1, \dots, \mathbf{h}^T)$$

$$\mathbf{h}^t = \sigma(\llbracket \mathbf{A}, \mathbf{B}, \mathbf{C} \rrbracket \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t + \mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$$



Relating Tensor Space to Function Space through Rank

Definition : $\mathcal{H}_{\text{CPRNN}}(R, n)$

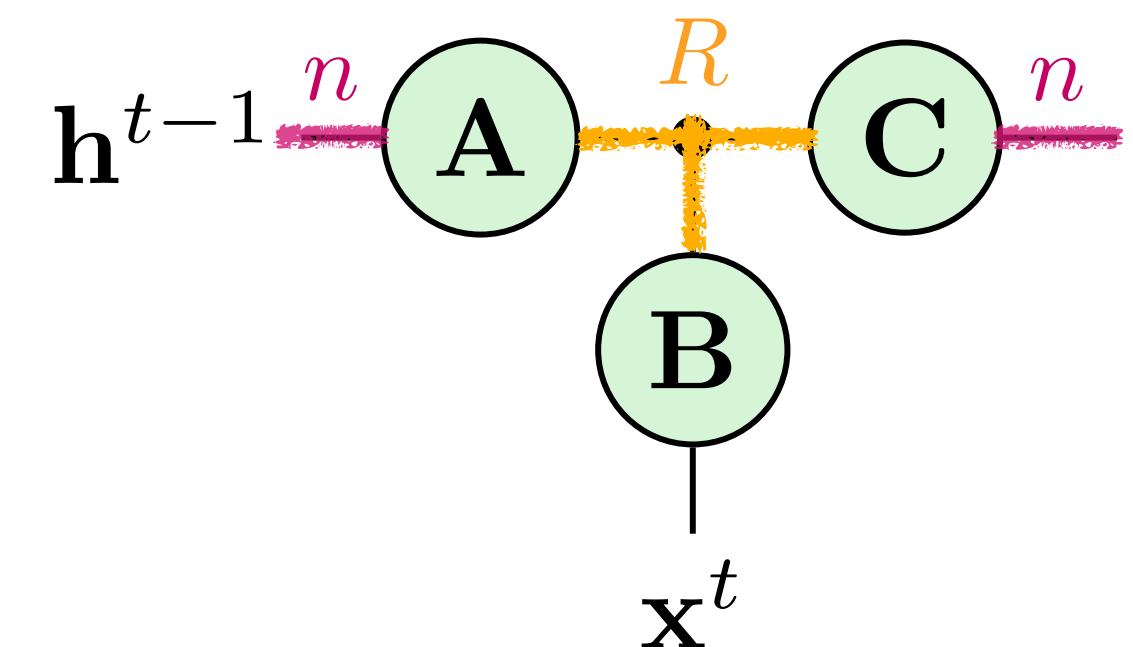
Set of functions h mapping
input sequences to hidden state sequences
computed by CPRNNs of
rank R and hidden state n

Similarly, we define :

$$\mathcal{H}_{\text{RNN}}(n) \quad \mathcal{H}_{\text{MIRNN}}(n) \quad \mathcal{H}_{2\text{RNN}}(n)$$

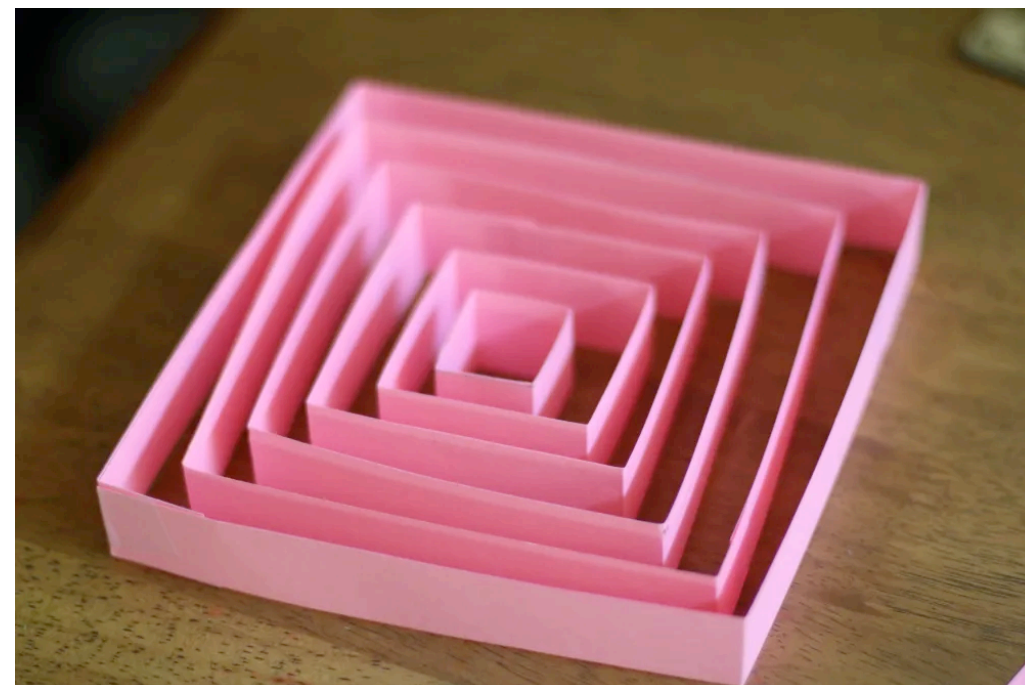
$$h : (\mathbf{x}^1, \dots, \mathbf{x}^T) \mapsto (\mathbf{h}^1, \dots, \mathbf{h}^T)$$

$$\mathbf{h}^t = \sigma(\llbracket \mathbf{A}, \mathbf{B}, \mathbf{C} \rrbracket \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t + \mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$$



Questions

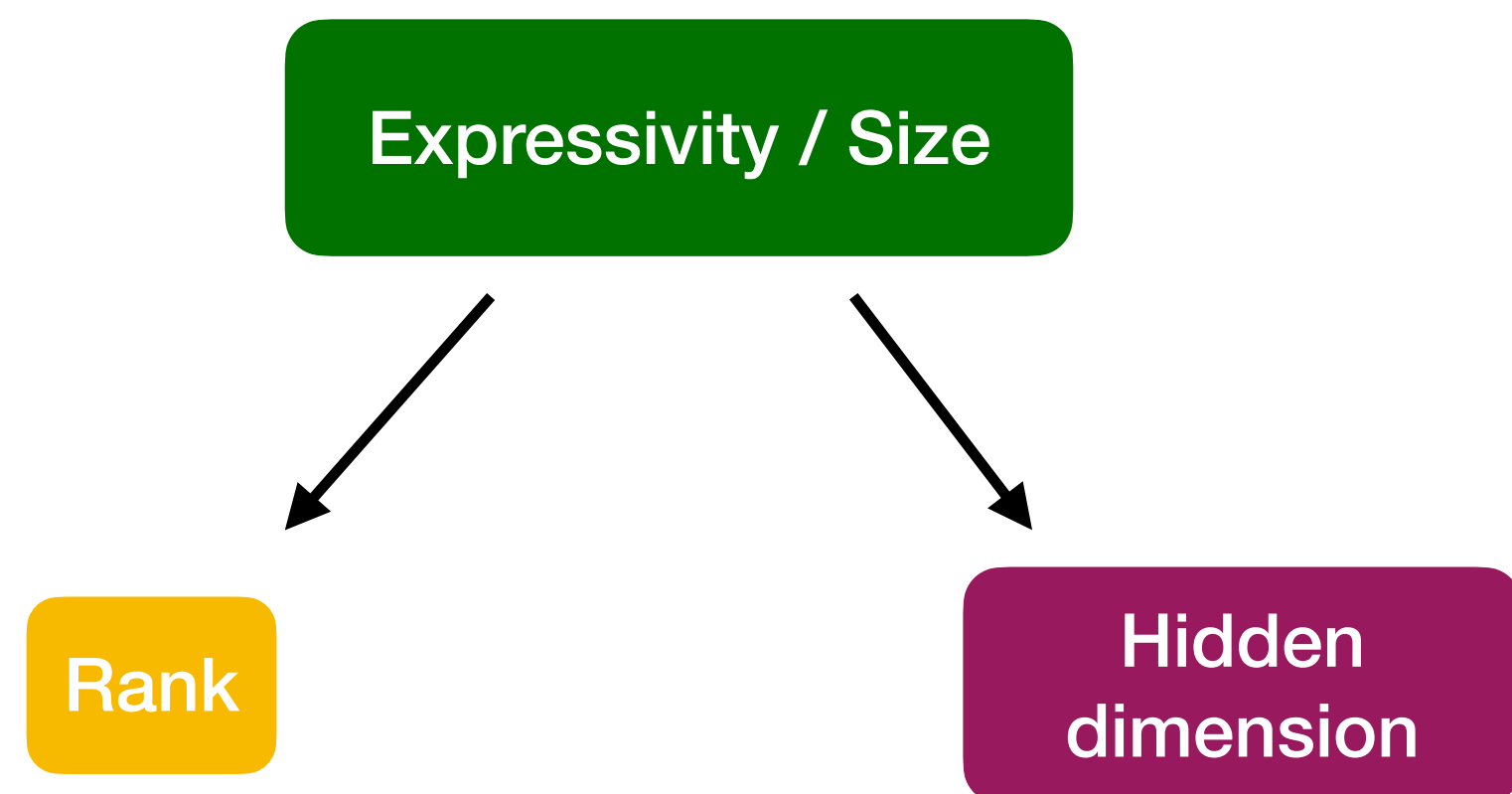
Nested family in tensor space



How does increasing the capacity of the tensor parameter relate to the **expressivity** of the CPRNN?

How does the **point of saturation** in **tensor space** translate in **function space**?

How do the **rank** and **hidden size** **interplay** in controlling the CPRNNs capacity?



Questions

Relations of **inclusions**

$=$ \subsetneq \subseteq

between **sets**

$$\mathcal{H}_{\text{CPRNN}}(R, n) \quad ? \quad \mathcal{H}_{\text{CPRNN}}(R + 1, n)$$

How does increasing the capacity of the tensor parameter relate to the **expressivity** of the CPRNN?

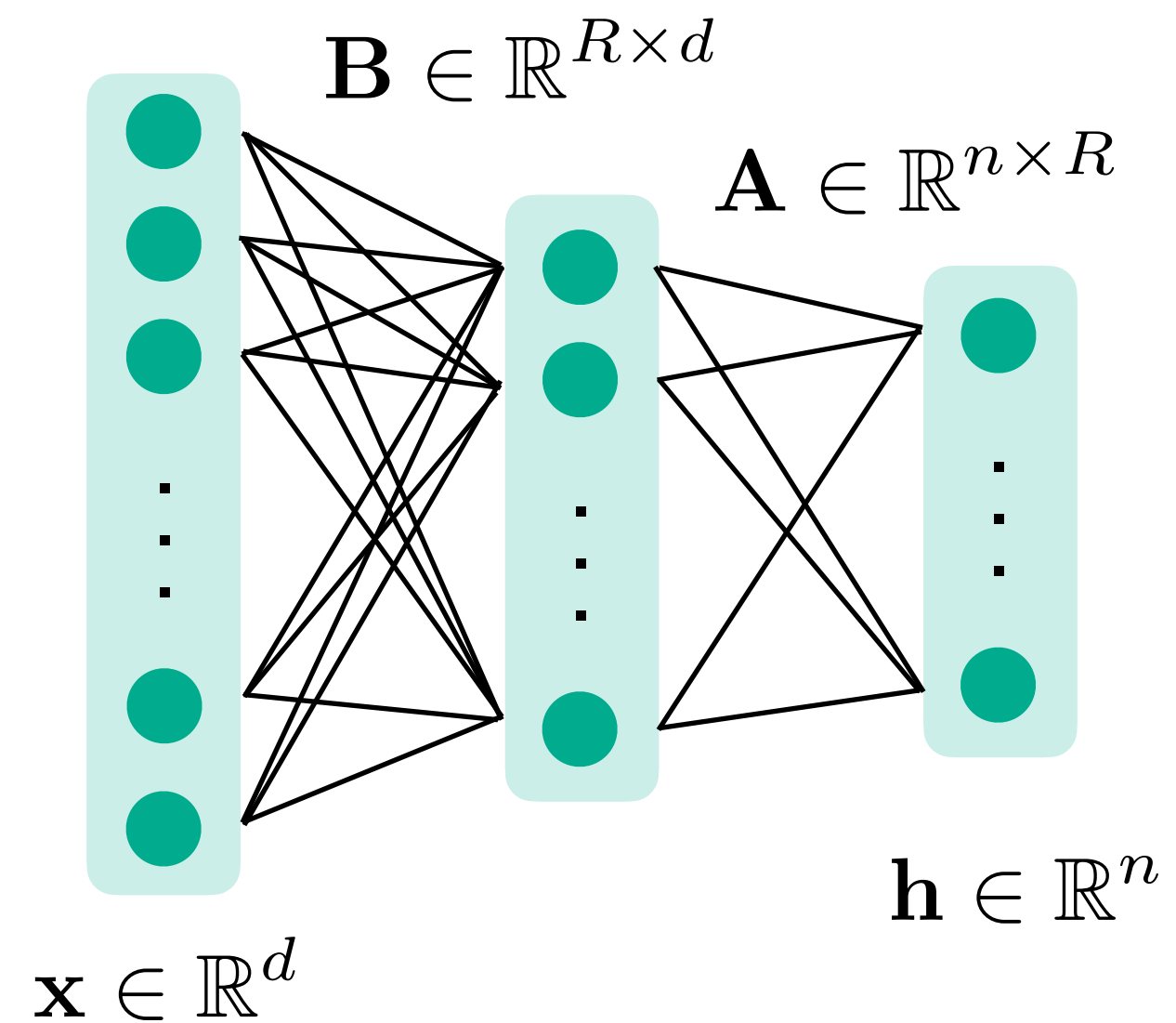
How does the **point of saturation** in **tensor space** translate in **function space**?

How do the **rank** and **hidden size** **interplay** in controlling the CPRNNs capacity?

Relating Tensor Space to Function Space through Rank

Intuition : Consider a linear non recurrent model

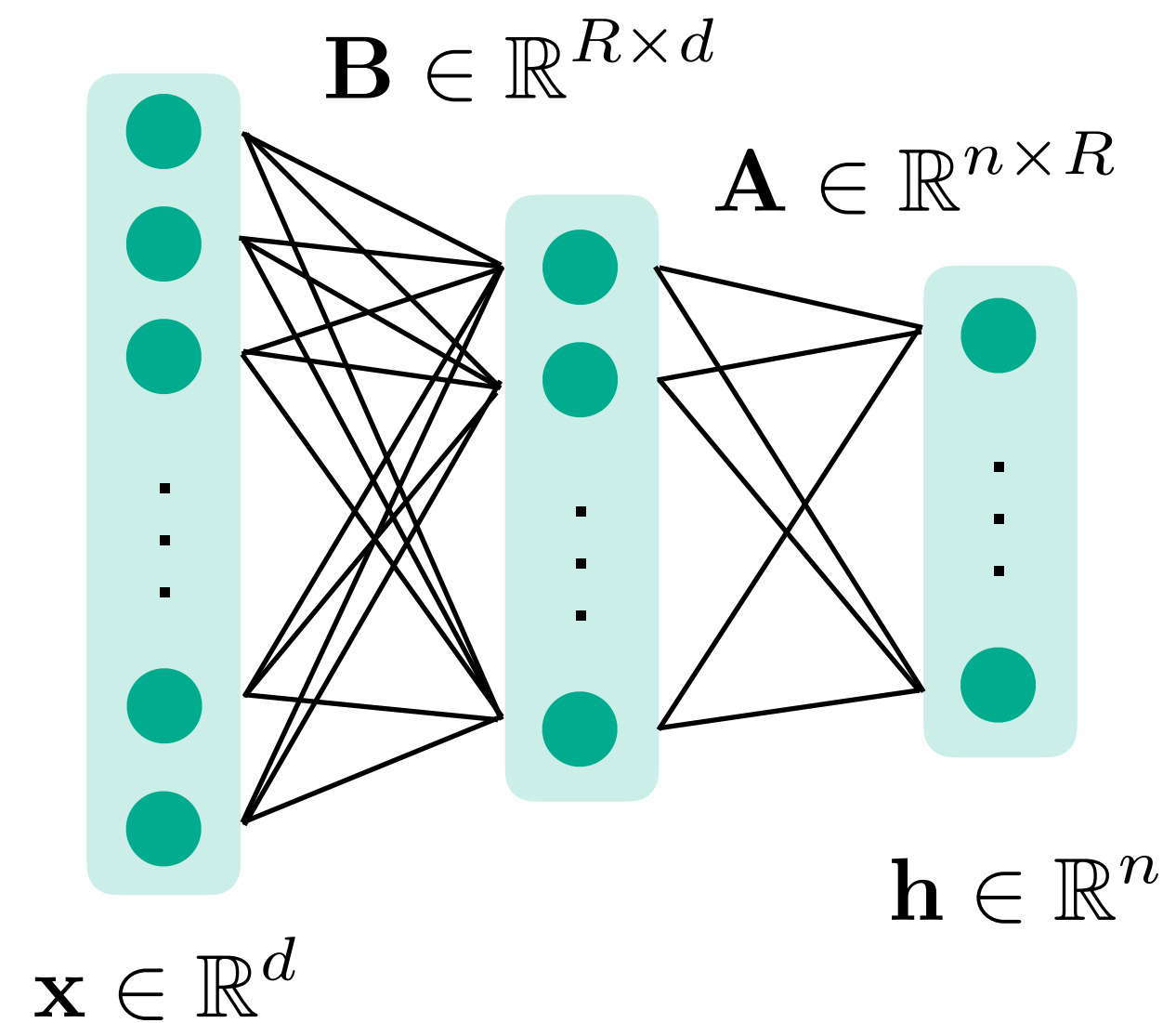
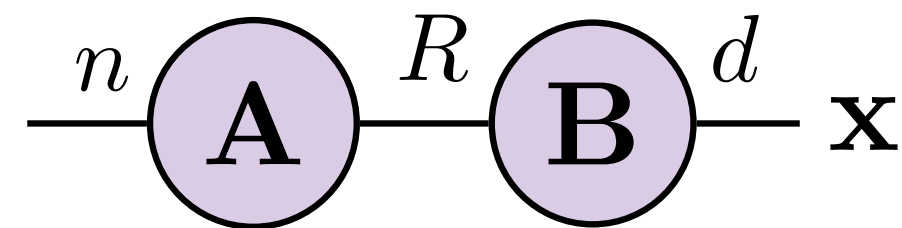
$$h(\mathbf{x}) = \mathbf{A}\mathbf{B}\mathbf{x}$$



Relating Tensor Space to Function Space through Rank

Intuition : Consider a linear non recurrent model

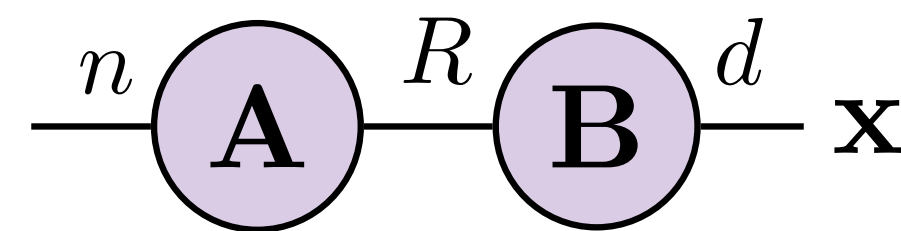
$$h(\mathbf{x}) = \mathbf{A}\mathbf{B}\mathbf{x}$$



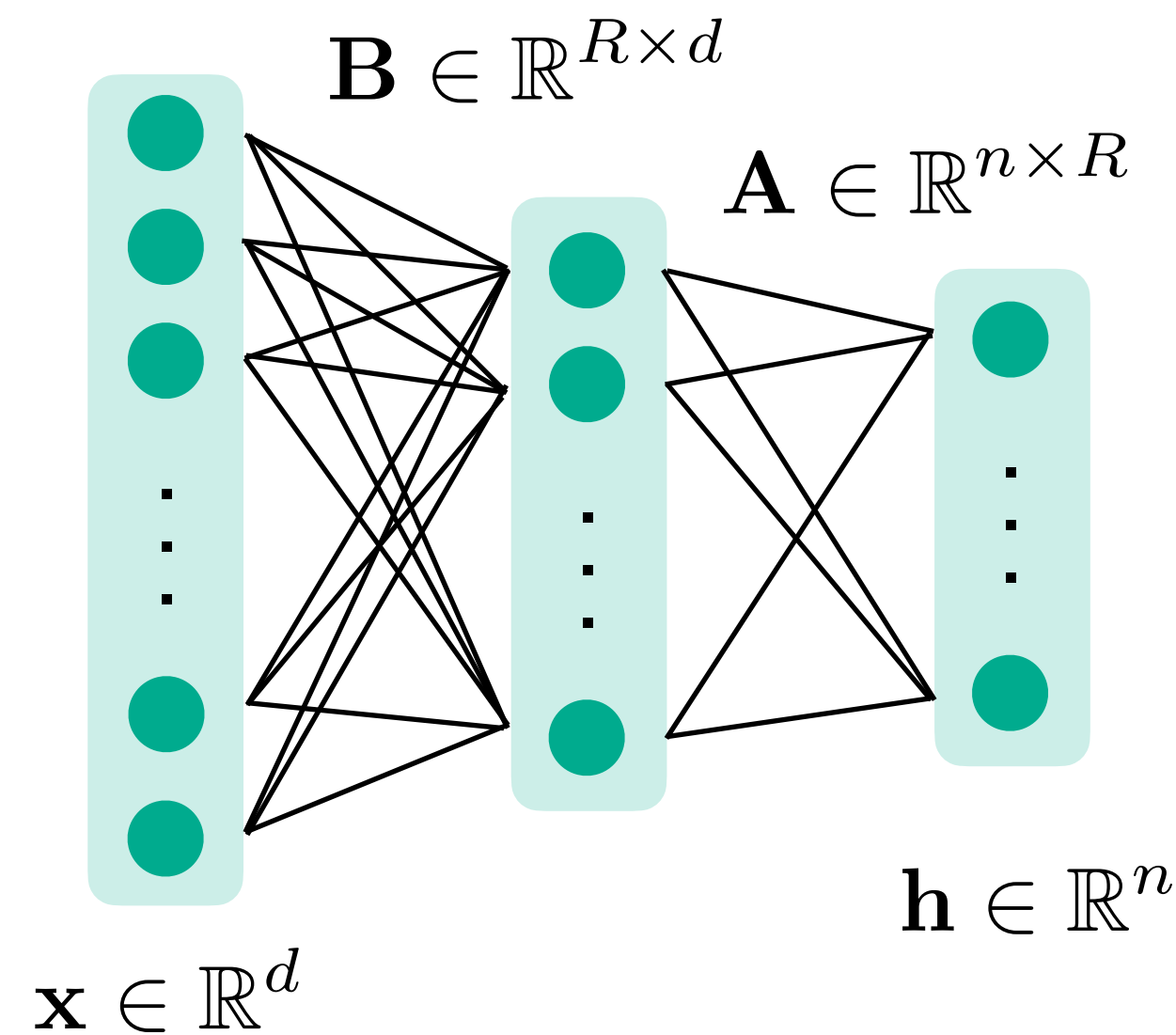
Relating Tensor Space to Function Space through Rank

Intuition : Consider a linear non recurrent model

$$h(\mathbf{x}) = \mathbf{A}\mathbf{B}\mathbf{x}$$



$$\mathcal{H}_{AB}(R, n)$$



Strict Inclusion

$$\mathcal{H}_{AB}(R, n) \subsetneq \mathcal{H}_{AB}(R+1, n)$$

Saturation

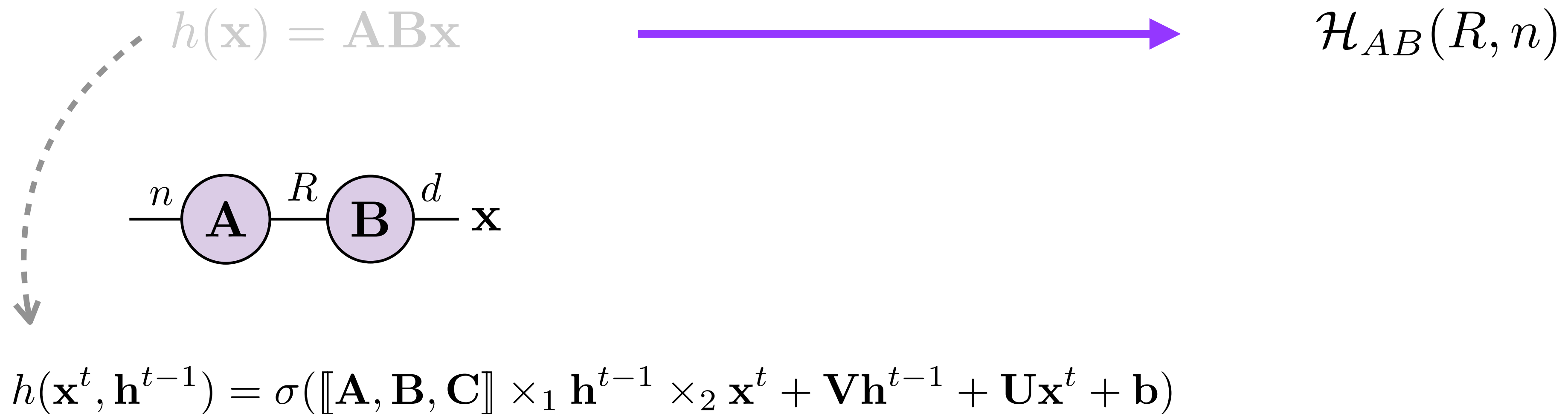
$$\mathcal{H}_{AB}(R, n) = \mathcal{H}_{AB}(R+1, n)$$

$$R_{\max} = \min\{n, d\}$$

R

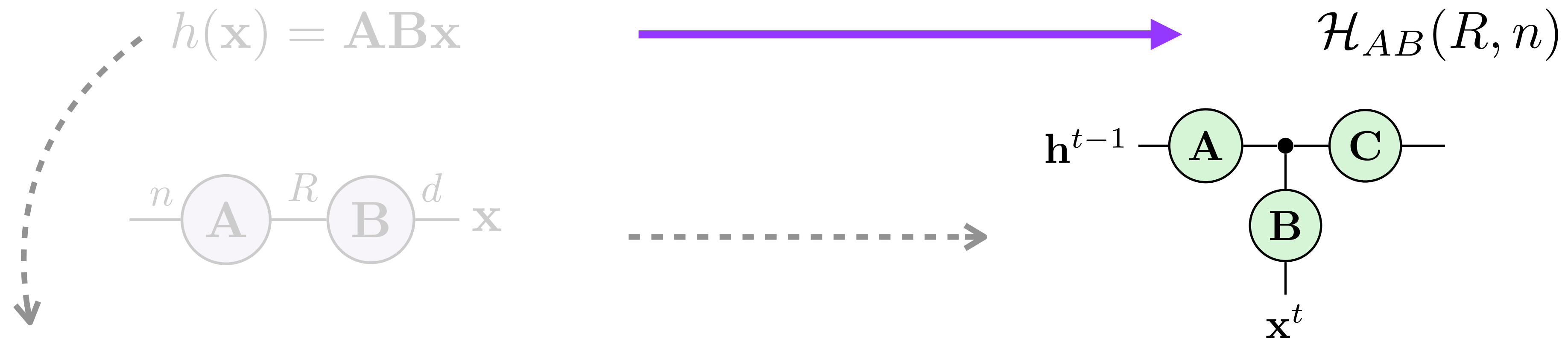
Relating Tensor Space to Function Space through Rank

Intuition : For CPRNNs



Relating Tensor Space to Function Space through Rank

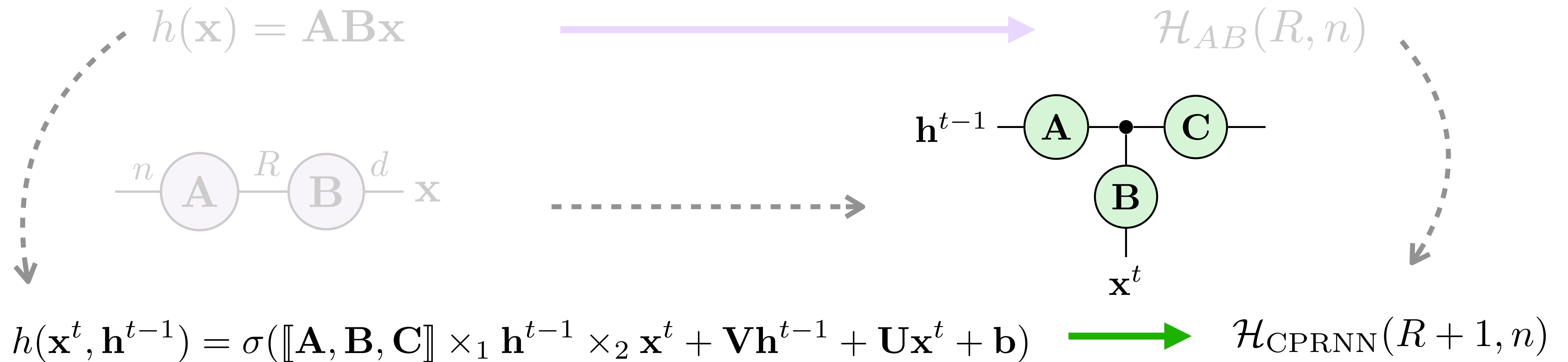
Intuition : For CPRNNs



$$h(\mathbf{x}^t, \mathbf{h}^{t-1}) = \sigma(\llbracket \mathbf{A}, \mathbf{B}, \mathbf{C} \rrbracket \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t + \mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$$

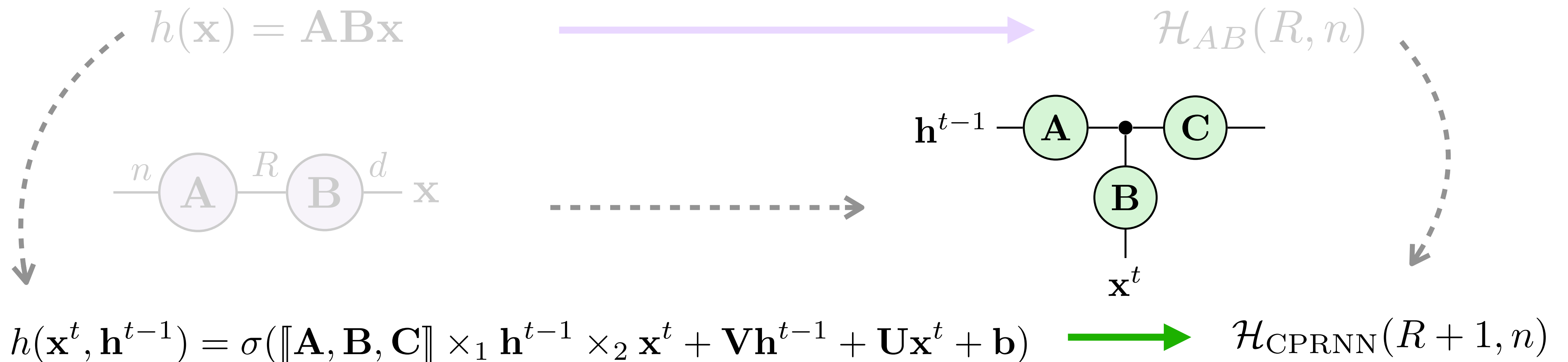
Relating Tensor Space to Function Space through Rank

Intuition : For CPRNNs



Relating Tensor Space to Function Space through Rank

Intuition : For CPRNNs



Strict Inclusion?

Saturation?

$$\mathcal{H}_{\text{CPRNN}}(R, n) \quad ? \quad \mathcal{H}_{\text{CPRNN}}(R+1, n)$$

$$\mathcal{H}_{\text{CPRNN}}(R, n) \quad ? \quad \mathcal{H}_{\text{CPRNN}}(R+1, n)$$

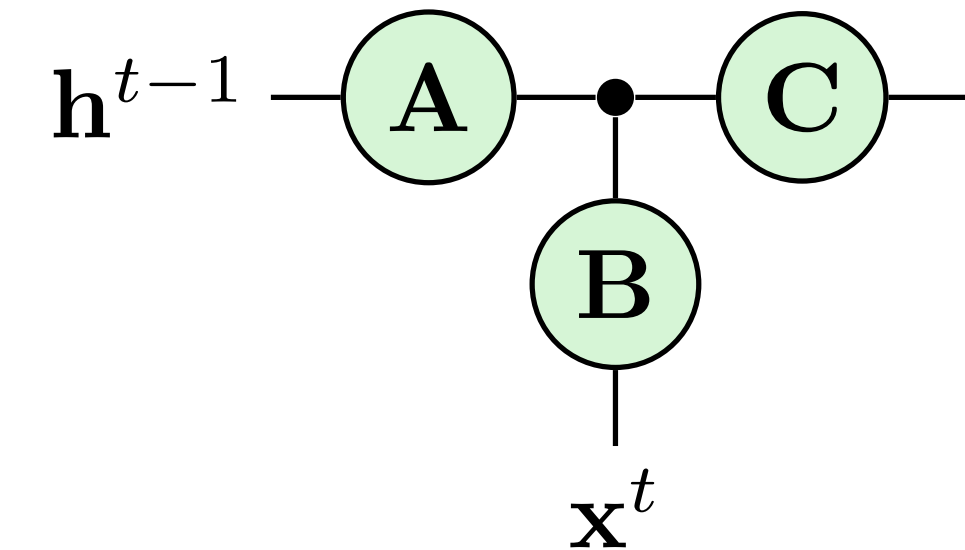
?

R

Relating Tensor Space to Function Space through Rank

Intuition : For CPRNNs, non trivial!

- 1) R_{\max} : maximal CP rank is unknown
- 2) Recurrence : \mathbf{h}^{t-1} is not “free”
 $(\mathbf{x}^t, \mathbf{h}^{t-1}) \mapsto \mathbf{h}^t$
- 3) Non linearities



Strict Inclusion?

$$\mathcal{H}_{\text{CPRNN}}(R, n) \quad ? \quad \mathcal{H}_{\text{CPRNN}}(R+1, n)$$

Saturation?



$$\mathcal{H}_{\text{CPRNN}}(R, n) \quad ? \quad \mathcal{H}_{\text{CPRNN}}(R+1, n)$$

?

R

Relating Tensor Space to Function Space through Rank

Theorem 1

- $\mathcal{H}_{\text{CPRNN}}(R, n) \subseteq \mathcal{H}_{\text{CPRNN}}(R + 1, n)$ for any R .  Inclusion (easy)
- $\mathcal{H}_{\text{CPRNN}}(R, n) = \mathcal{H}_{\text{CPRNN}}(R + 1, n)$ for any $R \geq R_{\max}$.  Saturation

Relating Tensor Space to Function Space through Rank

Theorem 1

- $\mathcal{H}_{\text{CPRNN}}(R, n) \subseteq \mathcal{H}_{\text{CPRNN}}(R + 1, n)$ for any R .
- $\mathcal{H}_{\text{CPRNN}}(R, n) = \mathcal{H}_{\text{CPRNN}}(R + 1, n)$ for any $R \geq R_{\max}$.

← Inclusion (easy)

← Saturation

Moreover, assuming $n \leq d$:

- $\mathcal{H}_{\text{CPBIRNN}}(R, n) \subsetneq \mathcal{H}_{\text{CPBIRNN}}(R + 1, n)$ for any $R < R_{\text{typ-max}}$ and any real analytic invertible activation function.
- $\mathcal{H}_{\text{CPRNN}}(R, n) \subsetneq \mathcal{H}_{\text{CPRNN}}(R + 1, n)$ for a linear activation function and any $R < R_{\text{typ-max}}$.

} ← Strict Inclusion (Non trivial!)

Relating Tensor Space to Function Space through Rank

Theorem 1

- $\mathcal{H}_{\text{CPRNN}}(R, n) \subseteq \mathcal{H}_{\text{CPRNN}}(R + 1, n)$ for any R .
- $\mathcal{H}_{\text{CPRNN}}(R, n) = \mathcal{H}_{\text{CPRNN}}(R + 1, n)$ for any $R \geq R_{\max}$.

← Inclusion (easy)
← Saturation

Moreover, assuming $n \leq d$:

- $\mathcal{H}_{\text{CPBIRNN}}(R, n) \subsetneq \mathcal{H}_{\text{CPBIRNN}}(R + 1, n)$ for any $R < R_{\text{typ-max}}$ and any real analytic invertible activation function.
- $\mathcal{H}_{\text{CPRNN}}(R, n) \subsetneq \mathcal{H}_{\text{CPRNN}}(R + 1, n)$ for a linear activation function and any $R < R_{\text{typ-max}}$.

← Strict Inclusion (Non trivial!)

Strict Inclusion

Saturation

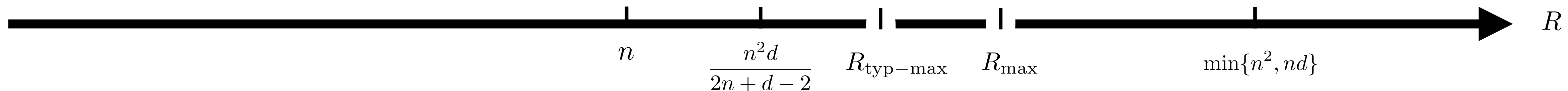
$$\mathcal{H}_{\text{CPBIRNN}}(R, n) \subsetneq \mathcal{H}_{\text{CPBIRNN}}(R + 1, n)$$

$$\mathcal{H}_{\text{CPRNN}}(R, n) \subsetneq \mathcal{H}_{\text{CPRNN}}(R + 1, n)$$

\subsetneq
?

$$\mathcal{H}_{\text{CPBIRNN}}(R, n) = \mathcal{H}_{\text{CPBIRNN}}(R + 1, n)$$

$$\mathcal{H}_{\text{CPRNN}}(R, n) = \mathcal{H}_{\text{CPRNN}}(R + 1, n)$$



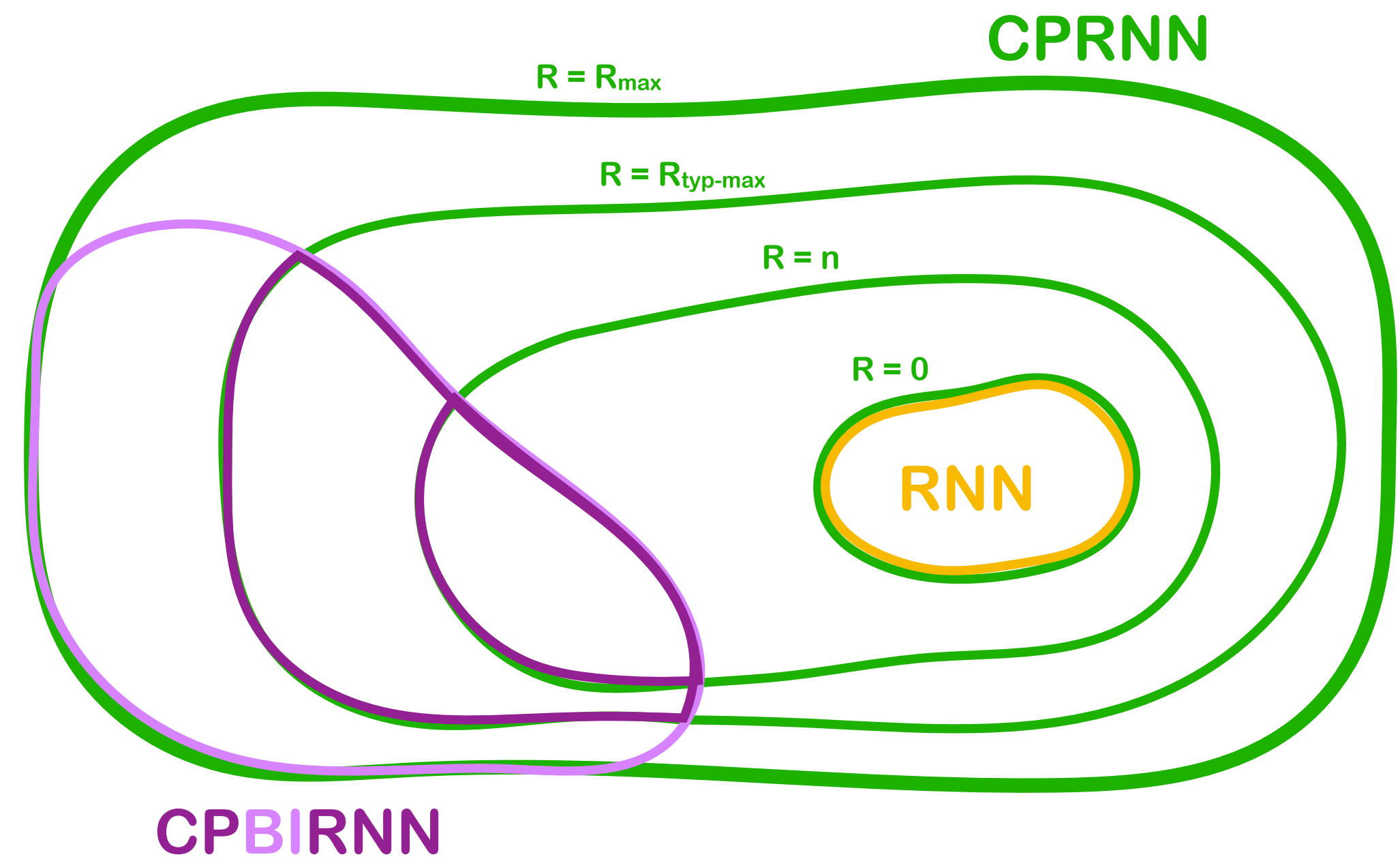
Relating Tensor Space to Function Space through Rank

Theorem 1

- $\mathcal{H}_{\text{CPRNN}}(R, n) \subseteq \mathcal{H}_{\text{CPRNN}}(R + 1, n)$ for any R .
- $\mathcal{H}_{\text{CPRNN}}(R, n) = \mathcal{H}_{\text{CPRNN}}(R + 1, n)$ for any $R \geq R_{\max}$.

Moreover, assuming $n \leq d$:

- $\mathcal{H}_{\text{CPBIRNN}}(R, n) \subsetneq \mathcal{H}_{\text{CPBIRNN}}(R + 1, n)$ for any $R < R_{\text{typ-max}}$ and any real analytic invertible activation function.
- $\mathcal{H}_{\text{CPRNN}}(R, n) \subsetneq \mathcal{H}_{\text{CPRNN}}(R + 1, n)$ for a linear activation function and any $R < R_{\text{typ-max}}$.



Strict Inclusion

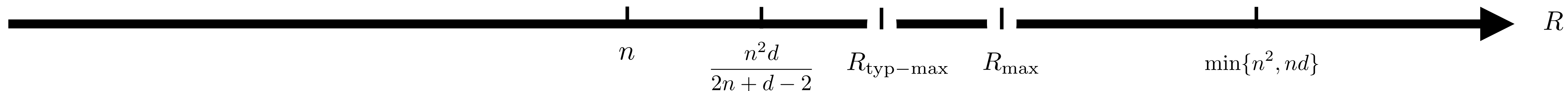
$$\mathcal{H}_{\text{CPBIRNN}}(R, n) \subsetneq \mathcal{H}_{\text{CPBIRNN}}(R + 1, n)$$

$$\mathcal{H}_{\text{CPRNN}}(R, n) \subsetneq \mathcal{H}_{\text{CPRNN}}(R + 1, n)$$

Saturation

$$\mathcal{H}_{\text{CPBIRNN}}(R, n) = \mathcal{H}_{\text{CPBIRNN}}(R + 1, n)$$

$$\mathcal{H}_{\text{CPRNN}}(R, n) = \mathcal{H}_{\text{CPRNN}}(R + 1, n)$$



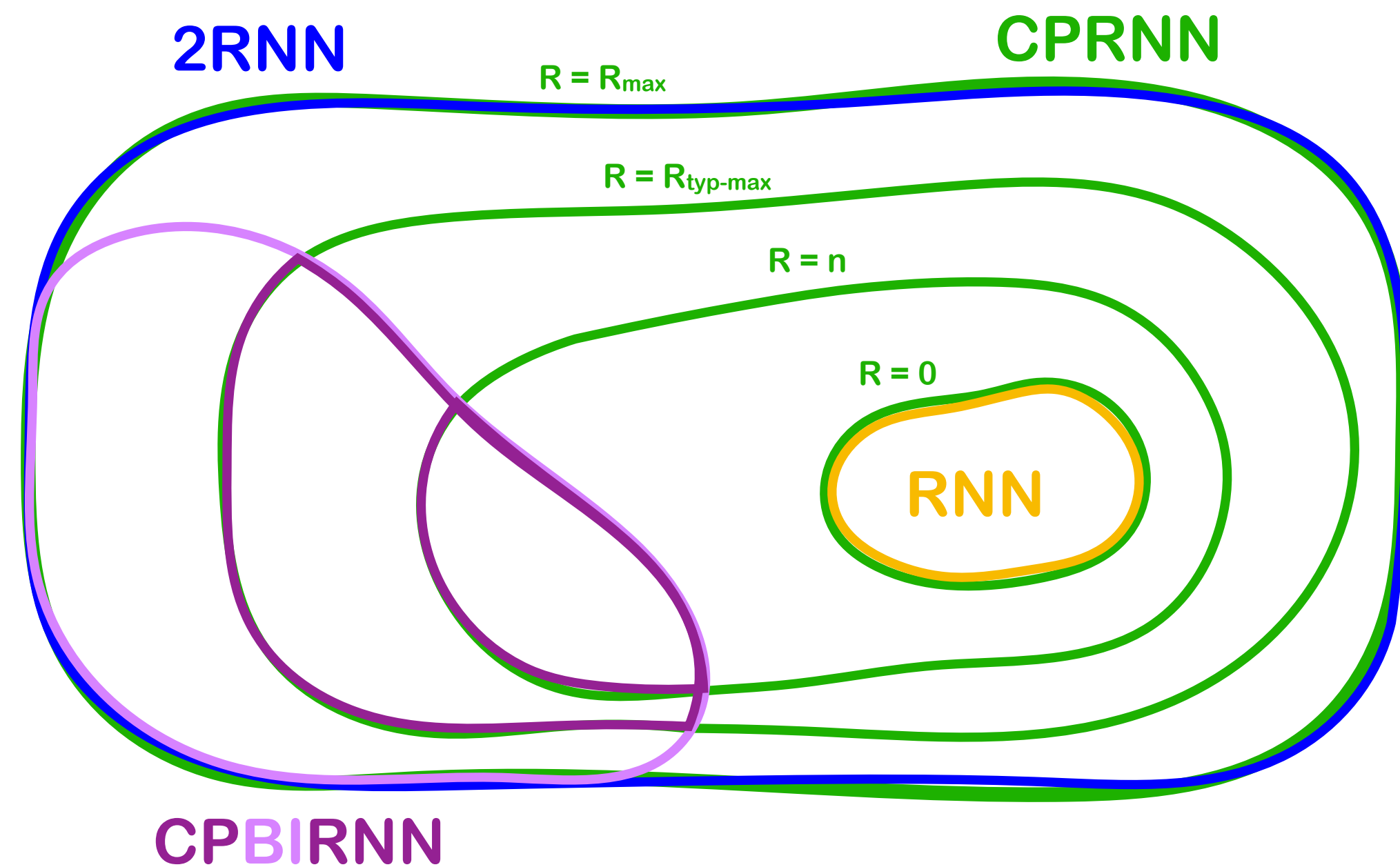
Relating Tensor Space to Function Space through Rank

Corollary 3

- $\mathcal{H}_{\text{CPRNN}}(R, n) = \mathcal{H}_{2\text{RNN}}(n)$ for any $R \geq R_{\max}$ (for any activation function)

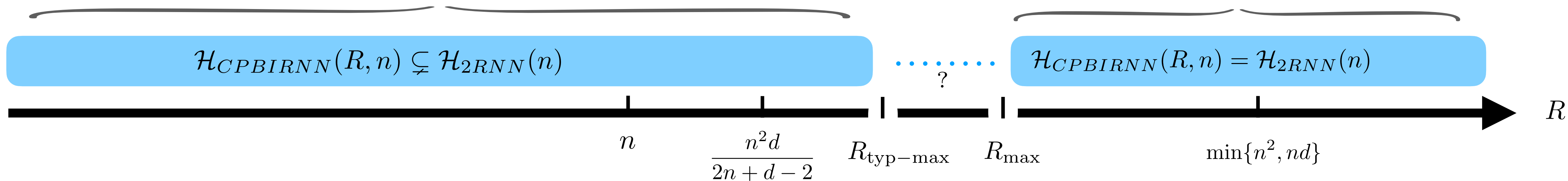
Moreover, assuming $n \leq d$:

- $\mathcal{H}_{\text{CPBIRNN}}(R, n) \subsetneq \mathcal{H}_{2\text{RNN}}(n)$ for any $R < R_{\text{typ-max}}$ and any real analytic invertible activation function.



Strict Inclusion

Saturation

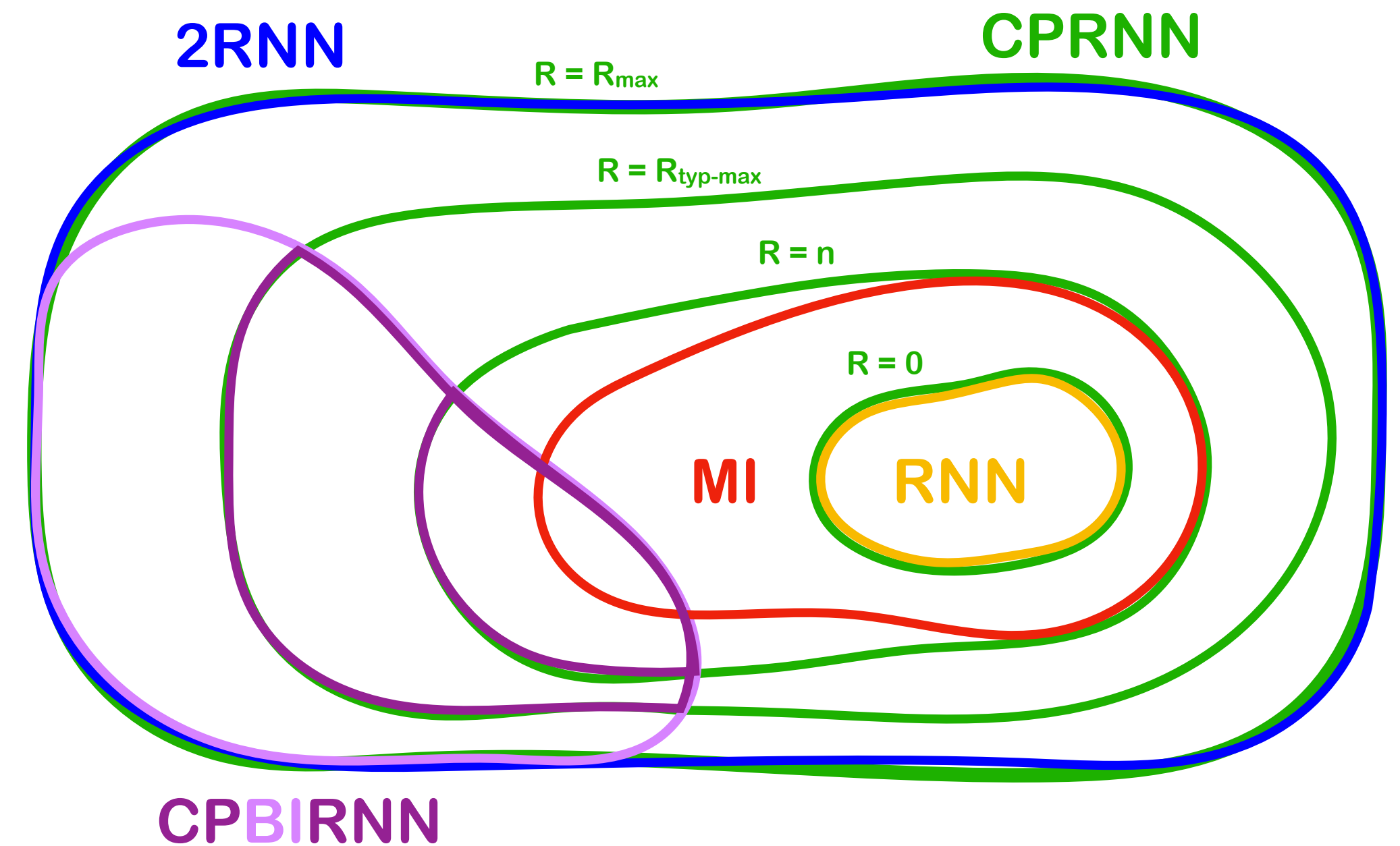


Relating Tensor Space to Function Space through Rank

Corollary 4

Assuming $n \leq d$, for any $R > n$,

- $\mathcal{H}_{\text{MIRNN}}(n) \subseteq \mathcal{H}_{\text{CPRNN}}(R, n)$
- $\mathcal{H}_{\text{MIRNN}}(n) \subsetneq \mathcal{H}_{\text{CPRNN}}(R, n)$ for linear activation function



No inclusion

Strict Inclusion

$$\mathcal{H}_{\text{MIRNN}}(n) \not\subseteq \mathcal{H}_{\text{CPRNN}}(R, n)$$

$$\mathcal{H}_{\text{MIRNN}}(n) \subsetneq \mathcal{H}_{\text{CPRNN}}(R, n)$$

n

$\frac{n^2 d}{2n + d - 2}$

$R_{\text{typ-max}}$

R_{max}

$\min\{n^2, nd\}$

R

Questions

Relations of **inclusions**

$=$ \subsetneq \subseteq

between **sets**

$\mathcal{H}_{\text{CPRNN}}(R, n) \checkmark \mathcal{H}_{\text{CPRNN}}(R + 1, n)$

How does increasing the capacity of the tensor parameter relate to the **expressivity** of the CPRNN? ✓

How does the **point of saturation** in **tensor space** translate in **function space**? ✓

How do the **rank** and **hidden size** **interplay** in controlling the CPRNNs capacity?

Questions

Relations of **inclusions**

$=$ \subsetneq \subseteq

between **sets**

$\mathcal{H}_{\text{CPRNN}}(R, n)$ \checkmark $\mathcal{H}_{\text{CPRNN}}(R + 1, n)$

$\mathcal{H}_{\text{CPRNN}}(R, n)$ $?$ $\mathcal{H}_{\text{CPRNN}}(R, n + 1)$

How does increasing the capacity of the tensor parameter relate to the **expressivity** of the CPRNN? \checkmark

How does the **point of saturation** in **tensor space** translate in **function space**? \checkmark

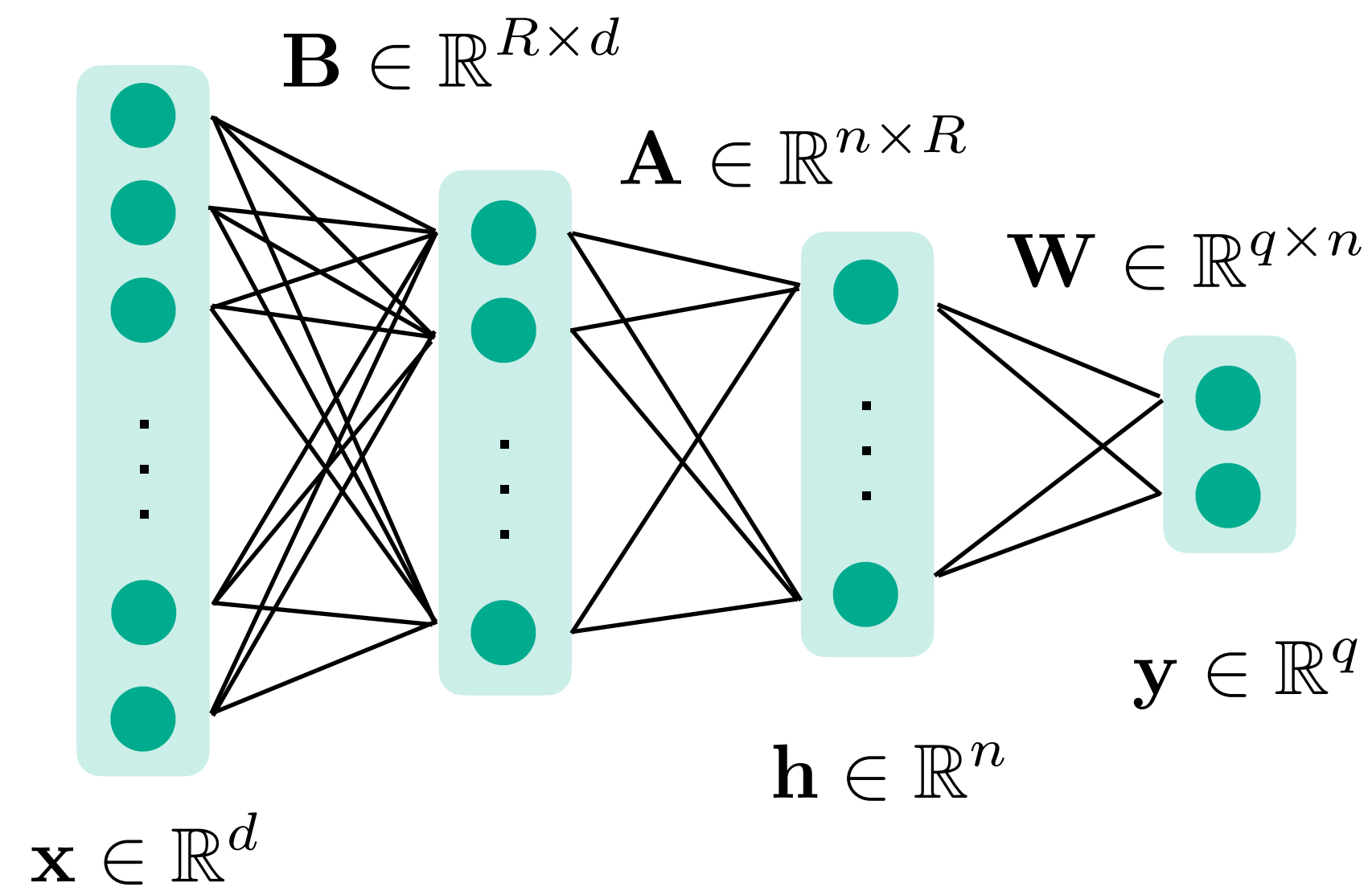
How do the **rank** and **hidden size** **interplay** in controlling the CPRNNs capacity?

Interplay between Rank and Hidden Size

Interplay between Rank and Hidden Size

Intuition : Consider a linear non recurrent model

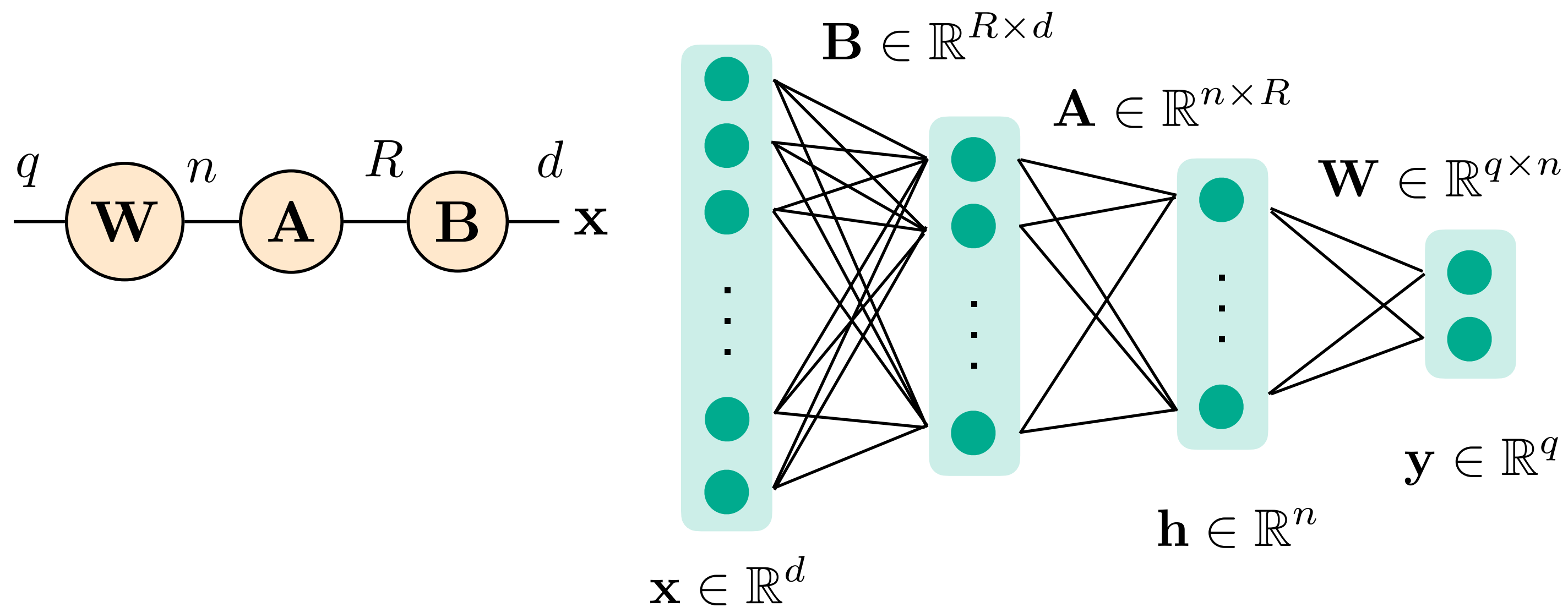
$$\mathbf{y}(\mathbf{x}) = \mathbf{W}(\mathbf{A}\mathbf{B}\mathbf{x})$$



Interplay between Rank and Hidden Size

Intuition : Consider a linear non recurrent model

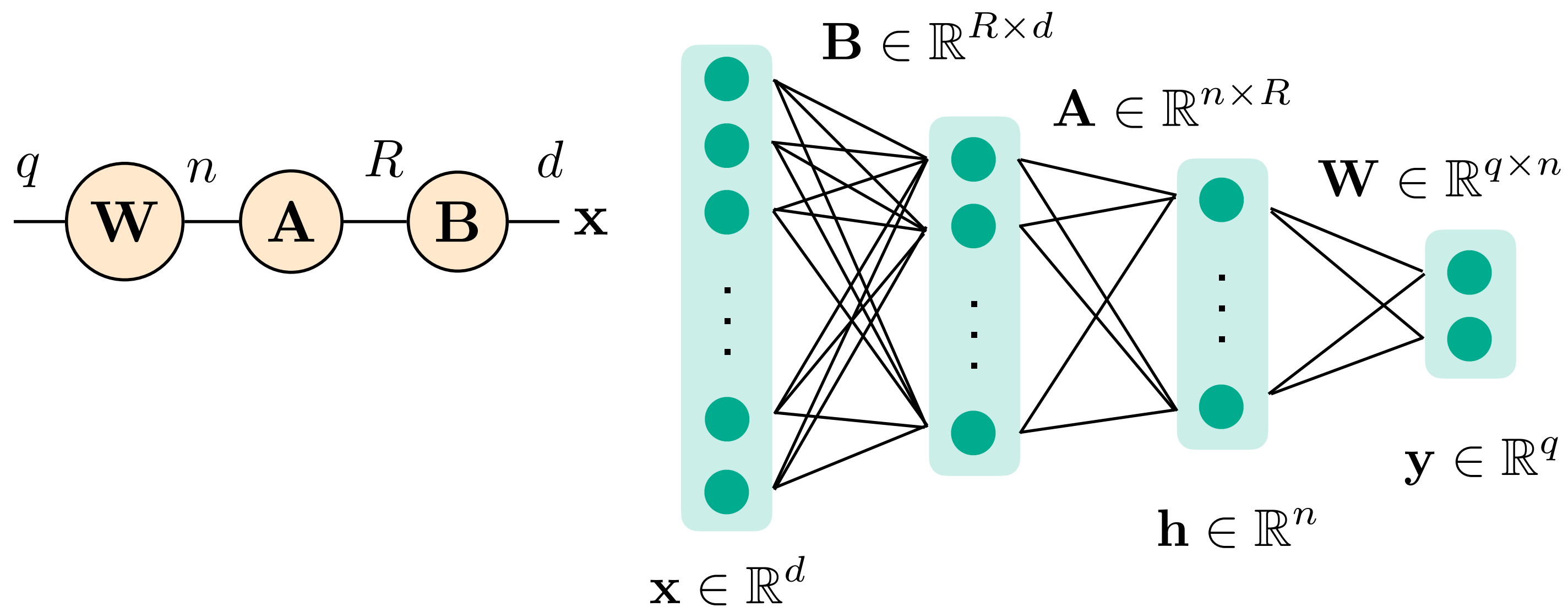
$$\mathbf{y}(\mathbf{x}) = \mathbf{W}(\mathbf{A}\mathbf{B}\mathbf{x})$$



Interplay between Rank and Hidden Size

Intuition : Consider a linear non recurrent model

$$y(\mathbf{x}) = \mathbf{W}(\mathbf{A}\mathbf{B}\mathbf{x}) \quad \longrightarrow \quad \mathcal{L}^{n,q} \circ \mathcal{H}_{AB}(R, n)$$



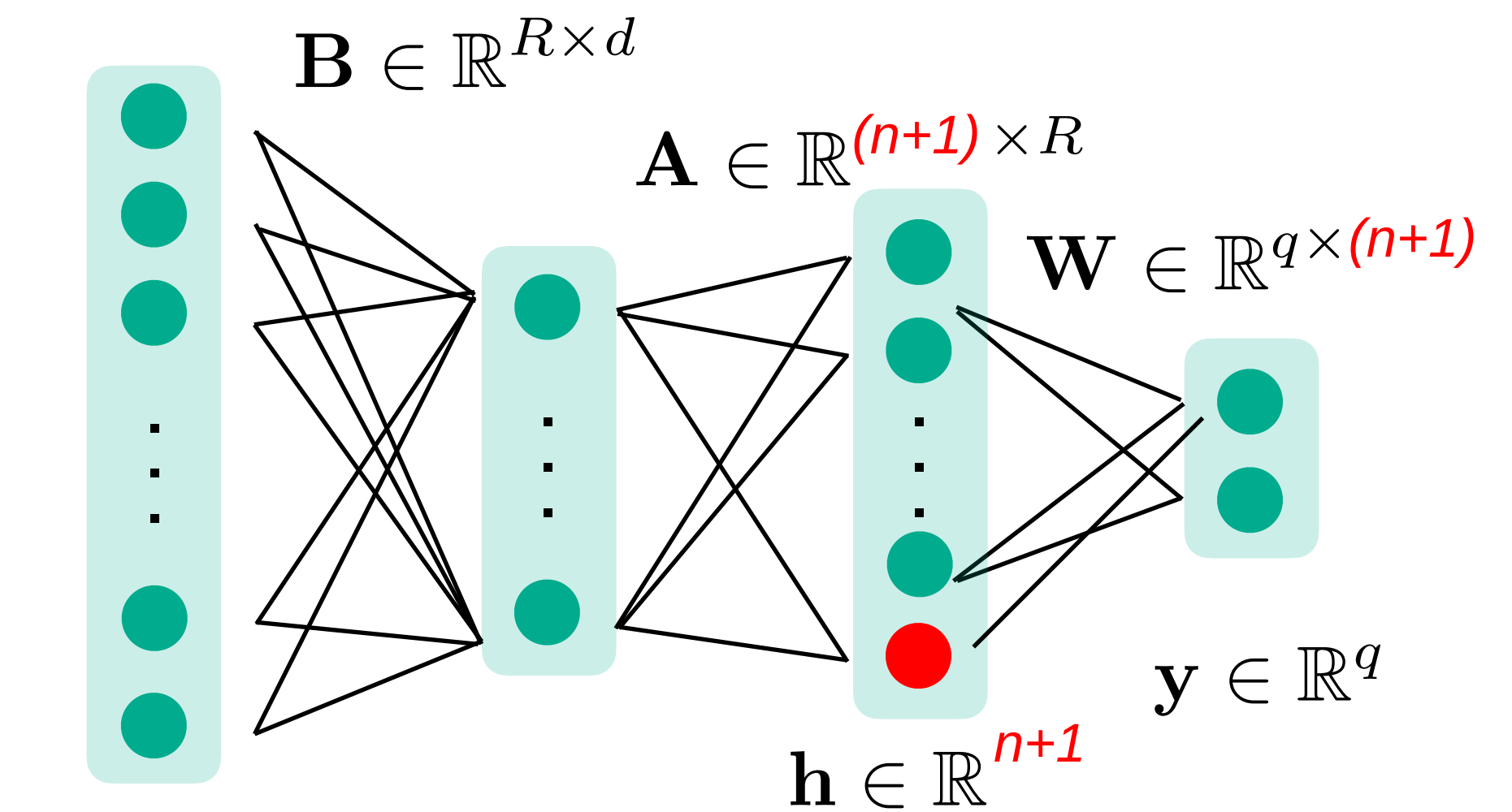
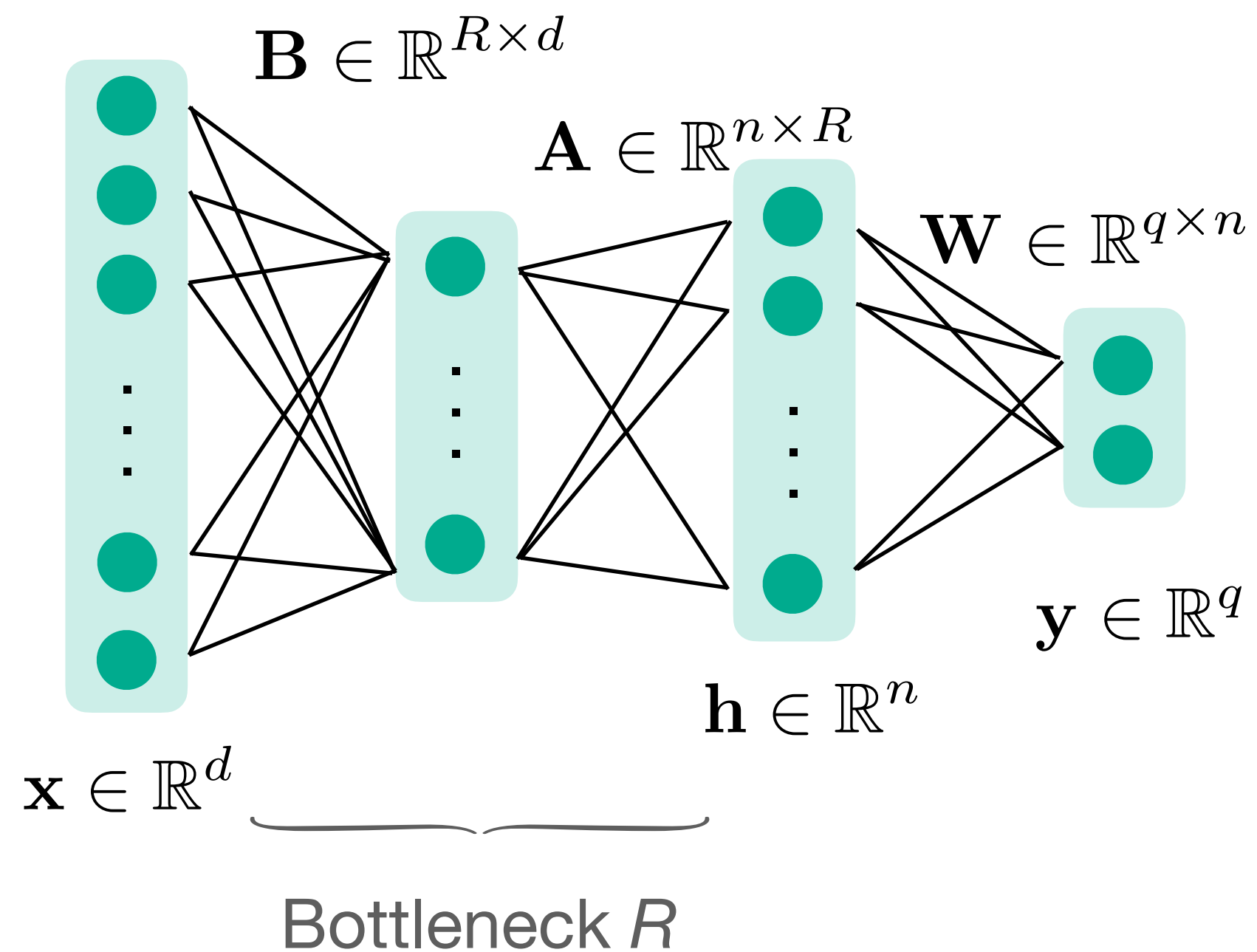
$$\mathcal{L}^{n,q} \circ \mathcal{H}_{AB}(R, n) \quad ? \quad \mathcal{L}^{n+1,q} \circ \mathcal{H}_{AB}(R, n+1)$$

n

R

Interplay between Rank and Hidden Size

Intuition : Consider a linear non recurrent model



$$\mathcal{L}^{n,q} \circ \mathcal{H}_{AB}(R, n) \quad ? \quad \mathcal{L}^{n+1,q} \circ \mathcal{H}_{AB}(R, n+1)$$

n

R

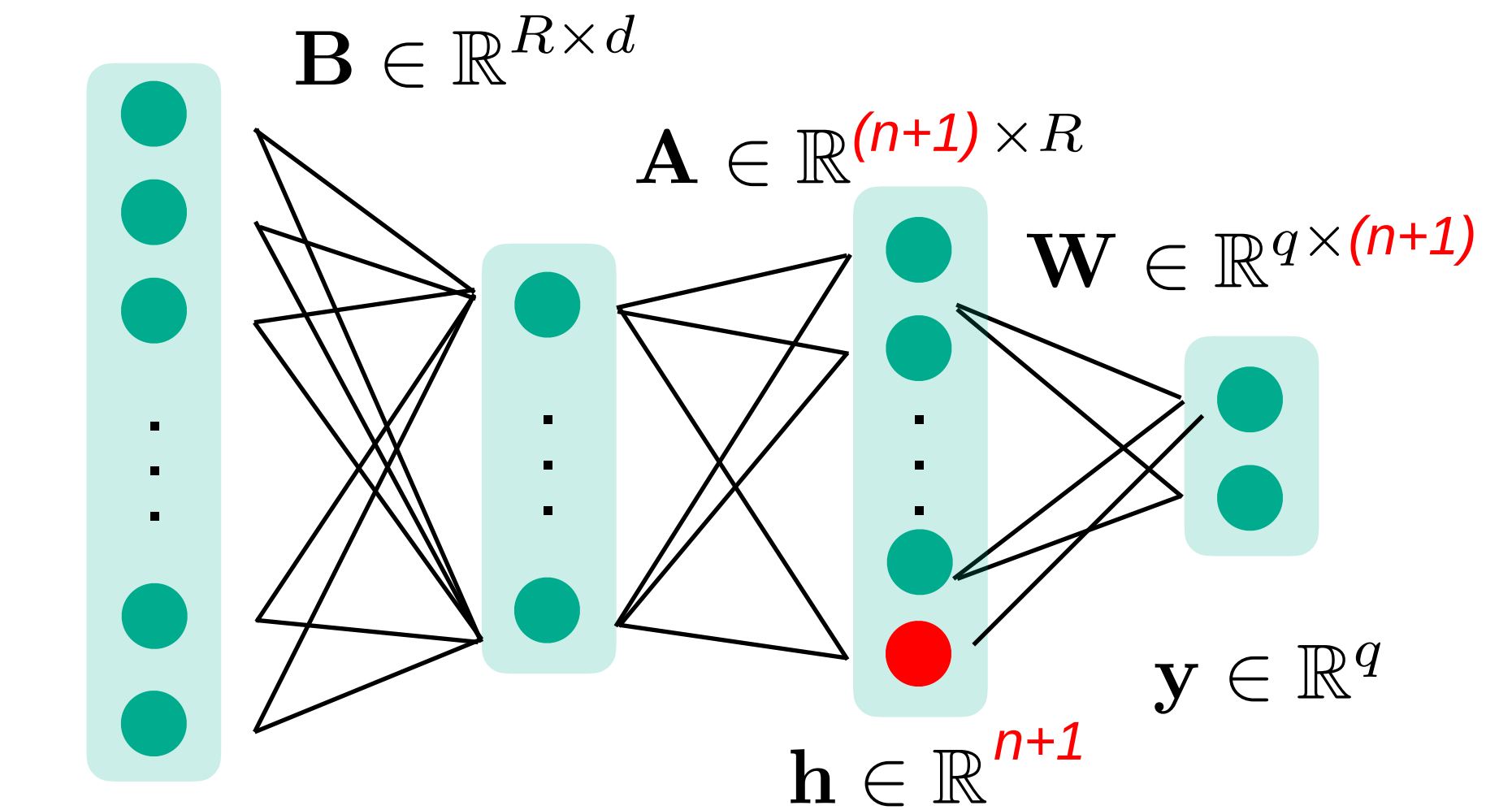
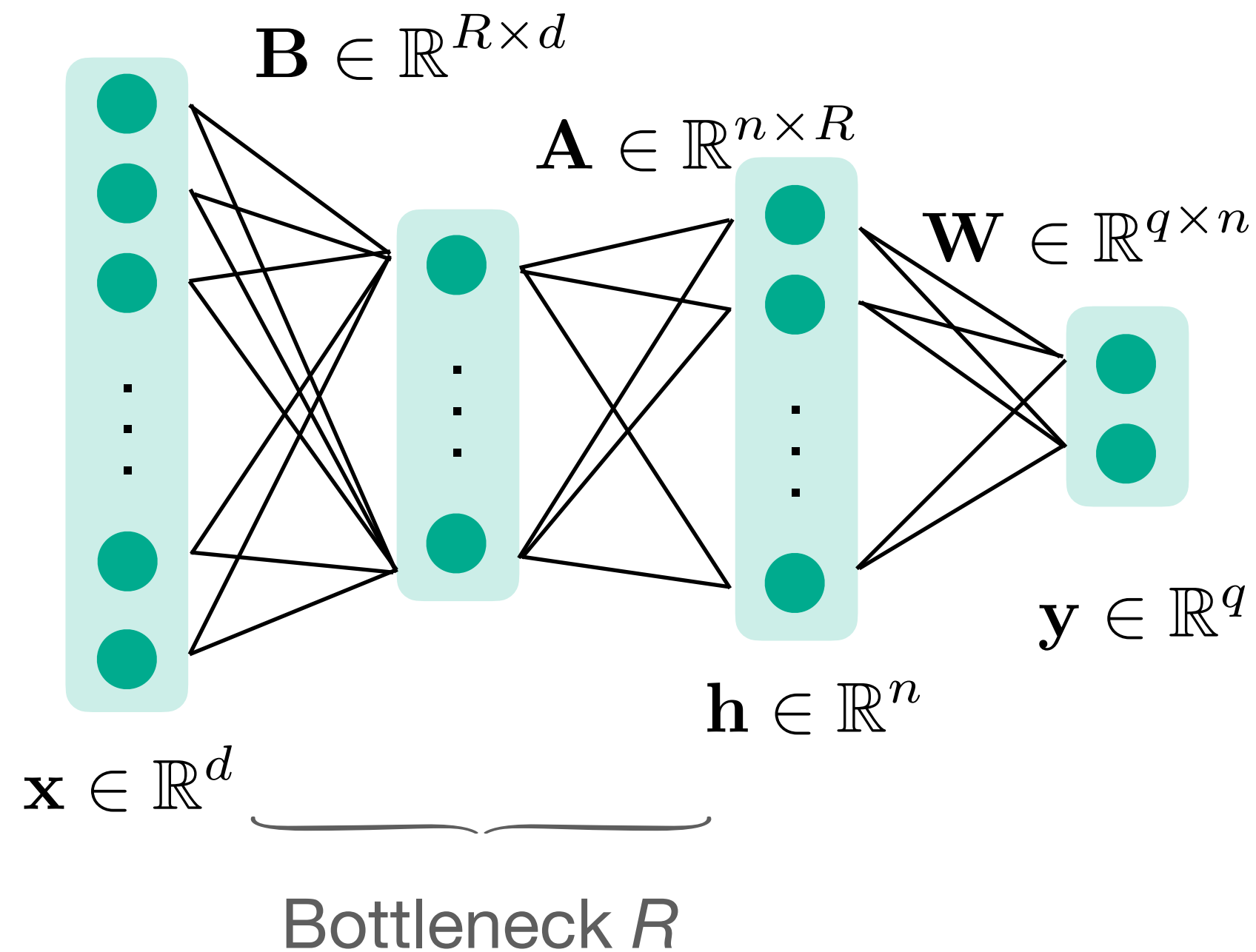
Interplay between Rank and Hidden Size

Intuition : Not trivial (not only for CPRNNs)!

1) If $q < n$

&

2) Non linearities





$$\mathcal{L}^{n,q} \circ \mathcal{H}_{AB}(R, n) \quad ? \quad \mathcal{L}^{n+1,q} \circ \mathcal{H}_{AB}(R, n+1)$$

n

R



Interplay between Rank and Hidden Size

Theorem 2


- $\mathcal{L}^{n,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n) \subseteq \mathcal{L}^{n+1,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n+1)$  Inclusion (easy)
for any R and n .
- $\mathcal{L}^{n,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n) = \mathcal{L}^{n+1,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n+1)$  Bottleneck R
for any $n \geq R$ and linear activation function.

Interplay between Rank and Hidden Size

Theorem 2

- $\mathcal{L}^{n,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n) \subseteq \mathcal{L}^{n+1,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n+1)$  Inclusion (easy)
for any R and n .
- $\mathcal{L}^{n,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n) = \mathcal{L}^{n+1,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n+1)$  Bottleneck R
for any $n \geq R$ and linear activation function.

Moreover, assuming $n \leq d$:

- $\mathcal{L}^{n,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n) \subsetneq \mathcal{L}^{n+1,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n+1)$  Strict Inclusion (Non trivial!)
for any $n < R$ and any invertible activation function
satisfying $\sigma(0) = 0$.

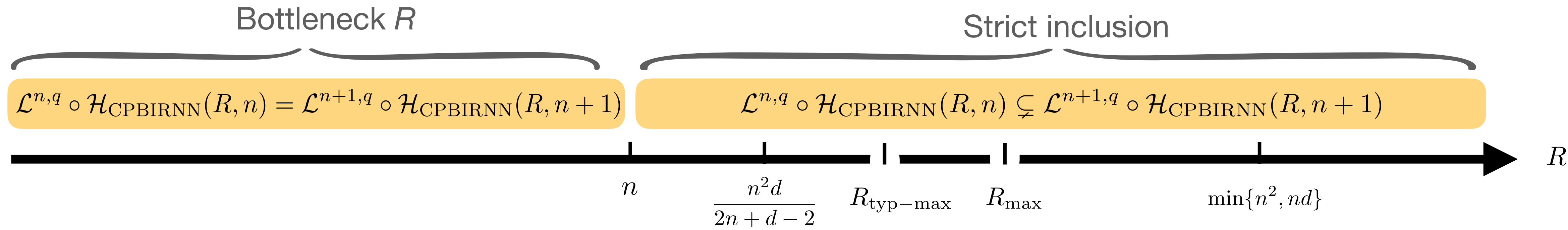
Interplay between Rank and Hidden Size

Theorem 2

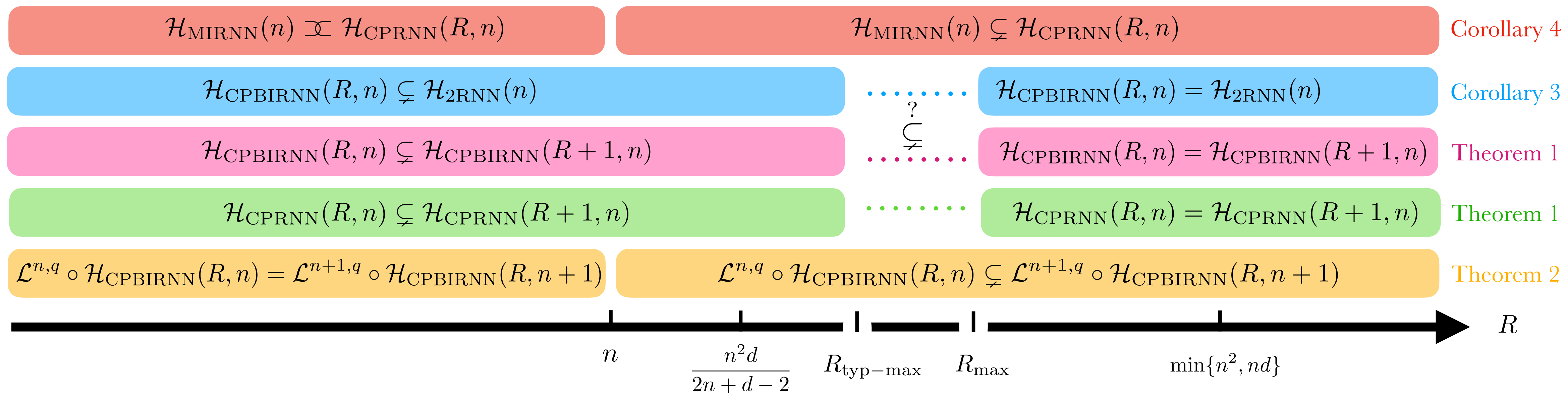
- $\mathcal{L}^{n,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n) \subseteq \mathcal{L}^{n+1,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n+1)$ for any R and n . ← Inclusion (easy)
- $\mathcal{L}^{n,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n) = \mathcal{L}^{n+1,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n+1)$ for any $n \geq R$ and linear activation function. ← Bottleneck R

Moreover, assuming $n \leq d$:

- $\mathcal{L}^{n,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n) \subsetneq \mathcal{L}^{n+1,q} \circ \mathcal{H}_{\text{CPBIRNN}}(R, n+1)$ for any $n < R$ and any invertible activation function satisfying $\sigma(0) = 0$. ← Strict Inclusion (Non trivial!)



Recap



Experiments

How does theory translate to practice?
i.e. Training with Gradient Descent Optimization

Experiments

How does theory translate to practice?
i.e. Training with Gradient Descent Optimization

Task : Language modelling (character level)

Data : Penn Tree bank

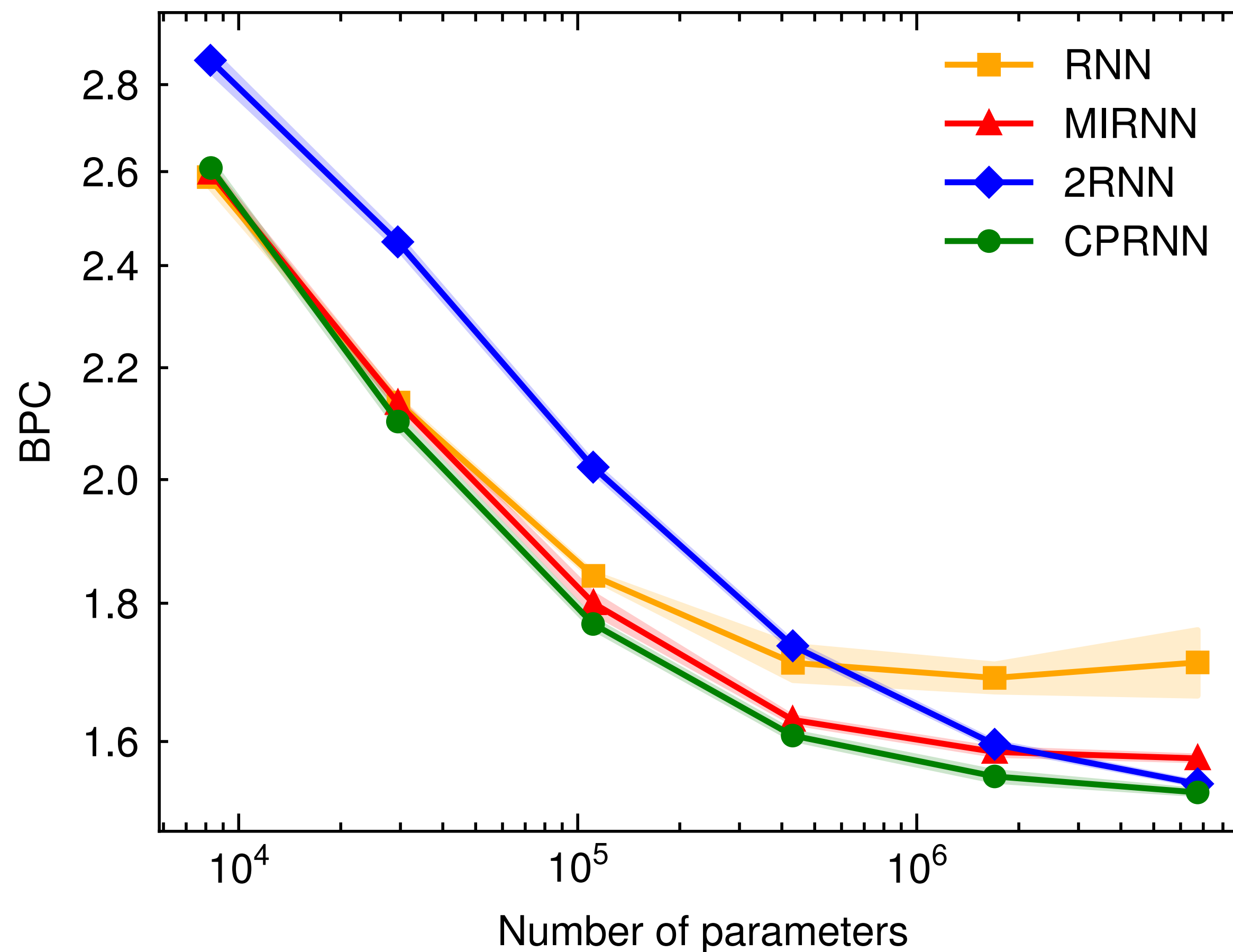
Models : RNN, MIRNN, CPRNN, 2RNN

Metric : Bits Per Character (~Perplexity)

Figure 1 is a line graph showing the relationship between BPC (bits per character) and Hidden Size (log scale) for various RNN architectures. The x-axis represents Hidden Size on a logarithmic scale from 10^1 to 10^3 . The y-axis represents BPC from 1.5 to 3.5. The legend includes: RNN (orange squares), MIRNN (red triangles), CPRNN R=50 (green dotted line with circles), CPRNN R=101 (green dashed line with circles), CPRNN R=350 (green dash-dot line with circles), CPRNN R=1500 (green solid line with circles), and 2RNN (blue diamonds). All architectures show a decreasing trend in BPC as Hidden Size increases, with 2RNN generally achieving the lowest BPC values.

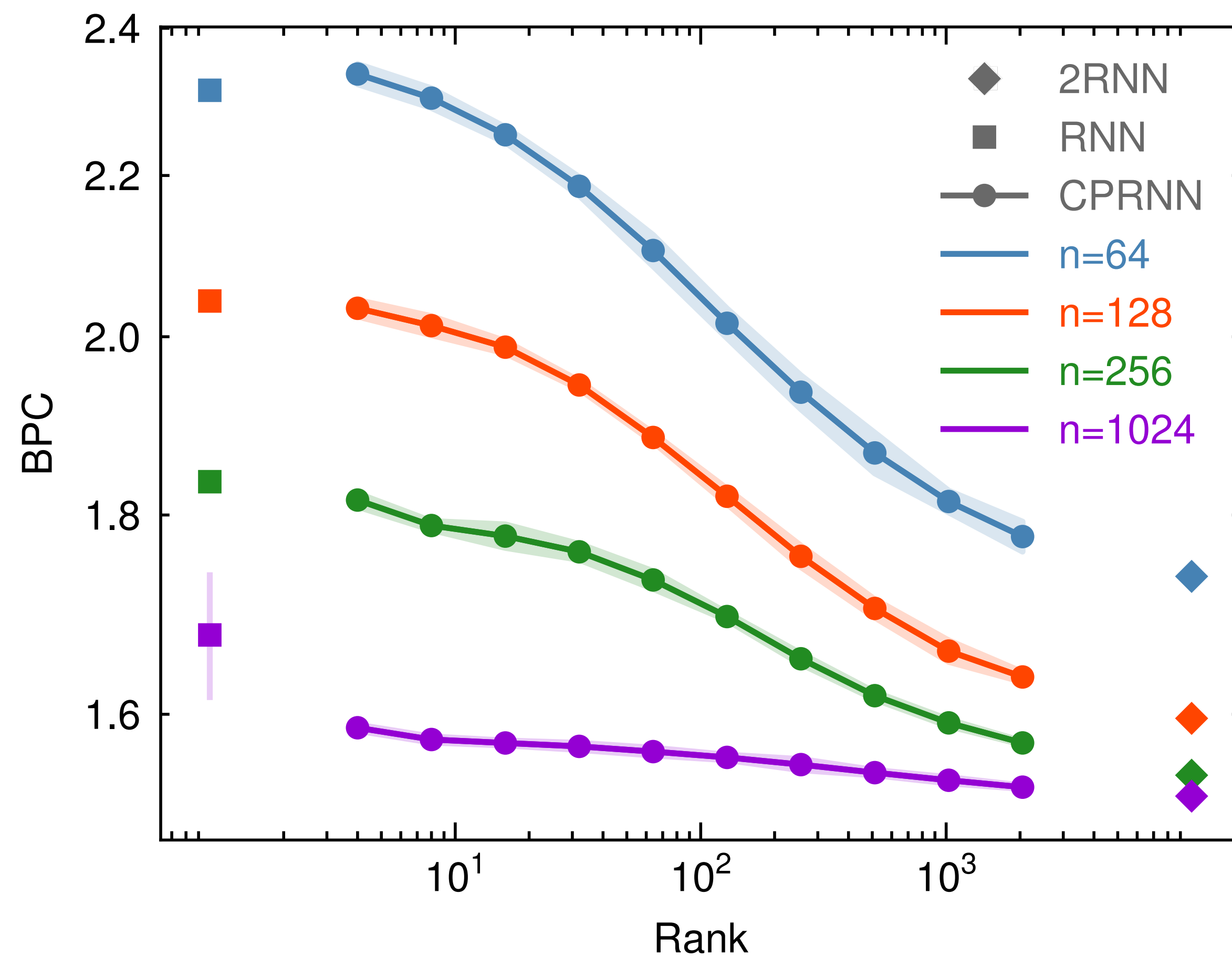
Fixed Size Comparison

There exist values of
rank and hidden size
such that
CPRNN outperforms
RNN, MIRNN and 2RNN



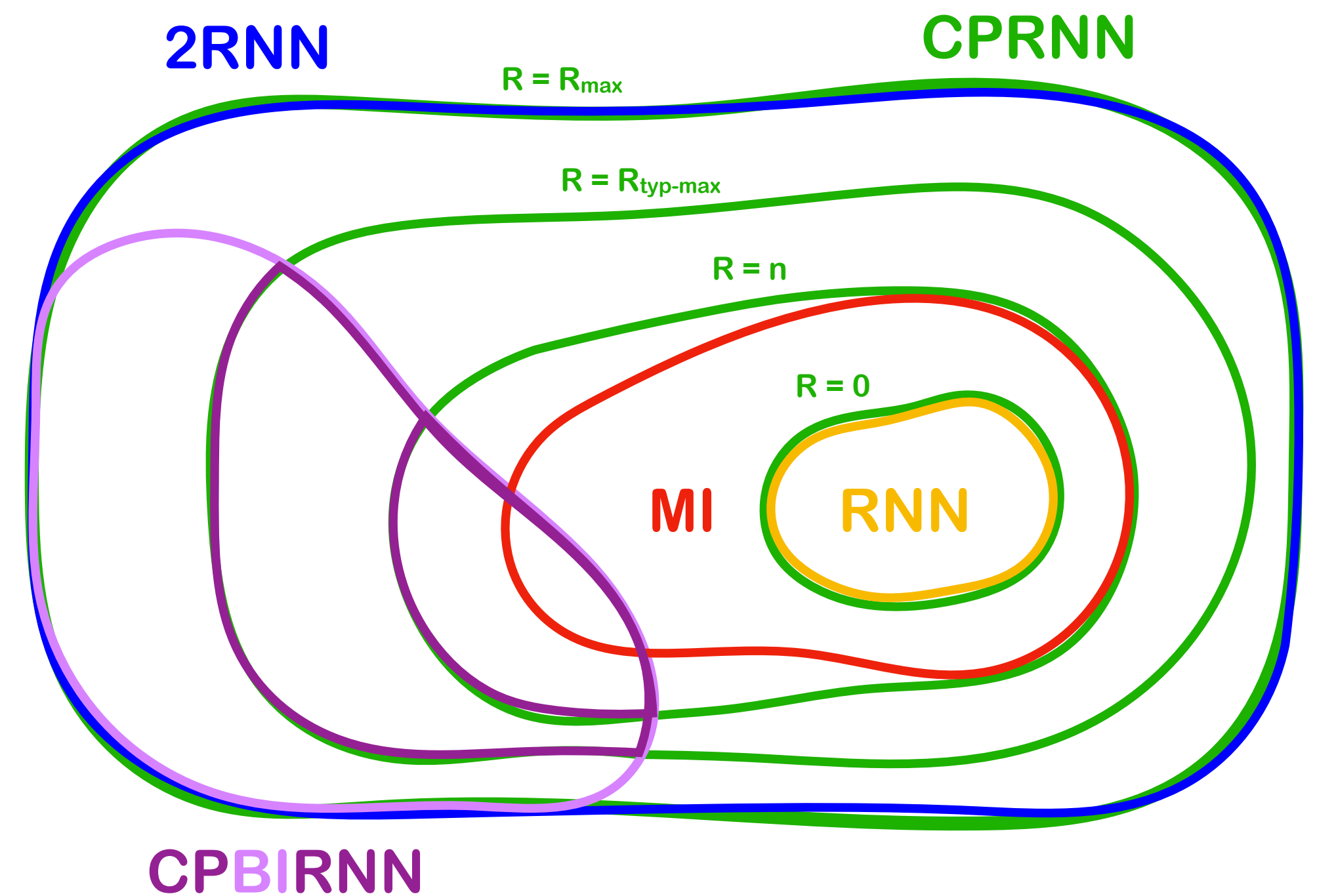
Rank vs Hidden Size

Interpolation
between
RNN and 2RNN
via
CPRNN rank



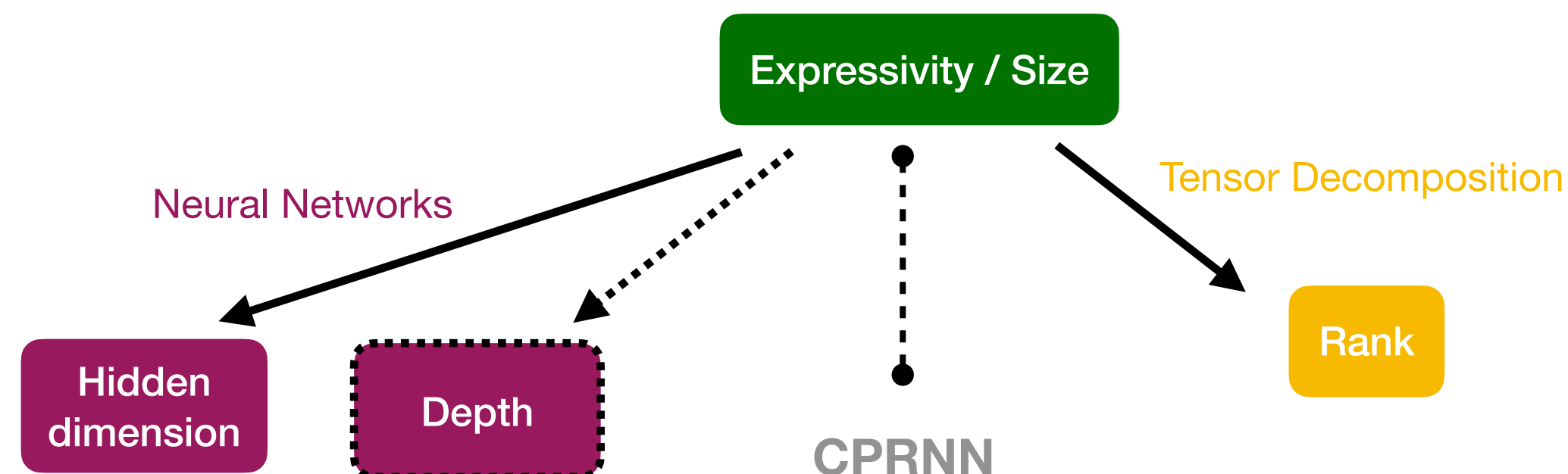
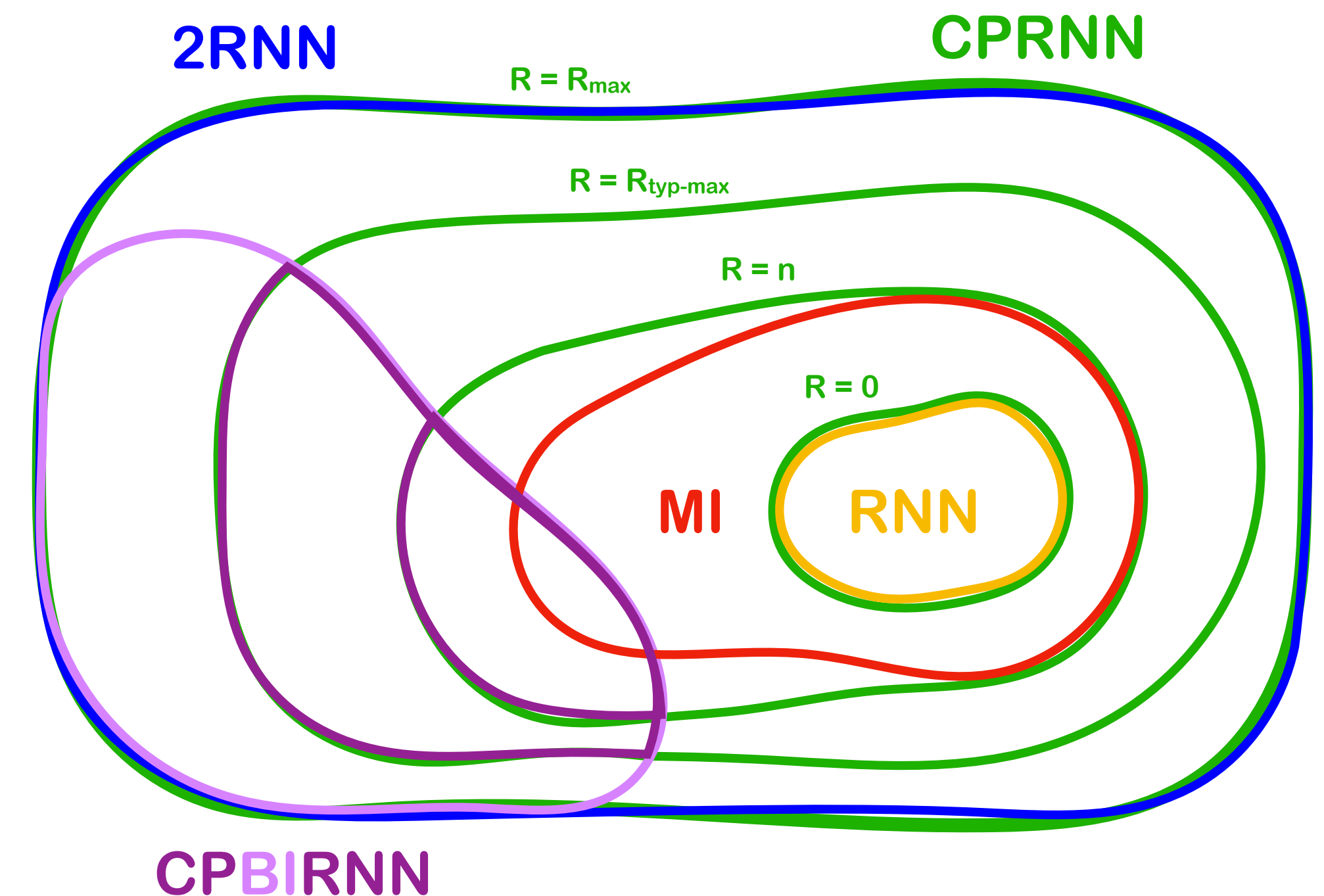
Conclusion

- Tensor Decomposition 🤘
- Formal characterization of Expressivity
 - ▶ Rank: Tuning parameter (up to saturation)
- CPRNN = Parameter efficient alternative to 2RNNs
- CPRNN interpolates RNN and 2RNN
- CPRNN outperforms RNN and 2RNN



Conclusion

- Tensor Decomposition 🙌
- Formal characterization of Expressivity
 - ▶ Rank: Tuning parameter (up to saturation)
- CPRNN = Parameter efficient alternative to 2RNNs
- CPRNN interpolates RNN and 2RNN
- CPRNN outperforms RNN and 2RNN
- Future work: Depth?



Simulating Weighted Automata with Transformers

(Over both sequences and trees!)

Michael Rizvi ¹³ Maude Lizaire ¹³ Clara Lacroce ²³ Guillaume
Rabusseau ¹³

¹Université de Montréal ²McGill University ³Mila

RIKEN-AIP, October 2024

Table of Contents

- 1 Introduction/Motivation
- 2 Transformers
- 3 WFA Refresher
- 4 Theoretical Results for WFAs
- 5 Theoretical Results for WTAs
- 6 Experiments
- 7 Conclusion

Table of Contents

- 1 Introduction/Motivation
- 2 Transformers
- 3 WFA Refresher
- 4 Theoretical Results for WFAs
- 5 Theoretical Results for WTAs
- 6 Experiments
- 7 Conclusion

- **Objective:** show sequential reasoning capacities of Transformer architecture

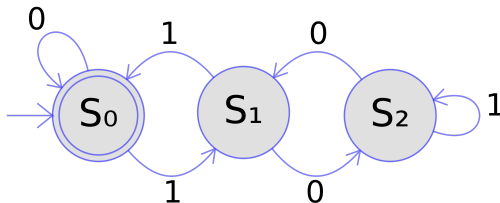
- **Objective:** show sequential reasoning capacities of Transformer architecture
- Result on the **expressivity** of the architecture not **learnability**!

- **Objective:** show sequential reasoning capacities of Transformer architecture
- Result on the **expressivity** of the architecture not **learnability**!
- Take the lense of **simulation**

What do we mean by simulation?

- Simulation = showing steps
- Previous results by Liu et al. introduce this idea for DFA
- For some DFA \mathcal{A} over Σ :
 - Input: $w \in \Sigma^*$
 - Output: sequence of visited states

Example of a DFA for multiples of 3 on $\Sigma = \{0, 1\}$



What do we mean by simulation?

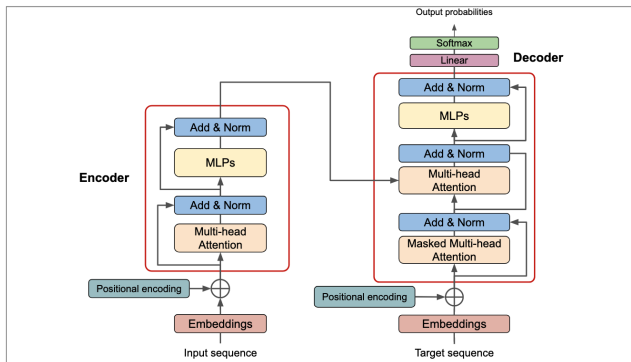
- Liu et al. showed transformers can **simulate DFA** up to length T with $\mathcal{O} \log T$ layers (even $\mathcal{O}(1)$ in some cases!)
- Notion of shortcuts: **shallow** transformers w.r.t. T
- General idea of the theorem
 - Input: a DFA and some sequence length T
 - Output: a transformer which can simulate the inner working of DFA for *any word* of length T
- Can we do this for **more complex models**?

Table of Contents

- 1 Introduction/Motivation
- 2 Transformers**
- 3 WFA Refresher
- 4 Theoretical Results for WFAs
- 5 Theoretical Results for WTAs
- 6 Experiments
- 7 Conclusion

Transformers

Transformer architecture in our construction is similar to the **encoder in the original transformer architecture**.



Transformers

The model is defined as follows

- Input: $\mathbf{X} \in \mathbb{R}^{T \times d}$ where T is sequence length and d is embedding dimension
- Self-attention block:

$$f(\mathbf{X}) = \text{softmax}(\mathbf{X}\mathbf{W}_Q\mathbf{W}_K^\top\mathbf{X}^\top)\mathbf{X}\mathbf{W}_V,$$

- Attention layer f_{attn} : h copies of f , concatenate the outputs
- Feedforward layer f_{mlp} : Simple feedforward MLP

Full L -layer model, with $f_{\text{tf}} : \mathbb{R}^{T \times d} \rightarrow \mathbb{R}^{T \times d}$:

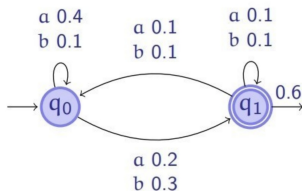
$$f_{\text{tf}} = f_{\text{mlp}}^{(L)} \circ f_{\text{attn}}^{(L)} \circ f_{\text{mlp}}^{(L-1)} \circ f_{\text{attn}}^{(L-1)} \circ \dots \circ f_{\text{mlp}}^{(1)} \circ f_{\text{attn}}^{(1)}.$$

Table of Contents

- 1 Introduction/Motivation
- 2 Transformers
- 3 WFA Refresher**
- 4 Theoretical Results for WFAs
- 5 Theoretical Results for WTAs
- 6 Experiments
- 7 Conclusion

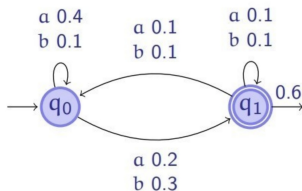
Weighted Finite Automata

- Weighted Finite Automata (WFA) generalize DFAs by **computing a function** over some word w (instead of simply accepting/rejecting)



Weighted Finite Automata

- Weighted Finite Automata (WFA) generalize DFAs by **computing a function** over some word w (instead of simply accepting/rejecting)



⇒ Exactly equivalent to Bi-RNN with linear activation function!

Models	Hidden state	Tensor Network (2nd order terms)
2RNN	$\sigma(\mathcal{A} \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t + \mathbf{V}\mathbf{h}^{t-1} + \mathbf{U}\mathbf{x}^t + \mathbf{b})$	
BIRNN	$\sigma(\mathcal{A} \times_1 \mathbf{h}^{t-1} \times_2 \mathbf{x}^t)$	

Weighted Finite Automata

A *weighted finite automaton* (WFA) of n states over Σ is a tuple

$\mathcal{A} = \langle \alpha, \{\mathbf{A}^\sigma\}_{\sigma \in \Sigma}, \beta \rangle$, where

- $\alpha, \beta \in \mathbb{R}^n$: initial/final weights
- $\mathbf{A}^\sigma \in \mathbb{R}^{n \times n}$: transition matrix for each $\sigma \in \Sigma$

Weighted Finite Automata

A *weighted finite automaton* (WFA) of n states over Σ is a tuple

$\mathcal{A} = \langle \alpha, \{\mathbf{A}^\sigma\}_{\sigma \in \Sigma}, \beta \rangle$, where

- $\alpha, \beta \in \mathbb{R}^n$: initial/final weights
- $\mathbf{A}^\sigma \in \mathbb{R}^{n \times n}$: transition matrix for each $\sigma \in \Sigma$

WFA \mathcal{A} computes a function $f_{\mathcal{A}} : \Sigma^* \rightarrow \mathbb{R}$:

$$f_{\mathcal{A}}(x) = f_{\mathcal{A}}(x_1 \cdots x_t) = \alpha^\top \mathbf{A}^{x_1} \cdots \mathbf{A}^{x_t} \beta = \alpha^\top \mathbf{A}^x \beta$$

Weighted Finite Automata

A *weighted finite automaton* (WFA) of n states over Σ is a tuple

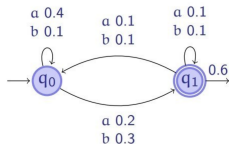
$\mathcal{A} = \langle \alpha, \{\mathbf{A}^\sigma\}_{\sigma \in \Sigma}, \beta \rangle$, where

- $\alpha, \beta \in \mathbb{R}^n$: initial/final weights
- $\mathbf{A}^\sigma \in \mathbb{R}^{n \times n}$: transition matrix for each $\sigma \in \Sigma$

WFA \mathcal{A} computes a function $f_{\mathcal{A}} : \Sigma^* \rightarrow \mathbb{R}$:

$$f_{\mathcal{A}}(x) = f_{\mathcal{A}}(x_1 \cdots x_t) = \alpha^\top \mathbf{A}^{x_1} \cdots \mathbf{A}^{x_t} \beta = \alpha^\top \mathbf{A}^x \beta$$

Example Consider the following WFA with **2** states on $\Sigma = \{a, b\}$



Operator Representation

$$\alpha = \begin{bmatrix} 1.0 \\ 0.0 \end{bmatrix} \quad \mathbf{A}^a = \begin{bmatrix} 0.4 & 0.2 \\ 0.1 & 0.1 \end{bmatrix}$$

$$\omega = \begin{bmatrix} 0.0 \\ 0.6 \end{bmatrix} \quad \mathbf{A}^b = \begin{bmatrix} 0.1 & 0.3 \\ 0.1 & 0.1 \end{bmatrix}$$

$$f(ab) = 0.4 \times 0.3 \times 0.6 + 0.2 \times 0.1 \times 0.6 = 0.084$$

$$= \alpha^\top \mathbf{A}^a \mathbf{A}^b \omega$$

Table of Contents

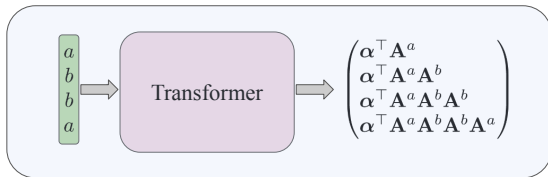
- 1 Introduction/Motivation
- 2 Transformers
- 3 WFA Refresher
- 4 Theoretical Results for WFAs**
- 5 Theoretical Results for WTAs
- 6 Experiments
- 7 Conclusion

Exact Simulation

Given a WFA \mathcal{A} over some alphabet Σ , a function $f : \Sigma^T \rightarrow \mathbb{R}^{T \times n}$ *exactly* simulates \mathcal{A} at length T if, for all $x \in \Sigma^T$ as input, we have $f(x) = \mathcal{A}(x)$, where $\mathcal{A}(x) = (\alpha^\top, \alpha^\top \mathbf{A}^{x_1}, \dots, \alpha^\top \mathbf{A}^{x_{1:T}})^\top$.

Exact Simulation

Given a WFA \mathcal{A} over some alphabet Σ , a function $f : \Sigma^T \rightarrow \mathbb{R}^{T \times n}$ *exactly* simulates \mathcal{A} at length T if, for all $x \in \Sigma^T$ as input, we have $f(x) = \mathcal{A}(x)$, where $\mathcal{A}(x) = (\alpha^\top, \alpha^\top \mathbf{A}^{x_1}, \dots, \alpha^\top \mathbf{A}^{x_{1:T}})^\top$.



Approximate Simulation

Given a WFA \mathcal{A} over some alphabet Σ , a function $f : \Sigma^T \rightarrow \mathbb{R}^{T \times n}$ *approximately* simulates \mathcal{A} at length T with precision $\epsilon > 0$ if for all $x \in \Sigma^T$, we have $\|f(x) - \mathcal{A}(x)\|_F < \epsilon$.

Main Results for WFAs

First Theorem: **exact** simulation:

Theorem 1

Theorem 1 Transformers using bilinear layers in place of an MLP and hard attention can *exactly* simulate all WFAs with n states at length T , with depth $\mathcal{O}(\log T)$, embedding dimension $\mathcal{O}(n^2)$, attention width $\mathcal{O}(n^2)$, MLP width $\mathcal{O}(n^2)$ and $\mathcal{O}(1)$ attention heads.

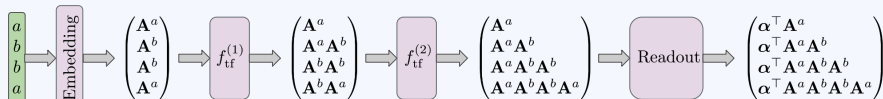
Main Results for WFAs

First Theorem: **exact** simulation:

Theorem 1

Theorem 1 Transformers using bilinear layers in place of an MLP and hard attention can *exactly* simulate all WFAs with n states at length T , with depth $\mathcal{O}(\log T)$, embedding dimension $\mathcal{O}(n^2)$, attention width $\mathcal{O}(n^2)$, MLP width $\mathcal{O}(n^2)$ and $\mathcal{O}(1)$ attention heads.

Transformer Simulation



Main Results for WFAs

Second Theorem: **approximate** simulation

Theorem 2

Transformers can *approximately* simulate all WFAs with n states at length T , up to arbitrary precision $\epsilon > 0$, with depth $\mathcal{O}(\log T)$, embedding dimension $\mathcal{O}(n^2)$, attention width $\mathcal{O}(n^2)$, MLP width $\mathcal{O}(n^4)$ and $\mathcal{O}(1)$ attention heads.

Main Results for WFAs

Second Theorem: **approximate** simulation

Theorem 2

Transformers can *approximately* simulate all WFAs with n states at length T , up to arbitrary precision $\epsilon > 0$, with depth $\mathcal{O}(\log T)$, embedding dimension $\mathcal{O}(n^2)$, attention width $\mathcal{O}(n^2)$, MLP width $\mathcal{O}(n^4)$ and $\mathcal{O}(1)$ attention heads.

Remark

Notice how in Theorem 2, the size of the construction **does not** depend on ϵ !

Main Results for WFAs

Second Theorem: **approximate** simulation

Theorem 2

Transformers can *approximately* simulate all WFAs with n states at length T , up to arbitrary precision $\epsilon > 0$, with depth $\mathcal{O}(\log T)$, embedding dimension $\mathcal{O}(n^2)$, attention width $\mathcal{O}(n^2)$, MLP width $\mathcal{O}(n^4)$ and $\mathcal{O}(1)$ attention heads.

Remark

Notice how in Theorem 2, the size of the construction **does not** depend on ϵ !

Theorem 4. (abridged version of Theorem 3.1 of (Chong, 2020)) Let $d \geq 2$ be an integer, let $f \in \mathcal{P}_{\leq d}(\mathbb{R}^{m_1}, \mathbb{R}^{m_2})$ and let ρ_{Θ}^{σ} be a two-layer MLP with activation function σ and parameters $\Theta = (\mathbf{W}_1, \mathbf{W}_2)$. If $\sigma \in \mathcal{C}(\mathbb{R}) \setminus \mathcal{P}_{\leq d-1}$, then for every $\epsilon > 0$, there exists some $\Theta \in \{(\mathbf{W}_1, \mathbf{W}_2) \mid \mathbf{W}_1 \in \mathbb{R}^{m_1 \times N}, \mathbf{W}_2 \in \mathbb{R}^{N \times m_2}\}$ with $N = \binom{m_1+d}{d}$ such that $\|f - \rho_{\Theta}^{\sigma}\|_{\infty} < \epsilon$.

Weighted Tree Automata

Simulation by a function

Given a WTA $\mathcal{A} = \langle \alpha, \mathcal{T}, \{\mathbf{v}_\sigma\}_{\sigma \in \Sigma} \rangle$ with n states on \mathcal{T}_Σ , we say that a function $f : (\Sigma \cup \{\llbracket, \rrbracket\})^T \rightarrow (\mathbb{R}^n)^T$ simulates \mathcal{A} at length T if for all trees $t \in \mathcal{T}_\Sigma$ such that $|\text{str}(t)| \leq T$, $f(\text{str}(t))_i = \mu(\tau_i)$ for all $i \in \mathcal{I}_t$.

Weighted Tree Automata

Simulation by a function

Given a WTA $\mathcal{A} = \langle \alpha, \mathcal{T}, \{\mathbf{v}_\sigma\}_{\sigma \in \Sigma} \rangle$ with n states on \mathcal{T}_Σ , we say that a function $f : (\Sigma \cup \{\llbracket, \rrbracket\})^T \rightarrow (\mathbb{R}^n)^T$ simulates \mathcal{A} at length T if for all trees $t \in \mathcal{T}_\Sigma$ such that $|\text{str}(t)| \leq T$, $f(\text{str}(t))_i = \mu(\tau_i)$ for all $i \in \mathcal{I}_t$.

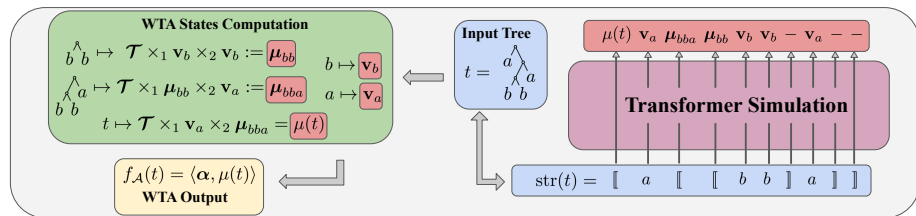


Table of Contents

- 1 Introduction/Motivation
- 2 Transformers
- 3 WFA Refresher
- 4 Theoretical Results for WFAs
- 5 Theoretical Results for WTAs**
- 6 Experiments
- 7 Conclusion

Main Results for WTAs

Theorem 3

Transformers can *approximately* simulate all WTAs \mathcal{A} with n states at length T , up to arbitrary precision $\epsilon > 0$, with embedding dimension $\mathcal{O}(n)$, attention width $\mathcal{O}(n)$, MLP width $\mathcal{O}(n^3)$ and $\mathcal{O}(1)$ attention heads.

Moreover:

- Simulation over arbitrary trees can be done with depth $\mathcal{O}(T)$
- Simulation over balanced trees (trees whose depth is of order $\log(T)$) with depth $\mathcal{O}(\log(T))$.

Main Results for WTAs

Theorem 3

Transformers can *approximately* simulate all WTAs \mathcal{A} with n states at length T , up to arbitrary precision $\epsilon > 0$, with embedding dimension $\mathcal{O}(n)$, attention width $\mathcal{O}(n)$, MLP width $\mathcal{O}(n^3)$ and $\mathcal{O}(1)$ attention heads.

Moreover:

- Simulation over arbitrary trees can be done with depth $\mathcal{O}(T)$
- Simulation over balanced trees (trees whose depth is of order $\log(T)$) with depth $\mathcal{O}(\log(T))$.

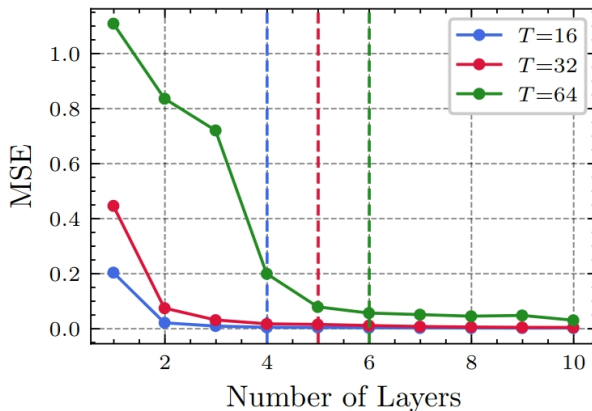
Remark

In the worst case, the tree is completely unbalanced in which case we recover the sequential WFA case!

Table of Contents

- 1 Introduction/Motivation
- 2 Transformers
- 3 WFA Refresher
- 4 Theoretical Results for WFAs
- 5 Theoretical Results for WTAs
- 6 Experiments**
- 7 Conclusion

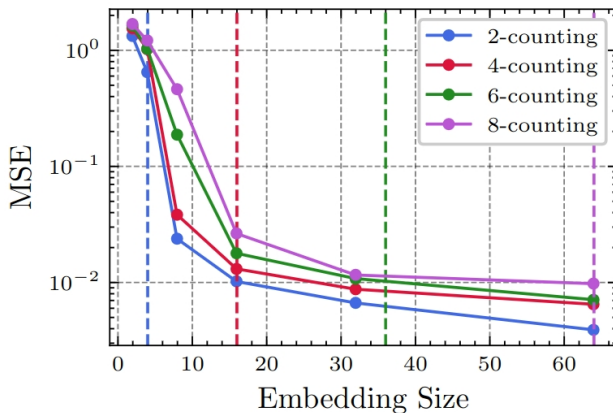
Depth vs. Length



target: 2 states WFA counting 0's in binary strings

theory: $\log T$ layers for sequences of length T

Width vs. #States



target: k states WFAs counting k symbols in a string
theory: n^2 width to simulate WFA with n states

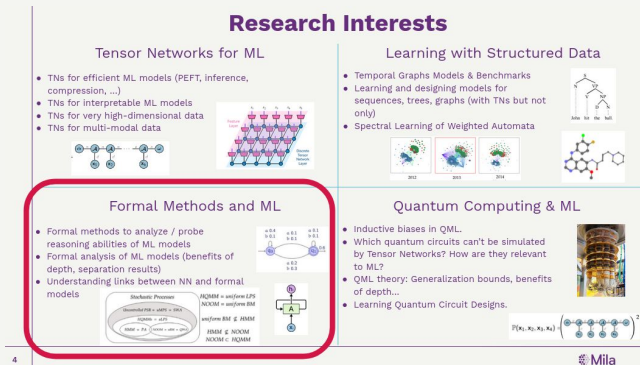
Table of Contents

- 1 Introduction/Motivation
- 2 Transformers
- 3 WFA Refresher
- 4 Theoretical Results for WFAs
- 5 Theoretical Results for WTAs
- 6 Experiments
- 7 Conclusion**

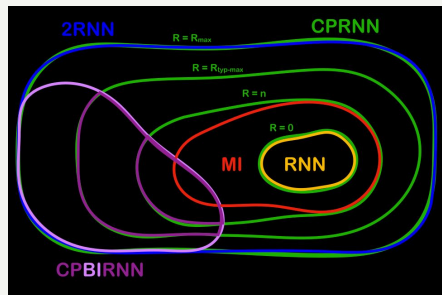
Conclusion

- We define simulation of **weighted automata** for **sequences and trees**
- We derive the notion of **approximate simulation** and how it applies to transformers
- We show that transformers can **simulate WFAs with $\mathcal{O}(\log T)$ layers**
- We show transformers can **simulate WTAs with $\mathcal{O}(\log T)$ layers**
- Our results extend the ones of Liu et al. for DFAs in **two directions**: from **boolean to real weights** and from **sequences to trees**

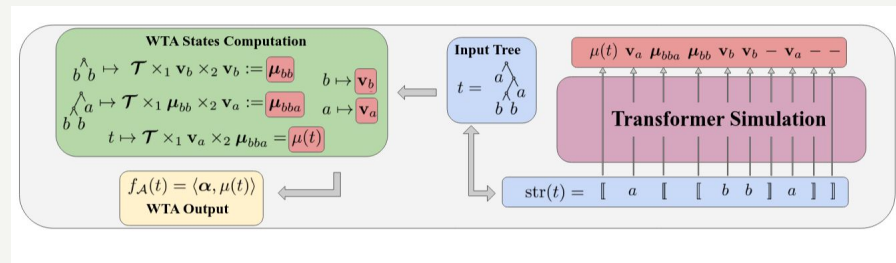
Conclusion



A Tensor Decomposition Perspective on 2nd Order RNNs



Simulating Weighted Automata with Transformers



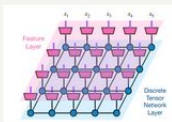
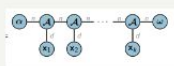
Conclusion

- I am here for 2 more weeks and happy to chat!
- I am interested in many topics, including (**but not limited to**):

Research Interests

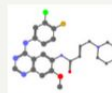
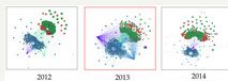
Tensor Networks for ML

- TNs for efficient ML models (PEFT, inference, compression, ...)
- TNs for interpretable ML models
- TNs for very high-dimensional data
- TNs for multi-modal data



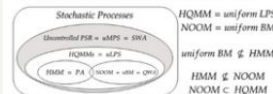
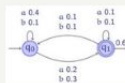
Learning with Structured Data

- Temporal Graphs Models & Benchmarks
- Learning and designing models for sequences, trees, graphs (with TNs but not only)
- Spectral Learning of Weighted Automata



Formal Methods and ML

- Formal methods to analyze / probe reasoning abilities of ML models
- Formal analysis of ML models (benefits of depth, separation results)
- Understanding links between NN and formal models



HQMM = uniform LPS
NOOM = uniform BM
uniform BM $\not\subseteq$ HMM
HMM $\not\subseteq$ NOOM
NOOM \subseteq HQMM



Quantum Computing & ML

- Inductive biases in QML.
- Which quantum circuits can't be simulated by Tensor Networks? How are they relevant to ML?
- QML theory: Generalization bounds, benefits of depth...
- Learning Quantum Circuit Designs.



$$\mathbb{P}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4) = \left(\text{Tensor Network Diagram} \right)^2$$

If you see me at my desk,
feel free to drop by (I like
random math questions)!

Conclusion

- I am here for 2 more weeks and happy to chat!
- I am interested in many topics, including (**but not limited to**):

Research Interests

Thank you for listening!

Questions?

If you see me at my desk,
feel free to drop by (I like
random math questions)!