Unsupervised Learning Through Prediction in a Model of Cortex

Santosh Vempala, Christos Papadimitriou

Robi Bhattacharjee
Outline:

1. Explain how PJOIN works
2. Discuss identifying patterns
3. Algorithm for identifying patterns
4. Proof
PJOIN

C = PJOIN(A,B)
PJOIN

\[ C = \text{PJOIN}(A,B) \]

- C fires when A, B fire (not nec simultaneous)
PJOIN

C = PJOIN(A, B)

- C fires when A, B fire (not nec simultaneous)
- C can "predict" A, B
PJOIN
PJOIN

1. B fires
PJOIN

1. B fires

2. C predicts A
PJOIN

1. B fires

2. C predicts A
   a. fires to A
PJOIN

1. B fires

2. C predicts A
   a. fires to A
   b. state to P(A)
PJOIN

1. B fires

2. C predicts A
   a. fires to A
   b. state to P(A)

3. A fires
PJOIN

1. B *fires*

2. C predicts A
   a. *fires* to A
   b. state to P(A)

3. A *fires*

4. C *fires*
PJOIN
PJOIN

1. C is predicted
PJOIN

1. C is predicted

2. C predicts A, B
PJOIN

1. C is predicted
2. C predicts A, B
3. B fires
PJOIN

1. C is predicted
2. C predicts A, B
3. B fires
4. A fires
PJOIN

1. C is predicted
2. C predicts A, B
3. B fires
4. A fires
5. C fires
Cascade Example
Cascade Example
Cascade Example
Cascade Example
Cascade Example
Cascade Example
Cascade Example
Cascade Example
Cascade Example
Cascade Example
Cascade Example
Patterns
Patterns

A pattern is an $x_1x_2x_3 \ldots x_n \in \{0, 1\}^n$
Patterns

A pattern is an \( x_1 x_2 x_3 \ldots x_n \in \{0, 1\}^n \)

Each \( x_i \) is stored in a sensory item \( S_1 \)
Patterns

A pattern is an $x_1x_2x_3\ldots x_n \in \{0, 1\}^n$

Each $x_i$ is stored in a sensory item $S_i$
Patterns

A pattern is an \( x_1 x_2 x_3 \ldots x_n \in \{0, 1\}^n \)

Each \( x_i \) is stored in a sensory item \( S_1 \)

\( S_i \) expresses its state through items \( 0_i, 1_i \)

\[ S_1 \quad S_2 \quad S_3 \quad \ldots \quad S_n \]
Patterns

A pattern is an $x_1x_2x_3 \ldots x_n \in \{0, 1\}^n$

Each $x_i$ is stored in a sensory item $S_i$

$S_i$ expresses its state through items $0_i, 1_i$
Sensory Item Details
1. $S_1$ fires to
   • $0_1$ if state = 0
   • $1_1$ if state = 1
Sensory Item Details

1. $S_1$ fires to
   - 0₁ if state = 0
   - 1₁ if state = 1
1. \( S_1 \) fires to
   - 0, if state = 0
   - 1, if state = 1
   - 0 or 1 fires
1. $S_1$ fires to
   - $0_1$ if state = 0
   - $1_1$ if state = 1
   - $0_1$ or $1_1$ fires
Sensory Item Details

1. $S_1$ fires to
   - $0_1$ if state = 0
   - $1_1$ if state = 1
   - $0_1$ or $1_1$ fires

2. $0_1$ or $1_1$ predicts
Sensory Item Details

1. $S_1$ fires to
   - $0_1$ if state = 0
   - $1_1$ if state = 1
   - $0_1$ or $1_1$ fires

2. $0_1$ or $1_1$ predicts
Sensory Item Details

1. $S_1$ fires to
   - $0_1$ if state = 0
   - $1_1$ if state = 1
   - $0_1$ or $1_1$ fires

2. $0_1$ or $1_1$ predicts
   - $S_1$ fires
Sensory Item Details

1. $S_1$ fires to
   - $0_1$ if state = 0
   - $1_1$ if state = 1
   - $0_1$ or $1_1$ fires

2. $0_1$ or $1_1$ predicts
   - $S_1$ fires
Sensory Item Details

1. $S_1$ fires to
   - $0_1$ if state = 0
   - $1_1$ if state = 1
   - $0_1$ or $1_1$ fires

2. $0_1$ or $1_1$ predicts
   - $S_1$ fires
Intuition about Patterns

Key: items in pattern do not fire simultaneously
Intuition about Patterns

Key: items in pattern do not fire simultaneously
Intuition about Patterns

Key: items in pattern do not fire simultaneously
Intuition about Patterns

Key: items in pattern do not fire simultaneously
Intuition about Patterns

Key: items in pattern do not fire simultaneously
Intuition about Patterns

Key: items in pattern do not fire simultaneously
Intuition about Patterns

Key: items in pattern do not fire simultaneously
Intuition about Patterns

Key: items in pattern do not fire simultaneously

Goal: item $I_p$ fires w.h.p if $p$ is stored in $S_i$
Intuition about Patterns

Key: items in pattern do not fire simultaneously

Goal: item $I_p$ fires w.h.p if $p$ is stored in $S_i$

$I_p$ is the PJOIN of the appropriate $0/1_i$

0_1 1_1 0_2 1_2 0_3 1_3 ... 0_n 1_n

S_1 S_2 S_3 ... S_n
Presenting a Pattern
Presenting a Pattern

1. Each sensory item $S_i$ is appropriately set
Presenting a Pattern

1. Each sensory item $S_i$ is appropriately set.

2. For $T$ steps, $S_i$ fire independently with probability $p$. 
Presenting a Pattern

1. Each sensory item $S_i$ is appropriately set

2. For T steps, $S_i$ fire independently with probability $p$
   a. Each $S_i$ fires at most once
Presenting a Pattern

1. Each sensory item $S_i$ is appropriately set

2. For $T$ steps, $S_i$ fire independently with probability $p$
   a. Each $S_i$ fires at most once
   b. Including in response to prediction from $0/1_i$
Algorithm Specifications
Algorithm Specifications

Input: m patterns that are presented in an arbitrary order
Algorithm Specifications

Input: m patterns that are presented in an arbitrary order

Output: m items $I_1, I_2, ..., I_m$
Algorithm Specifications

Input: m patterns that are presented in an arbitrary order

Output: m items $I_1, I_2, \ldots, I_m$
  - Each $I_i$ uniquely represents the appropriate pattern
Algorithm Specifications

Input: \( m \) patterns that are presented in an arbitrary order

Output: \( m \) items \( I_1, I_2, \ldots, I_m \)
- Each \( I_i \) uniquely represents the appropriate pattern
- After a bounded point, no more items are created
The Algorithm
The Algorithm

Item Eligibility:
The Algorithm

Item Eligibility:

- An item becomes eligible $D$ steps after it fires
The Algorithm

Item Eligibility:
- An item becomes eligible $D$ steps after it fires
- Unless any of its parents have fired in that time
The Algorithm

Item Eligibility:
- An item becomes eligible $D$ steps after it fires
- Unless any of its parents have fired in that time

Repeat Ad Nauseum:
The Algorithm

Item Eligibility:
- An item becomes eligible $D$ steps after it fires
- Unless any of its parents have fired in that time

Repeat Ad Nauseum:
- Randomly (probability $q$) PJOIN eligible items
The Algorithm

Item Eligibility:
- An item becomes eligible D steps after it fires
- Unless any of its parents have fired in that time

Repeat Ad Nauseum:
- Randomly (probability q) PJOIN eligible items
- Items immediately fire upon creation
Example

\begin{align*}
S_1 & \quad 0_1 \quad 1_1 \\
S_2 & \quad 0_2 \quad 1_2 \\
S_3 & \quad 0_3 \quad 1_3
\end{align*}
Example

\[ S_1 \]

\[ 0_1 \quad 1_1 \]

\[ S_2 \]

\[ 0_2 \quad 1_2 \]

\[ S_3 \]

\[ 0_3 \quad 1_3 \]
Example

\[ 0_1 \xrightarrow{\text{S}_1} 1_1 \]
\[ 0_2 \xrightarrow{\text{S}_2} 1_2 \]
\[ 0_3 \xrightarrow{\text{S}_3} 1_3 \]
Example
Example

A

\(0_1 \rightarrow S_1\)

\(1_1 \rightarrow S_1\)

\(0_2 \rightarrow S_2\)

\(1_2 \rightarrow S_2\)

\(0_3 \rightarrow S_3\)

\(1_3 \rightarrow S_3\)
Example
Example
Example

A

I

S_1

S_2

S_3

0_1 1_1

0_2 1_2

0_3 1_3
Example

\[
\begin{align*}
 &I \\
 &\quad \downarrow \quad \downarrow \\
 &A \\
 &\quad \downarrow \\
 &S_1 \\
 &\quad \downarrow \\
 &S_2 \\
 &\quad \downarrow \\
 &S_3
\end{align*}
\]

0_1 \quad 1_1 \\
0_2 \quad 1_2 \\
0_3 \quad 1_3
Example

I

A

0₁ 1₁
S₁

0₂ 1₂
S₂

0₃ 1₃
S₃
Example
Example
Example
Example

Diagram of a computational process with nodes labeled as follows:

- I
- A
- S₁
- S₂
- S₃

The nodes are connected by arrows indicating the flow of information or operations.
Example
Example
Example
Example
Example
Theorem
Theorem

If
Theorem

If \[ T \geq 4 \log n + 2 \log n/p \]
Theorem

If \( T \geq 4 \log n + 2 \log n/p \) \quad D = O(l)
Theorem

If \[ T \geq 4 \log n + 2 \log n/p \quad D = O(l) \]

1. An item will be successfully made for each pattern
Theorem

If \[ T \geq 4 \log n + 2 \log n/p \quad D = O(l) \]

1. An item will be successfully made for each pattern

2. No more items will be made after each pattern is presented \( O(\log m + \log n) \) times
Proof (Sketch)
Proof (Sketch)

- In $2 \log n/p$ steps, w.h.p, every $S_i$ fires
Proof (Sketch)

- In $2 \log n/p$ steps, w.h.p, every $S_i$ fires
- W.h.p, the height of a random tree is $O(\log n)$
Proof (Sketch)

- In $2 \log n / p$ steps, w.h.p, every $S_i$ fires.
- W.h.p, the height of a random tree is $O(\log n)$
- $T \geq 4 \log n + 2 \log n / p$ allows the tree to be formed.
Proof (Sketch)
Proof (Sketch)

- Fix $p$
Proof (Sketch)

- Fix $p$

- Item $X$ is said to have a good parent $Y$ if
Proof (Sketch)

- Fix p

- Item X is said to have a good parent Y if
  - X fires w.h.p when p is presented
Proof (Sketch)

- Fix $p$

- Item $X$ is said to have a good parent $Y$ if
  - $X$ fires w.h.p when $p$ is presented
  - $Y$ fires when $p$ is presented
Proof (Sketch)

- Fix $p$

- Item $X$ is said to have a good parent $Y$ if
  - $X$ fires w.h.p when $p$ is presented
  - $Y$ fires when $p$ is presented
  - $Y$ prevents $X$ from being eligible during $p$
Proof (Sketch)

- Fix $p$

- Item $X$ is said to have a good parent $Y$ if
  - $X$ fires w.h.p when $p$ is presented
  - $Y$ fires when $p$ is presented
  - $Y$ prevents $X$ from being eligible during $p$

- Given enough time, any item gets a good parent
Proof (Sketch)

- Fix p

- Item X is said to have a good parent Y if
  - X fires w.h.p when p is presented
  - Y fires when p is presented
  - Y prevents X from being eligible during p

- Given enough time, any item gets a good parent

- Danger = new items
Proof (Sketch)
Proof (Sketch)

- If new items come from PJOINs during p, no problem
Proof (Sketch)

- If new items come from PJOINs during \( p \), no problem

- New items come from PJOINs during \( q \) that also fire on \( p \)
Proof (Sketch)

- If new items come from PJOINs during p, no problem

- New items come from PJOINs during q that also fire on p

- Key: in order for the new item to fire during p, it must be a PJOIN of items that fire in both p and q
Proof (Sketch)
Proof (Sketch)

- At most \( n \) items at base in \( pq \)
Proof (Sketch)

● At most \( n \) items at base in \( pq \)

● \( O(n) \) items total in \( pq \)
Proof (Sketch)

- At most n items at base in pq
- $O(n)$ items total in pq
- $O(mn)$ total new items added into p
Proof (Sketch)

- At most $n$ items at base in $pq$
- $O(n)$ items total in $pq$
- $O(mn)$ total new items added into $p$
- Done in $O(\log mn)$ presentations
Sketchiness
Sketchiness

- Good parents hinge on D
Sketchiness

- Good parents hinge on D

- Falls apart if item gets joined with someone at vastly different level
Sketchiness

- Good parents hinge on D
- Falls apart if item gets joined with someone at vastly different level
- Argue that this doesn’t happen often
Sketchiness

- Good parents hinge on D
- Falls apart if item gets joined with someone at vastly different level
- Argue that this doesn’t happen often
  - They do not address this at all
Sketchiness

- Good parents hinge on D
- Falls apart if item gets joined with someone at vastly different level
- Argue that this doesn’t happen often
  - They do not address this at all
  - Left as exercise to reader?
Sketchiness

● Good parents hinge on D

● Falls apart if item gets joined with someone at vastly different level

● Argue that this doesn’t happen often
  ○ They do not address this at all
  ○ Left as exercise to reader?

● Proof by programming
Vicinal?
Vicinal?

- PJJOIN operation can be done vicinally
Vicinal?

- PJOIN operation can be done vicinally
  - If you believe in JOIN, you can believe in PJOIN
Vicinal?

- PJOIN operation can be done vicinally
  - If you believe in JOIN, you can believe in PJOIN

- Algorithm can be done vicinally
Vicinal?

- PJOIN operation can be done vicinally
  - If you believe in JOIN, you can believe in PJOIN

- Algorithm can be done vicinally
  - They took this to be fairly obvious