Indexing, Storing and Retrieving Data in Neural Networks

Vincent Gripon

July 1st, 2017

Outline

Outline

1. Computer Vision and Neural Networks
2. Hopfield Neural Networks
3. Willshaw Neural Networks
4. Conclusion

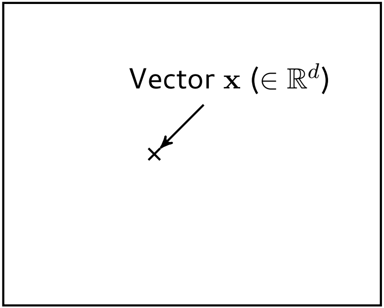
Computer vision problems

Vector space ($\mathbb{R}^d$)

1. Supervised learning,
2. Unsupervised learning,
3. Indexing,
4. Search,
5. ...

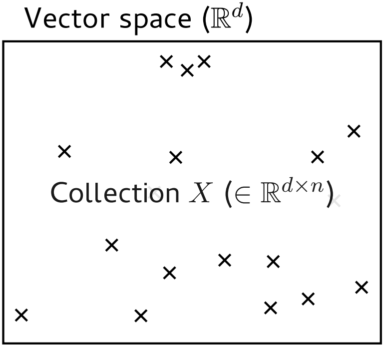
Computer vision problems

Vector space ($\mathbb{R}^d$)



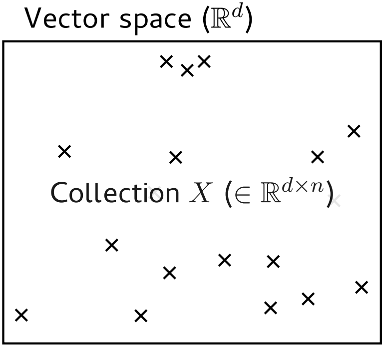
1. Supervised learning,
2. Unsupervised learning,
3. Indexing,
4. Search,
5. ...

Computer vision problems



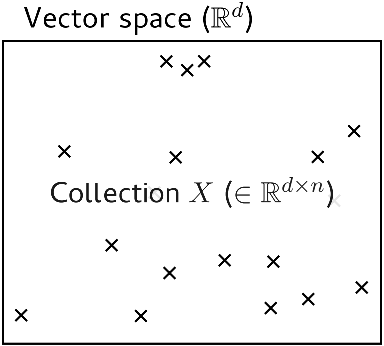
1. Supervised learning,
2. Unsupervised learning,
3. Indexing,
4. Search,
5. ...

Computer vision problems



1. Supervised learning,
2. Unsupervised learning,
3. Indexing,
4. Search,
5. ...

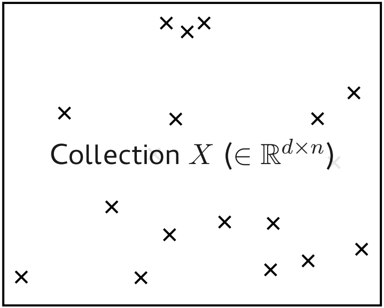
Computer vision problems



1. Supervised learning,
2. Unsupervised learning,
3. Indexing,
4. Search,
5. . . .

Computer vision problems

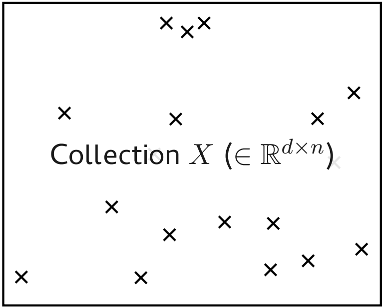
Vector space ($\mathbb{R}^d$)



1. Supervised learning,
2. Unsupervised learning,
3. Indexing,
4. Search,
5. ...

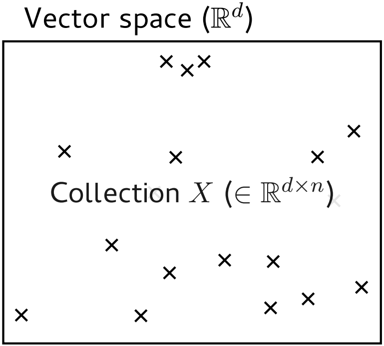
Computer vision problems

Vector space ($\mathbb{R}^d$)



1. Supervised learning,
2. Unsupervised learning,
3. Indexing,
4. Search,
5. ...

Computer vision problems



1. Supervised learning,
2. Unsupervised learning,
3. Indexing,
4. Search,
5. ...

Relation with images

Relation with images

0.88461	0.52899	0.39796	0.156	0.22615	0.16447	0.84366	0.7841	0.05846	0.3335	0.04146	0.79913
0.33711	0.22133	0.46954	0.8197	0.97514	0.79205	0.19736	0.33366	0.05208	0.04472	0.23042	0.93124
0.8572	0.45831	0.87231	0.71693	0.63678	0.54683	0.24892	0.32603	0.82655	0.08347	0.76076	0.59149
0.33481	0.62067	0.78087	0.55115	0.82832	0.46957	0.5429	0.7357	0.49622	0.09038	0.59702	0.38432
0.7534	0.19463	0.41368	0.23335	0.01205	0.18668	0.9122	0.00722	0.64043	0.78145	0.94182	0.77094
0.45204	0.64851	0.0368	0.38763	0.99484	0.14494	0.76273	0.27692	0.33253	0.70724	0.7361	0.36882
0.35962	0.0953	0.47678	0.92337	0.72545	0.3611	0.05582	0.48013	0.5318	0.27792	0.90964	0.15971
0.528	0.4521	0.6933	0.3117	0.57884	0.00188	0.06187	0.60576	0.94542	0.62769	0.82405	0.40215
0.4817	0.3089	0.50847	0.56479	0.91013	0.38911	0.1955	0.19717	0.80548	0.0926	0.54935	0.2212
0.2007	0.39793	0.76196	0.40977	0.5557	0.13638	0.11624	0.72516	0.711	0.37856	0.34254	0.67796
0.18808	0.495	0.61931	0.85258	0.15338	0.95236	0.7579	0.83098	0.89072	0.30334	0.79318	0.93652
0.73792	0.10391	0.66104	0.11888	0.31796	0.11823	0.3503	0.21704	0.67531	0.10696	0.15614	0.88287
0.87881	0.5232	0.7498	0.49826	0.56987	0.82922	0.50221	0.17014	0.14153	0.50203	0.71329	0.5883
0.82059	0.15565	0.77045	0.65742	0.69325	0.81161	0.6689	0.2689	0.3157	0.30891	0.10176	0.50745
0.35197	0.55392	0.10223	0.35126	0.41571	0.52171	0.06691	0.90193	0.5367	0.32317	0.99646	0.32294
0.22276	0.84074	0.81502	0.19206	0.1831	0.95694	0.19064	0.521	0.00731	0.2918	0.75076	0.04846
0.83913	0.02509	0.23013	0.56805	0.48898	0.59413	0.00085	0.5568	0.94305	0.58304	0.97273	0.23122
0.8672	0.99681	0.55971	0.3113	0.93437	0.85008	0.31642	0.69461	0.92604	0.10722	0.9894	0.79062
0.73033	0.55927	0.65729	0.24986	0.83891	0.9922	0.34003	0.88657	0.22913	0.68306	0.26389	0.36389
0.7484	0.26044	0.84335	0.92747	0.1045	0.4189	0.38507	0.61227	0.37956	0.96606	0.76146	0.66606
0.80537	0.69485	0.216	0.07784	0.07887	0.0033	0.78531	0.68555	0.85399	0.55963	0.3055	0.38026
0.58958	0.92455	0.1638	0.98196	0.19221	0.06602	0.34438	0.62881	0.6117	0.7972	0.42224	0.32251
0.91573	0.59866	0.83585	0.58174	0.16328	0.43048	0.93353	0.46086	0.84715	0.23059	0.77397	0.77893
0.9289	0.79767	0.2249	0.59573	0.66231	0.51534	0.836	0.75488	0.11812	0.52908	0.59819	0.46626
0.62892	0.8288	0.5407	0.22375	0.89394	0.67827	0.91756	0.87721	0.45978	0.0395	0.76498	0.70155

Relation with images

Relation with images

Relation with images

Relation with images

1. Features extraction
2. Features comparison

25% error rate in 2011

Supervised learning

Learning

To learn is to **generalize** ($\neq$ memorize),

Supervised learning

- Regression,
- Requires an expert,
- Has a lot of applications:
  - Playing games,
  - Pattern recognition,
  - Attending to a lesson...

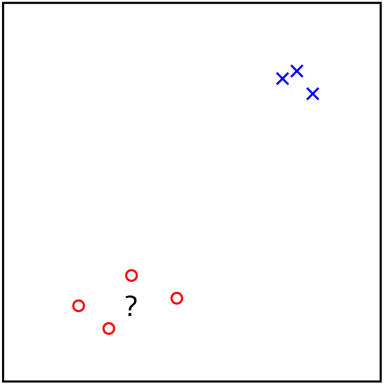
Supervised learning

Learning

To learn is to **generalize** ($\neq$ memorize),

Supervised learning

- Regression,
- Requires an expert,
- Has a lot of applications:
  - Playing games,
  - Pattern recognition,
  - Attending to a lesson...



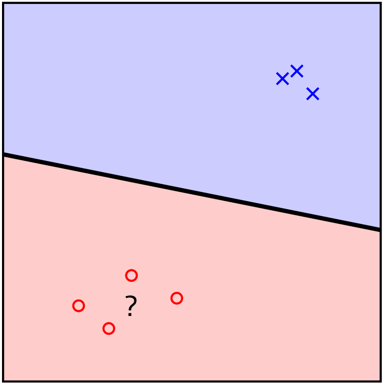
Supervised learning

Learning

To learn is to **generalize** ($\neq$ memorize),

Supervised learning

- Regression,
- Requires an expert,
- Has a lot of applications:
  - Playing games,
  - Pattern recognition,
  - Attending to a lesson...



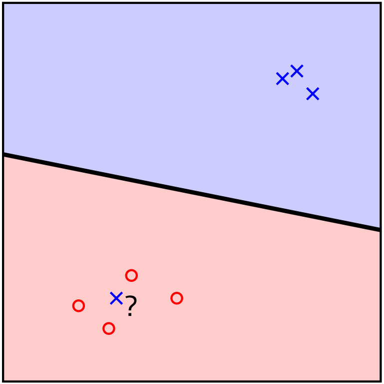
Supervised learning

Learning

To learn is to **generalize** ($\neq$ memorize),

Supervised learning

- Regression,
- Requires an expert,
- Has a lot of applications:
  - Playing games,
  - Pattern recognition,
  - Attending to a lesson...



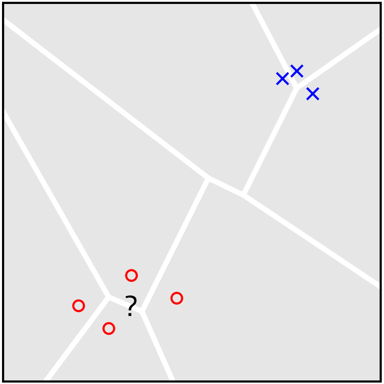
Supervised learning

Learning

To learn is to **generalize** ($\neq$ memorize),

Supervised learning

- Regression,
- Requires an expert,
- Has a lot of applications:
  - Playing games,
  - Pattern recognition,
  - Attending to a lesson...

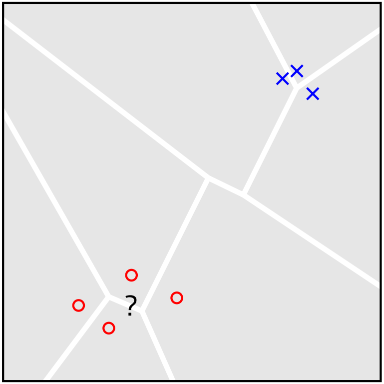


Learning

To learn is to **generalize** ($\neq$ memorize),

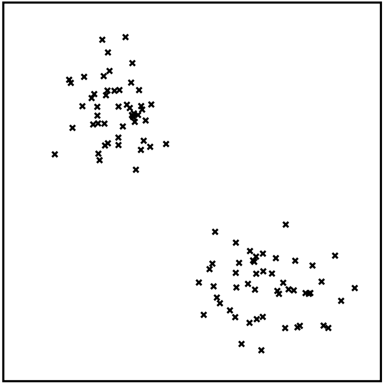
Supervised learning

- Regression,
- Requires an expert,
- Has a lot of applications:
  - Playing games,
  - Pattern recognition,
  - Attending to a lesson...



Classical methods: SVM, k-NN, Random Forests, LR, MLP, CNN...

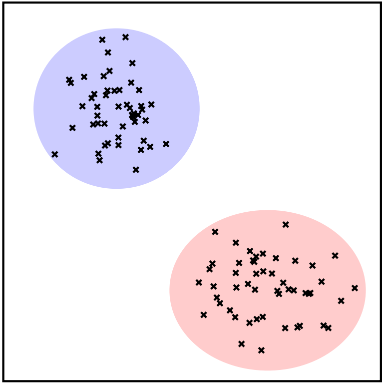
Unsupervised learning



Unsupervised learning

- Partitioning,
- Requires an oracle,
- Many think this is the true support of intelligence:
  - Efficient representations, language,
  - Compression,
  - Automatic generation of hypothesis...

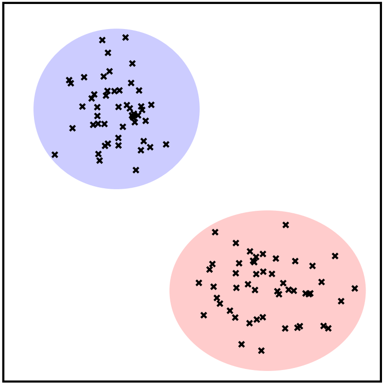
Unsupervised learning



Unsupervised learning

- Partitioning,
- Requires an oracle,
- Many think this is the true support of intelligence:
  - Efficient representations, language,
  - Compression,
  - Automatic generation of hypothesis...

Unsupervised learning



Unsupervised learning

- Partitioning,
- Requires an oracle,
- Many think this is the true support of intelligence:
  - Efficient representations, language,
  - Compression,
  - Automatic generation of hypothesis...

Classical methods: k-means, db-scan, Kohonen maps, autoencoders, EM...

Indexing

Definition

Given a collection $X \in \mathbb{R}^{d \times n}$ and a query vector $\mathbf{x}$:

1. Is $\mathbf{x} \in X$?
2. Do we have $\mathbf{x}' \in X$ s.t. $\mathbf{x}' \approx \mathbf{x}$?

Exhaustive search

- Pros: no error, simple, concurrent,
- Cons: linear with both d and n.

Example of database: SIFT1B:

- $n = 1,000,000,000$,
- $d = 128$,
- 10,000 tests,
- On my laptop, takes approximatively 4 years.

Indexing

Definition

Given a collection $X \in \mathbb{R}^{d \times n}$ and a query vector $\mathbf{x}$:

1. Is $\mathbf{x} \in X$?
2. Do we have $\mathbf{x}' \in X$ s.t. $\mathbf{x}' \approx \mathbf{x}$?

Exhaustive search

- Pros: no error, simple, concurrent,
- Cons: linear with both d and n.

Example of database: SIFT1B:

- $n = 1,000,000,000$,
- $d = 128$,
- 10,000 tests,
- On my laptop, takes approximatively 4 years.

Indexing

Definition

Given a collection $X \in \mathbb{R}^{d \times n}$ and a query vector $\mathbf{x}$:

1. Is $\mathbf{x} \in X$?
2. Do we have $\mathbf{x}' \in X$ s.t. $\mathbf{x}' \approx \mathbf{x}$?

Exhaustive search

- Pros: no error, simple, concurrent,
- Cons: linear with both d and n.

Example of database: SIFT1B:

- $n = 1,000,000,000$,
- $d = 128$,
- 10,000 tests,
- On my laptop, takes approximatively 4 years.

Indexing

Definition

Given a collection $X \in \mathbb{R}^{d \times n}$ and a query vector $\mathbf{x}$:

1. Is $\mathbf{x} \in X$?
2. Do we have $\mathbf{x}' \in X$ s.t. $\mathbf{x}' \approx \mathbf{x}$?

Exhaustive search

- Pros: no error, simple, concurrent,
- Cons: linear with both d and n.

Example of database: SIFT1B:

- $n = 1,000,000,000$,
- $d = 128$,
- 10,000 tests,
- On my laptop, takes approximatively 4 years.

Indexing

Definition

Given a collection $X \in \mathbb{R}^{d \times n}$ and a query vector $\mathbf{x}$:

1. Is $\mathbf{x} \in X$?
2. Do we have $\mathbf{x}' \in X$ s.t. $\mathbf{x}' \approx \mathbf{x}$?

Exhaustive search

- Pros: no error, simple, concurrent,
- Cons: linear with both d and n.

Example of database: SIFT1B:

- $n = 1,000,000,000$,
- $d = 128$,
- 10,000 tests,
- On my laptop, takes approximately 4 years.

Methods

Hash tables (exact)

- Store items in an array of lists,
- Access array addresses using hash functions.

Bloom filters (approximate)

- Retain already-seen hashes,
- Compare with probed ones.

Locality Sensitive Hashing (LSH)

- Use smooth hashes,
- Convert Euclidean to Hamming.

Methods

Hash tables (exact)

- Store items in an array of lists,
- Access array addresses using hash functions.

Bloom filters (approximate)

- Retain already-seen hashes,
- Compare with probed ones.

Locality Sensitive Hashing (LSH)

- Use smooth hashes,
- Convert Euclidean to Hamming.

Methods

Hash tables (exact)

- Store items in an array of lists,
- Access array addresses using hash functions.

Bloom filters (approximate)

- Retain already-seen hashes,
- Compare with probed ones.

Locality Sensitive Hashing (LSH)

- Use smooth hashes,
- Convert Euclidean to Hamming.

Methods

Hash tables (exact)

- Store items in an array of lists,
- Access array addresses using hash functions.

Bloom filters (approximate)

- Retain already-seen hashes,
- Compare with probed ones.

Locality Sensitive Hashing (LSH)

- Use smooth hashes,
- Convert Euclidean to Hamming.

Methods

Hash tables (exact)

- Store items in an array of lists,
- Access array addresses using hash functions.

Bloom filters (approximate)

- Retain already-seen hashes,
- Compare with probed ones.

Locality Sensitive Hashing (LSH)

- Use smooth hashes,
- Convert Euclidean to Hamming.

Methods

Hash tables (exact)

- Store items in an array of lists,
- Access array addresses using hash functions.

Bloom filters (approximate)

- Retain already-seen hashes,
- Compare with probed ones.

Locality Sensitive Hashing (LSH)

- Use smooth hashes,
- Convert Euclidean to Hamming.

Search

Definition

Given a collection $X \in \mathbb{R}^{d\times n}$ and a query vector $\mathbf{x}$, find:

$$\mathbf{x}' = \arg\min_{\mathbf{x}'\in X} \|\mathbf{x} - \mathbf{x}'\|,$$

given some metric.

Methods

- Exhaustive search again,
- Act on n and/or d:
  - On n, partition the search space (problems with high dimensions),
  - On d, quantify the collection and/or the probe (e.g. Product Quantization).

Search

Definition

Given a collection $X \in \mathbb{R}^{d \times n}$ and a query vector $\mathbf{x}$, find:

$$\mathbf{x}' = \arg\min_{\mathbf{x}' \in X} \|\mathbf{x} - \mathbf{x}'\|,$$

given some metric.

Methods

- Exhaustive search again,
- Act on n and/or d:
  - On n, partition the search space (problems with high dimensions),
  - On d, quantify the collection and/or the probe (e.g. Product Quantization).

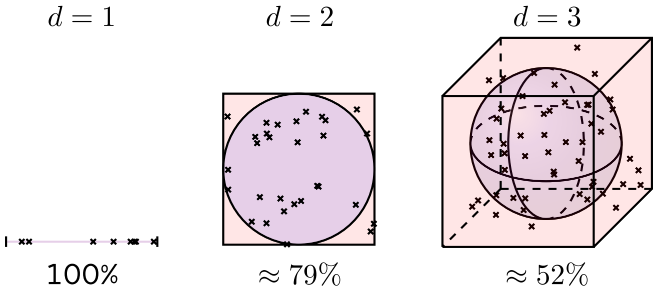
Curse of dimensionality

"Intuition is wrong in high dimension."

$$V_d^s = \frac{\pi^{d/2} R^d}{\Gamma(d/2+1)} \text{ versus } V_d^c = (2R)^d$$

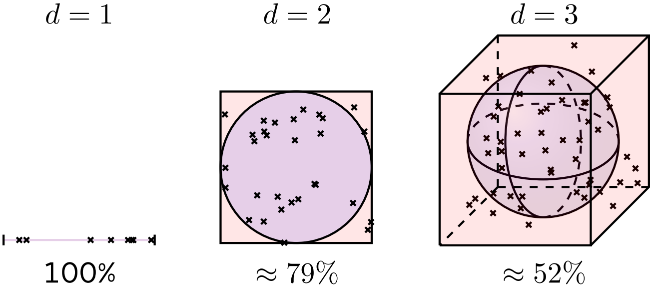
Curse of dimensionality

"Intuition is wrong in high dimension."



$$V_d^s = \frac{\pi^{d/2} R^d}{\Gamma(d/2+1)} \text{ versus } V_d^c = (2R)^d$$

"Intuition is wrong in high dimension."



$$V_d^s = \frac{\pi^{d/2} R^d}{\Gamma(d/2+1)} \text{ versus } V_d^c = (2R)^d$$

Definition

A neural network performs iteratively the concatenation of a linear and a nonlinear function.

$$\mathbf{y} = h_1(W_1(h_2(W_2 \dots h_i(W_i\mathbf{x})))).$$

Nonlinear functions

- Sigmoids (e.g. $x \mapsto 1/\left(1+\exp(-x)\right)$),
- Relus (e.g. $x \mapsto \max(0, x)$),
- Winner-Takes-All (WTA)...

A note on Neural Networks methods for vision

Definition

A neural network performs iteratively the concatenation of a linear and a nonlinear function.

$$\mathbf{y} = h_1(W_1(h_2(W_2 \dots h_i(W_i\mathbf{x})))).$$

Nonlinear functions

- Sigmoids (e.g. $x \mapsto 1/\left(1 + \exp(-x)\right)$),
- Relus (e.g. $x \mapsto \max(0, x)$),
- Winner-Takes-All (WTA)...

Outline

Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d \times n}$,

- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1?1
- $W = \sum_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x}))))))$.

Hopfield Neural Networks

Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d \times n}$,

- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1?1
- $W = \sum_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.

Hopfield Neural Networks

Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d \times n}$,

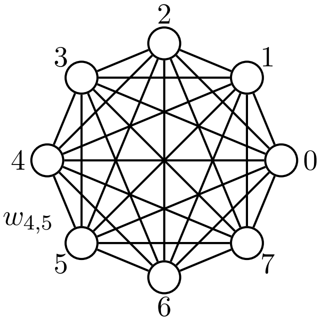
- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1?1
- $W = \sum_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.

2 3 1 4 0 5 7 6

Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d\times n}$,

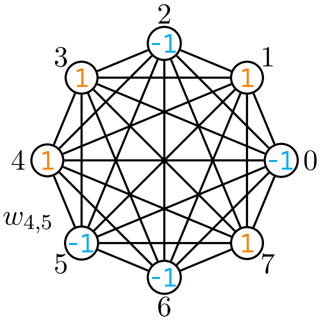
- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1?1
- $W = \sum_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.



Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d \times n}$,

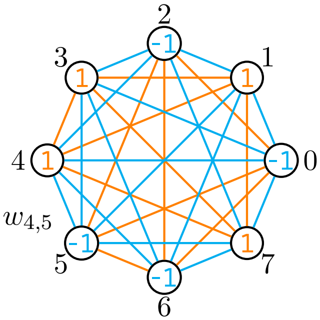
- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1?1
- $W = \sum_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x}))))).$



Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d \times n}$,

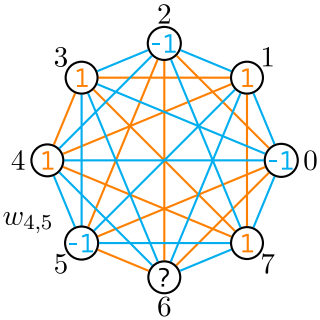
- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1?1
- $W = \sum_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.



Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d \times n}$,

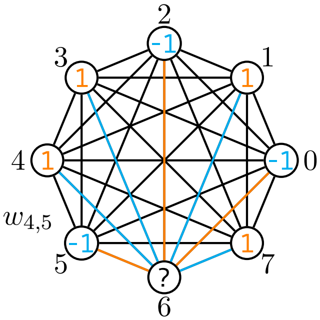
- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1?1
- $W = \sum_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.



Framework

- $\mathbf{x} \in \{-1, 1\}^d$, $X \subset \{-1, 1\}^{d \times n}$,

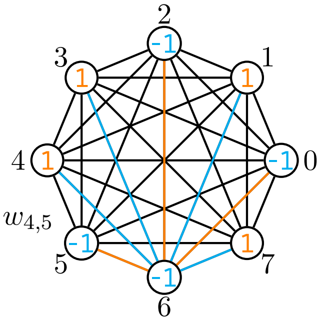
- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1?1
- $W = \sum_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x}))))).$



Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d\times n}$,

- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1-11
- $W = \sum_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.

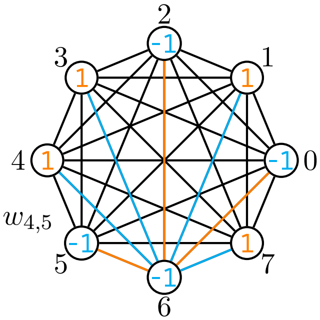


Hopfield Neural Networks

Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d\times n}$,

- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1-11
- $W = \sum_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.

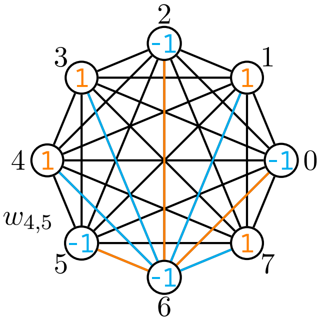


Hopfield Neural Networks

Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d\times n}$,

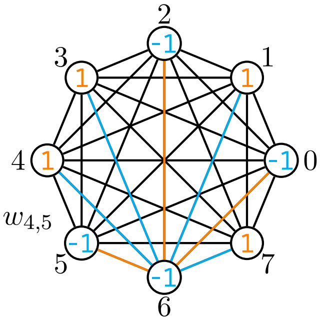
- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1-11
- $W = \sum_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.



Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d \times n}$,

- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1-11
- $W = \sum_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.

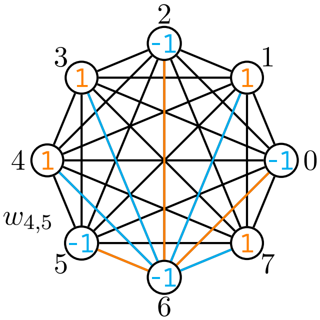


Hopfield Neural Networks

Framework

- $\mathbf{x} \in \{-1,1\}^d$, $X \subset \{-1,1\}^{d \times n}$,

- Example:
  - Storing binary message -11-111-1-11
  - Retrieve it from -11-111-1-11
- $W = \sum_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top) = XX^\top - diag(XX^\top)$,
- $\mathbf{y} = sgn(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.



Stability of stored vectors

Theorem [1]

Consider $n = \frac{d}{\gamma \log(d)}$:

- If $\gamma > 6$, then for $d \to \infty$, $\mathbb{P}[\liminf_d \{\cap_{\mathbf{x} \in X} \{U(\mathbf{x}) = \mathbf{x}\}] = 1$,
- If $\gamma > 4$, then $\mathbb{P}[\cap_{\mathbf{x}} \{U(\mathbf{x}) = \mathbf{x}\}] \to 1$.

Memory efficiency

- $\binom{d}{2}$ connections with $n+1$ possible values each $\Rightarrow$ takes $\binom{d}{2} \log_2(n+1)$ bits without compression,
- To be compared to the entropy of $X \approx nd$ (Why $\approx$?).
- When patterns are stable, we obtain $\eta \leq \frac{1}{2\log(d)\log_2(n+1)}$.

[1] "Étude asymptotique d'un réseau neuronal: le modèle de mémoire associative de Hopfield", Franck Vermet

Stability of stored vectors

Theorem [1]

Consider $n = \frac{d}{\gamma \log(d)}$:

- If $\gamma > 6$, then for $d \to \infty$, $\mathbb{P}[\liminf_{d} \{\cap_{\mathbf{x} \in X} \{U(\mathbf{x}) = \mathbf{x}\}] = 1$,
- If $\gamma > 4$, then $\mathbb{P}[\cap_{\mathbf{x}} \{U(\mathbf{x}) = \mathbf{x}\}] \to 1$.

Memory efficiency

- $\binom{d}{2}$ connections with $n+1$ possible values each $\Rightarrow$ takes $\binom{d}{2} \log_2(n+1)$ bits without compression,
- To be compared to the entropy of $X \approx nd$ (Why $\approx$?).
- When patterns are stable, we obtain $\eta \leq \frac{1}{2\log(d)\log_2(n+1)}$.

[1] "Étude asymptotique d'un réseau neuronal: le modèle de mémoire associative de Hopfield", Franck Vermet

Stability of stored vectors

Theorem [1]

Consider $n = \frac{d}{\gamma \log(d)}$:

- If $\gamma > 6$, then for $d \to \infty$, $\mathbb{P}[\liminf_d \{\cap_{\mathbf{x} \in X}\{U(\mathbf{x}) = \mathbf{x}\}] = 1$,
- If $\gamma > 4$, then $\mathbb{P}[\cap_{\mathbf{x}}\{U(\mathbf{x}) = \mathbf{x}\}] \to 1$.

Memory efficiency

- $\binom{d}{2}$ connections with $n+1$ possible values each $\Rightarrow$ takes $\binom{d}{2}\log_2(n+1)$ bits without compression,
- To be compared to the entropy of $X \approx nd$ (Why $\approx$?).
- When patterns are stable, we obtain $\eta \leq \frac{1}{2\log(d)\log_2(n+1)}$.

[1] "Étude asymptotique d'un réseau neuronal: le modèle de mémoire associative de Hopfield", Franck Vermet

Stability of stored vectors

Theorem [1]

Consider $n = \frac{d}{\gamma \log(d)}$:

- If $\gamma > 6$, then for $d \to \infty$, $\mathbb{P}[\liminf_d \{\cap_{\mathbf{x} \in X}\{U(\mathbf{x}) = \mathbf{x}\}] = 1$,
- If $\gamma > 4$, then $\mathbb{P}[\cap_{\mathbf{x}}\{U(\mathbf{x}) = \mathbf{x}\}] \to 1$.

Memory efficiency

- $\binom{d}{2}$ connections with $n+1$ possible values each $\Rightarrow$ takes $\binom{d}{2}\log_2(n+1)$ bits without compression,
- To be compared to the entropy of $X \approx nd$ (Why $\approx$?).
- When patterns are stable, we obtain $\eta \leq \frac{1}{2\log(d)\log_2(n+1)}$.

[1] "Étude asymptotique d'un réseau neuronal: le modèle de mémoire associative de Hopfield", Franck Vermet

Retrievability of stored vectors

Theorem [1]

Consider $\rho \in [0, 1/2[$ and $n = (1-2\rho)^2 \frac{d}{\gamma \log(d)}$. $\tilde{\mathbf{x}}$ is such that it contains at most ρd symbols different from $\mathbf{x}$, then:

- If $\gamma > 6$, then $\mathbb{P}[\liminf_d \{\cap_{\mathbf{x}} \{U(\tilde{\mathbf{x}}) = \mathbf{x}\}] = 1$,
- If $\gamma > 4$, then $\mathbb{P}[\cap_{\mathbf{x}} \{U(\tilde{\mathbf{x}}) = \mathbf{x}\}] \to 1$.

Hamiltonian

The quantity $H(\mathbf{x}) = -\frac{1}{d}\sum_{i,j=1}^{d} W_{ij}\mathbf{x}_i\mathbf{x}_j$ is nonincreasing with u. More generally, $U(\mathbf{x})$ is a local minimum for H.

[1] "Étude asymptotique d'un réseau neuronal: le modèle de mémoire associative de Hopfield", Franck Vermet

Retrievability of stored vectors

Theorem [1]

Consider $\rho \in [0, 1/2[$ and $n = (1-2\rho)^2 \frac{d}{\gamma \log(d)}$. $\tilde{\mathbf{x}}$ is such that it contains at most ρd symbols different from $\mathbf{x}$, then:

- If $\gamma > 6$, then $\mathbb{P}[\liminf_d \{\cap_{\mathbf{x}} \{U(\tilde{\mathbf{x}}) = \mathbf{x}\}] = 1$,
- If $\gamma > 4$, then $\mathbb{P}[\cap_{\mathbf{x}} \{U(\tilde{\mathbf{x}}) = \mathbf{x}\}] \to 1$.

Hamiltonian

The quantity $H(\mathbf{x}) = -\frac{1}{d}\sum_{i,j=1}^{d} W_{ij}\mathbf{x}_i\mathbf{x}_j$ is nonincreasing with u. More generally, $U(\mathbf{x})$ is a local minimum for H.

[1] "Étude asymptotique d'un réseau neuronal: le modèle de mémoire associative de Hopfield", Franck Vermet

Retrievability of stored vectors

Theorem [1]

Consider $\rho \in [0, 1/2[$ and $n = (1-2\rho)^2 \frac{d}{\gamma \log(d)}$. $\tilde{\mathbf{x}}$ is such that it contains at most ρd symbols different from $\mathbf{x}$, then:

- If $\gamma > 6$, then $\mathbb{P}[\liminf_d \{\cap_{\mathbf{x}} \{U(\tilde{\mathbf{x}}) = \mathbf{x}\}] = 1$,
- If $\gamma > 4$, then $\mathbb{P}[\cap_{\mathbf{x}} \{U(\tilde{\mathbf{x}}) = \mathbf{x}\}] \to 1$.

Hamiltonian

The quantity $H(\mathbf{x}) = -\frac{1}{d} \sum_{i,j=1}^{d} W_{ij} \mathbf{x}_i \mathbf{x}_j$ is nonincreasing with u. More generally, $U(\mathbf{x})$ is a local minimum for H.

[1] "Étude asymptotique d'un réseau neuronal: le modèle de mémoire associative de Hopfield", Franck Vermet

Retrievability of stored vectors

Theorem [1]

Consider $\rho \in [0, 1/2[$ and $n = (1-2\rho)^2 \frac{d}{\gamma \log(d)}$. $\tilde{\mathbf{x}}$ is such that it contains at most ρd symbols different from $\mathbf{x}$, then:

- If $\gamma > 6$, then $\mathbb{P}[\liminf_d \{\cap_{\mathbf{x}} \{U(\tilde{\mathbf{x}}) = \mathbf{x}\}] = 1$,
- If $\gamma > 4$, then $\mathbb{P}[\cap_{\mathbf{x}} \{U(\tilde{\mathbf{x}}) = \mathbf{x}\}] \to 1$.

Hamiltonian

The quantity $H(\mathbf{x}) = -\frac{1}{d} \sum_{i,j=1}^{d} W_{ij} \mathbf{x}_i \mathbf{x}_j$ is nonincreasing with u.
More generally, $U(\mathbf{x})$ is a local minimum for H.

[1] "Étude asymptotique d'un réseau neuronal: le modèle de mémoire associative de Hopfield", Franck Vermet

Retrievability of stored vectors

Theorem [1]

Consider $\rho \in [0, 1/2[$ and $n = (1-2\rho)^2 \frac{d}{\gamma \log(d)}$. $\tilde{\mathbf{x}}$ is such that it contains at most ρd symbols different from $\mathbf{x}$, then:

- If $\gamma > 6$, then $\mathbb{P}[\liminf_d \{\cap_{\mathbf{x}} \{U(\tilde{\mathbf{x}}) = \mathbf{x}\}] = 1$,
- If $\gamma > 4$, then $\mathbb{P}[\cap_{\mathbf{x}} \{U(\tilde{\mathbf{x}}) = \mathbf{x}\}] \to 1$.

Hamiltonian

The quantity $H(\mathbf{x}) = -\frac{1}{d} \sum_{i,j=1}^{d} W_{ij} \mathbf{x}_i \mathbf{x}_j$ is nonincreasing with u. More generally, $U(\mathbf{x})$ is a local minimum for H.

[1] "Étude asymptotique d'un réseau neuronal: le modèle de mémoire associative de Hopfield", Franck Vermet

Demonstration

1. Create network,
2. Test stability,
3. Test retrievability.

Outline

Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d \times n}$.

- Example:
  - Storing binary message 01011001
  - Retrieve it from 010?10?1
- $W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d \times d}$ $(= XX^\top$ for min-max-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.

Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d \times n}$.

- Example:
  - Storing binary message 01011001
  - Retrieve it from 010?10?1
- $W = \max_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d\times d}$ $(= XX^\top$ for $\mathrm{min\text{-}max}$-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\ldots u(\mathbf{x}))))).$

Willshaw Neural Networks

Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d \times n}$.

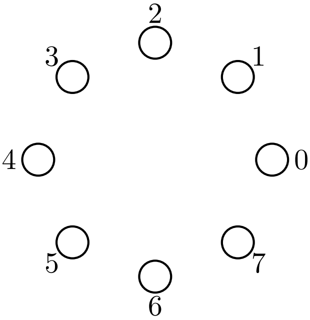
- Example:
  - Storing binary message 01011001
  - Retrieve it from 010?10?1
- $W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d \times d}$ ($= XX^\top$ for min-max-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x}))))).$

Willshaw Neural Networks

Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d \times n}$.

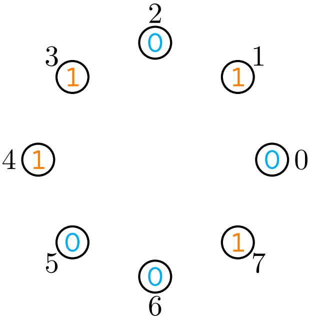
- Example:
  - Storing binary message 01011001
  - Retrieve it from 010?10?1
- $W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d \times d}$ ($= XX^\top$ for min-max-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x}))))).$



Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d \times n}$.

- Example:
  - Storing binary message 01011001
  - Retrieve it from 010?10?1
- $W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d \times d}$ ($= XX^\top$ for $\mathrm{min\text{-}max}$-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.

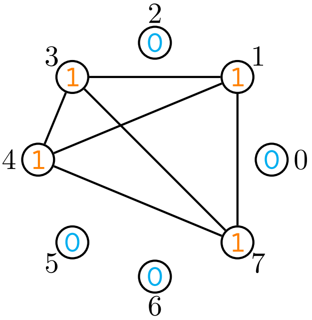


Willshaw Neural Networks

Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d \times n}$.

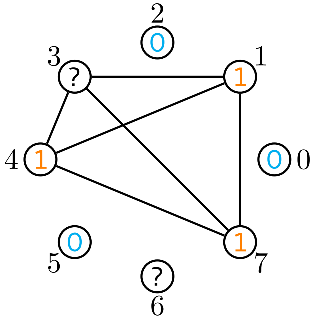
- Example:
  - Storing binary message 01011001
  - Retrieve it from 010?10?1
- $W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d \times d}$ ($= XX^\top$ for min-max-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x}))))).$



Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d\times n}$.

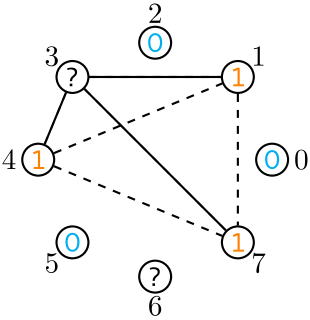
- Example:
  - Storing binary message 01011001
  - Retrieve it from 010?10?1
- $W = \max_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d\times d}$ ($= XX^\top$ for $\mathrm{min\text{-}max}$-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x}))))).$



Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d \times n}$.

- Example:
  - Storing binary message 01011001
  - Retrieve it from 010?10?1
- $W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d \times d}$ ($= XX^\top$ for min-max-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.

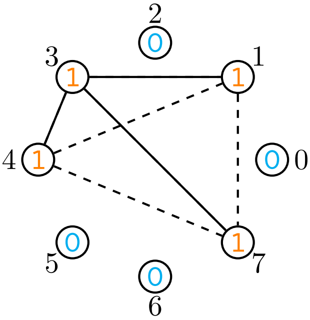


Willshaw Neural Networks

Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d\times n}$.

- Example:
  - Storing binary message 01011001
  - Retrieve it from 01011001
- $W = \max_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d\times d}$ ($= XX^\top$ for min-max-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x}))))).$

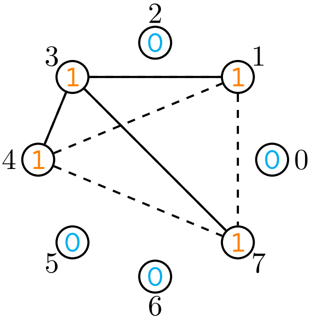


Willshaw Neural Networks

Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d \times n}$.

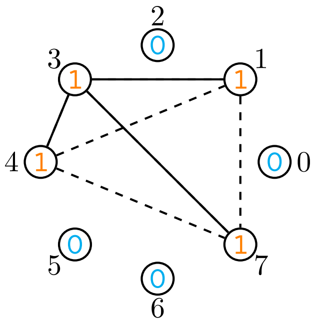
- Example:
  - Storing binary message 01011001
  - Retrieve it from 01011001
- $W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d \times d}$ ($= XX^\top$ for min-max-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\ldots u(\mathbf{x})))))$.



Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d \times n}$.

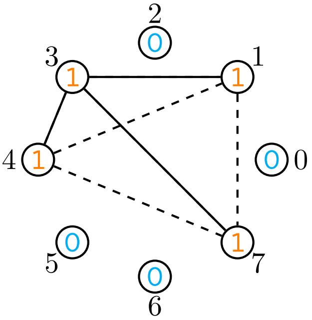
- Example:
  - Storing binary message 01011001
  - Retrieve it from 01011001
- $W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d \times d}$ ($= XX^\top$ for min-max-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.



Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d \times n}$.

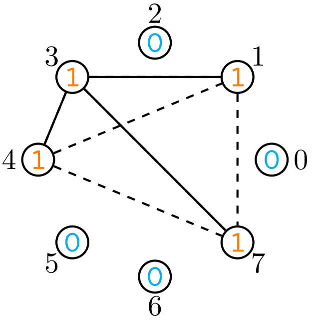
- Example:
  - Storing binary message 01011001
  - Retrieve it from 01011001
- $W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d \times d}$ ($= XX^\top$ for $\min$-$\max$-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\dots u(\mathbf{x})))))$.



Framework

- $\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$, $X \subset \{0,1\}^{d \times n}$.

- Example:
  - Storing binary message 01011001
  - Retrieve it from 01011001
- $W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top \in \{0,1\}^{d \times d}$ ($= XX^\top$ for $\mathrm{min\text{-}max}$-algebra),
- $\mathbf{y} = WTA(W\mathbf{x}) = u(\mathbf{x})$,
- The update can be sequential (one coordinate at a time),
- $U(\mathbf{x}) \triangleq u(u(u(u(\ldots u(\mathbf{x})))))$.



Stability of stored vectors

Theorem [2]

Consider X generated with $\|\mathbf{x}\|_0 = \lfloor \log(d)/d \rfloor$ and $\mathbf{x}$ chosen at random such that $\|\mathbf{x}\|_0 = \lfloor \log(d)/d \rfloor$. With $n = \alpha d^2 \log\log(d)/\log^2(d)$:

- If $\alpha > 2$, $\mathbb{P}[u(\mathbf{x}) = \mathbf{x}] \to 1$,
- If $\alpha = 2$, $\exists \gamma > 0$, for d large enough, $\mathbb{P}[u(\mathbf{x}) = \mathbf{x}] \geq \gamma$,
- If $\alpha < 2$, $\mathbb{P}[u(\mathbf{x}) = \mathbf{x}] \to 0$,

[2] “A Comparative Study of Sparse Associative Memories”, G. et al.

Stability of stored vectors

Theorem [2]

Consider X generated with $\|\mathbf{x}\|_0 = \lfloor \log(d)/d \rfloor$ and $\mathbf{x}$ chosen at random such that $\|\mathbf{x}\|_0 = \lfloor \log(d)/d \rfloor$. With $n = \alpha d^2 \log\log(d)/\log^2(d)$:

- If $\alpha > 2$, $\mathbb{P}[u(\mathbf{x}) = \mathbf{x}] \to 1$,
- If $\alpha = 2$, $\exists \gamma > 0$, for d large enough, $\mathbb{P}[u(\mathbf{x}) = \mathbf{x}] \geq \gamma$,
- If $\alpha < 2$, $\mathbb{P}[u(\mathbf{x}) = \mathbf{x}] \to 0$,

[2] "A Comparative Study of Sparse Associative Memories", G. et al.

Stability of stored vectors

Theorem [2]

Consider X generated with $\|\mathbf{x}\|_0 = \lfloor \log(d)/d \rfloor$ and $\mathbf{x}$ chosen at random such that $\|\mathbf{x}\|_0 = \lfloor \log(d)/d \rfloor$. With $n = \alpha d^2 \log\log(d)/\log^2(d)$:

- If $\alpha > 2$, $\mathbb{P}[u(\mathbf{x}) = \mathbf{x}] \to 1$,
- If $\alpha = 2$, $\exists \gamma > 0$, for d large enough, $\mathbb{P}[u(\mathbf{x}) = \mathbf{x}] \geq \gamma$,
- If $\alpha < 2$, $\mathbb{P}[u(\mathbf{x}) = \mathbf{x}] \to 0$,

[2] "A Comparative Study of Sparse Associative Memories", G. et al.

Memory efficiency

Memory efficiency

- $\binom{d}{2}$ connections with 2 possible values each $\Rightarrow$ takes $\binom{d}{2}$ bits without compression,
- To be compared to the entropy of X:

$$\approx ndH_2(\log(d)/d).$$

- When patterns are stable, we obtain

$$\eta \geq \frac{\alpha d(\log\log(d))^2}{2\log(d)} \to +\infty$$

Why?

Retrievability of stored vectors

Theorem [2]

Consider $n = \alpha d^2 / \log^2(d)$, $\rho \in [0, 1[$ such that $\lfloor \rho \log(d) \rfloor$ of 1s in $\mathbf{x}$ are erased to obtain $\tilde{\mathbf{x}}$. Then:

- If $\alpha < -\log(1 - \exp(-1/(1-\rho)))$, then $\mathbb{P}[u(\tilde{\mathbf{x}}) = \mathbf{x}] \to 1$,
- If $\alpha > -\log(1 - \exp(-1/(1-\rho)))$, then $\mathbb{P}[u(\tilde{\mathbf{x}}) \neq \mathbf{x}] \to 1$.

[2] "A Comparative Study of Sparse Associative Memories", G. et al.

Retrievability of stored vectors

Theorem [2]

Consider $n = \alpha d^2 / \log^2(d)$, $\rho \in [0, 1[$ such that $\lfloor \rho \log(d) \rfloor$ of 1s in $\mathbf{x}$ are erased to obtain $\tilde{\mathbf{x}}$. Then:

- If $\alpha < -\log(1 - \exp(-1/(1-\rho)))$, then $\mathbb{P}[u(\tilde{\mathbf{x}}) = \mathbf{x}] \to 1$,
- If $\alpha > -\log(1 - \exp(-1/(1-\rho)))$, then $\mathbb{P}[u(\tilde{\mathbf{x}}) \neq \mathbf{x}] \to 1$.

[2] "A Comparative Study of Sparse Associative Memories", G. et al.

Retrievability of stored vectors

Theorem [2]

Consider $n = \alpha d^2 / \log^2(d)$, $\rho \in [0, 1[$ such that $\lfloor \rho \log(d) \rfloor$ of 1s in $\mathbf{x}$ are erased to obtain $\tilde{\mathbf{x}}$. Then:

- If $\alpha < -\log(1 - \exp(-1/(1-\rho)))$, then $\mathbb{P}[u(\tilde{\mathbf{x}}) = \mathbf{x}] \to 1$,
- If $\alpha > -\log(1 - \exp(-1/(1-\rho)))$, then $\mathbb{P}[u(\tilde{\mathbf{x}}) \neq \mathbf{x}] \to 1$.

[2] "A Comparative Study of Sparse Associative Memories", G. et al.

Demonstration

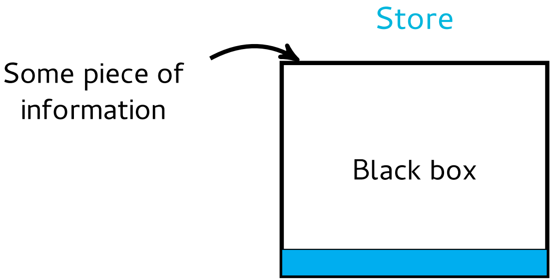
1. Create network,
2. Test stability,
3. Test retrievability.

Outline

Store

Black box

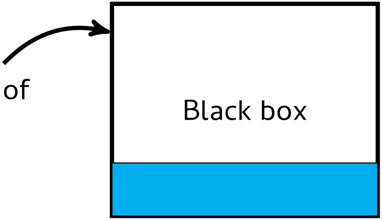
Associative memories



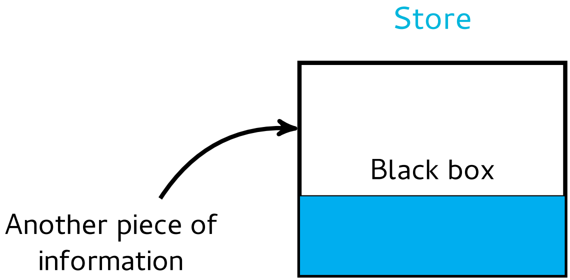
Associative memories

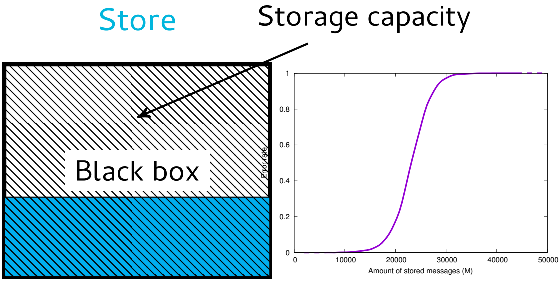
Store

Another piece of information

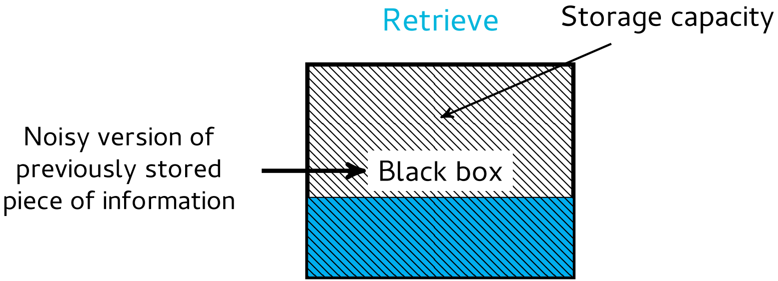


Associative memories

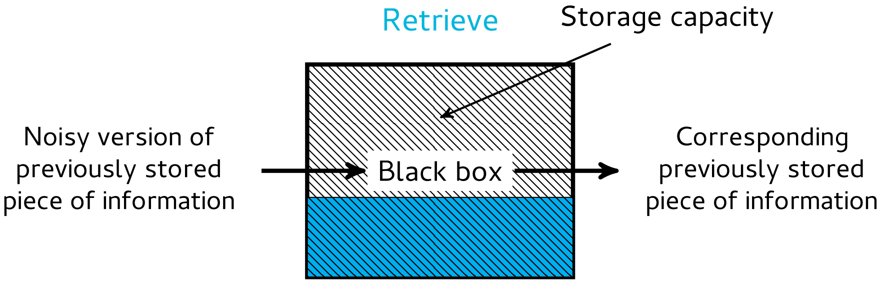



Store
Storage capacity
Black box
Error rate
0
0.2
0.4
0.6
0.8
1
0
10000
20000
30000
40000
50000
Amount of stored messages (M)

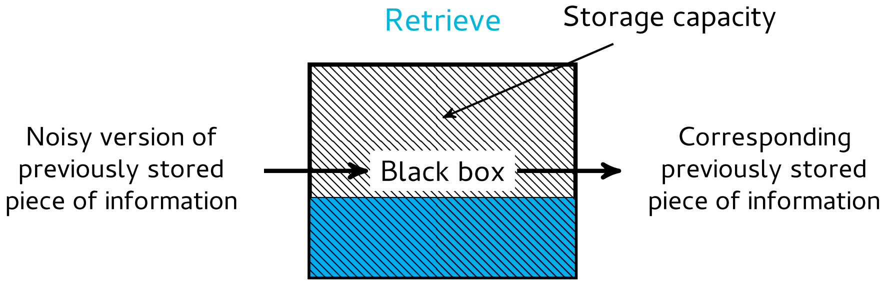
Associative memories



Associative memories



Associative memories



Piece of information = message, retrieve = decode
associative memory = universal decoder

Summary

	Hopfield	Willshaw
Framework	$\mathbf{x} \in \{-1,1\}^d$	$\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$
Memory	$\mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top)$	$\mathbf{x}\mathbf{x}^\top$
Aggregation	$W = \sum_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top)$	$W = \max_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top$
Search	$u(W \cdot \tilde{\mathbf{x}})$	$u(W \otimes \tilde{\mathbf{x}})$

Summary

	Hopfield	Willshaw
Framework	$\mathbf{x} \in \{-1, 1\}^d$	$\mathbf{x} \in \{0, 1\}^d$, $\|\mathbf{x}\|_0 \ll d$
Memory	$\mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top)$	$\mathbf{x}\mathbf{x}^\top$
Aggregation	$W = \sum_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top)$	$W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top$
Search	$u(W \cdot \tilde{\mathbf{x}})$	$u(W \otimes \tilde{\mathbf{x}})$

Summary

	Hopfield	Willshaw
Framework	$\mathbf{x} \in \{-1, 1\}^d$	$\mathbf{x} \in \{0, 1\}^d$, $\|\mathbf{x}\|_0 \ll d$
Memory	$\mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top)$	$\mathbf{x}\mathbf{x}^\top$
Aggregation	$W = \sum_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top)$	$W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top$
Search	$u(W \cdot \tilde{\mathbf{x}})$	$u(W \otimes \tilde{\mathbf{x}})$

Summary

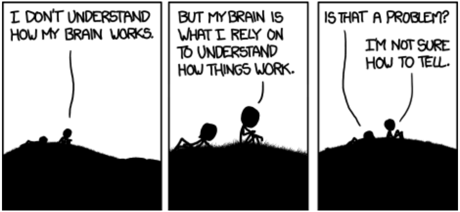
	Hopfield	Willshaw
Framework	$\mathbf{x} \in \{-1,1\}^d$	$\mathbf{x} \in \{0,1\}^d$, $\|\mathbf{x}\|_0 \ll d$
Memory	$\mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top)$	$\mathbf{x}\mathbf{x}^\top$
Aggregation	$W = \sum_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top)$	$W = \max_{\mathbf{x}\in X} \mathbf{x}\mathbf{x}^\top$
Search	$u(W \cdot \tilde{\mathbf{x}})$	$u(W \otimes \tilde{\mathbf{x}})$

Summary

	Hopfield	Willshaw
Framework	$\mathbf{x} \in \{-1, 1\}^d$	$\mathbf{x} \in \{0, 1\}^d$, $\|\mathbf{x}\|_0 \ll d$
Memory	$\mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top)$	$\mathbf{x}\mathbf{x}^\top$
Aggregation	$W = \sum_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top - diag(\mathbf{x}\mathbf{x}^\top)$	$W = \max_{\mathbf{x} \in X} \mathbf{x}\mathbf{x}^\top$
Search	$u(W \cdot \tilde{\mathbf{x}})$	$u(W \otimes \tilde{\mathbf{x}})$

Conclusion/Take home message

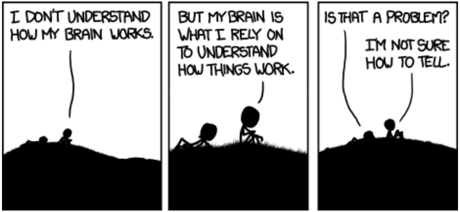
- Neural networks can do much more than learning,
- Neural networks are not just big mathematical functions,
- Storing and indexing boils down to Gram matrices and strange algebras.



XKCD

Conclusion/Take home message

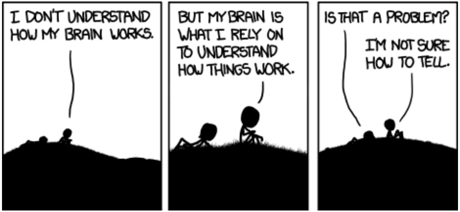
- Neural networks can do much more than learning,
- Neural networks are not just big mathematical functions,
- Storing and indexing boils down to Gram matrices and strange algebras.



XKCD

Conclusion/Take home message

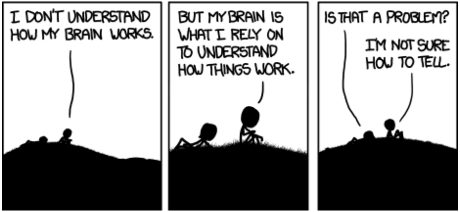
- Neural networks can do much more than learning,
- Neural networks are not just big mathematical functions,
- Storing and indexing boils down to Gram matrices and strange algebras.



XKCD

Conclusion/Take home message

- Neural networks can do much more than learning,
- Neural networks are not just big mathematical functions,
- Storing and indexing boils down to Gram matrices and strange algebras.



XKCD