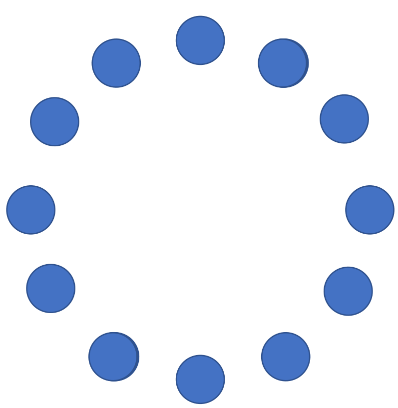
Hopfield Network

1

Architecture

Autoassociative:

Every unit has a connection to and from every other unit (except itself)

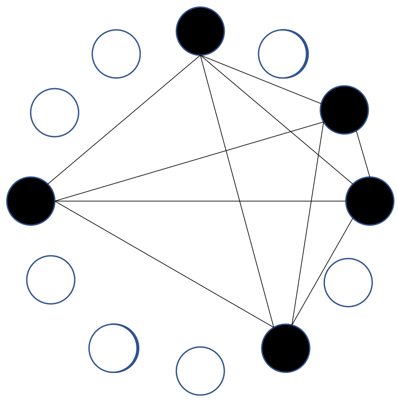


2

What the Network Does

Auto-association: Learn patterns

Train a pattern by activating its units and updating the weights

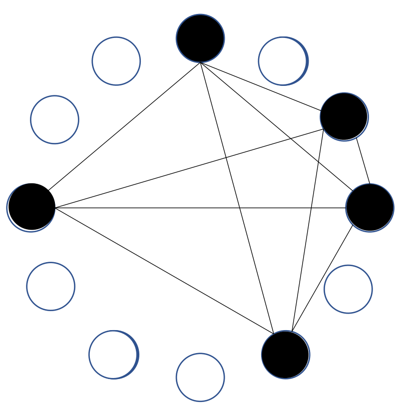


3

What the Network Does

Auto-association: Learn patterns

Run the network by activating a partial pattern and letting it fill in the rest



Psychology:
Simulates *Content-Addressable* Memory

Engineering:
Performs pattern completion/restoration (e.g., in signal processing)

4

Activation Function

Hopfield's Notation:

$$V_i^t = \begin{cases} 1 \; if \; \sum_j T_{ij} V_j^{t-1} > \theta_i \\ 0 \; otherwise \end{cases}$$

Net Input:

$$n_i = \sum_j w_{ij} a_j$$

In our notation:

$$a_i^t = \begin{cases} 1 \; if \; \boxed{\sum_j w_{ij} a_j^{t-1}} > \theta_i \\ 0 \; otherwise \end{cases}$$

Binary Threshold Unit:
Activation = 1 iff net input on previous iteration, *t*, is *greater than* threshold.

NB: Not "greater than or equal to", strictly *greater than*

5

Learning Rule

Hopfield's Notation:

$$T_{ij} = \sum_s (2V_i^s - 1)(2V_j^s - 1)$$

In our notation:

$$\Delta w_{ij}^p = (2a_i^p - 1)(2a_j^p - 1)$$

a_i	a_j	Δw_{ij}
0	0	+1
0	1	-1
1	0	-1
1	1	+1

If the activations are the same, then increase the weight.
If they are different, then decrease the weight.

6

Energy

Hopfield's Notation:

$$E = -\frac{1}{2}\sum_{ij} T_{ij}V_iV_j$$

In our notation:

$$E = -\frac{1}{2}\sum_{ij} w_{ij}a_ia_j$$

$w_{ij}a_ia_j$ is local match: "goodness" or "fit"

The sum of these is the total goodness/fit

The negation of goodness/fit is "badness": *Energy*

7

The Energy Landscape

- A super useful (and interesting) epiphenomenon of the Hopfield model
- An *n*-dimensional network is an *n*-dimensional vector space
- Any vector of activations – *any state of the network* – is one point in this space
- There are 2^n such points in an *n*-dimensional Hopfield network
  - (each one is a corner of the *n*-dimensional hypercube forming the space)
- Every vector (state) of activations in a Hopfield network has an *energy*
  - (which is determined by the *weight matrix*, i.e., the set of all the weights in the network)
- The set of all possible states of activation forms an *n*-dimensional "landscape", where each point in that landscape has an energy (think "altitude" or "height")
- A Hopfield network "moves around" in this space, changing the activations of units in a way that goes "downhill" in the energy landscape, from points of higher energy to points of lower energy

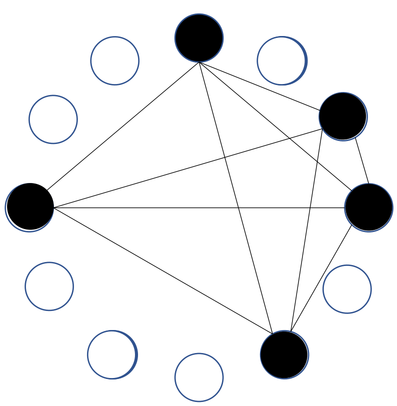
8

Training the Network Creates the Energy Landscape

- An untrained network has a perfectly flat energy landscape: energy = 0 at all points
  - Recall: $E = -\frac{1}{2}\sum_{ij} w_{ij}a_i a_j$.
  - Before training, all $w_{ij} = 0$, so the above will always be the sum of a bunch of 0s, regardless of a_i and a_j (that is, for *all possible* activation states, $\mathbf{a}$)
  - Since all activation states have the same energy (0), the landscape is flat
- The energy landscape of the network is defined by the set (i.e., matrix), **W**, of all the weights, w_{ij}
  - (The activation state, $\mathbf{a}$, sets the value of a_i and a_j, so the only free variable is w_{ij}.)
- Training the network sets the values of the weights
- Therefore, training the network creates the energy landscape
  - Different training set → different energy landscape

9

Training one Pattern Creates an Energy Well*



Recall this training pattern, and let the black lines be connection weights of 1 (all others are zero).

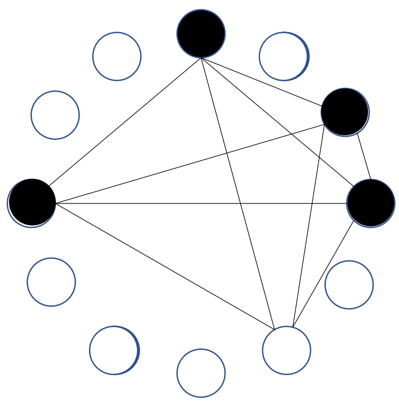
The energy of this trained pattern is -10:
- 20 weights ($5^2 - 5$), each with a value of 1
- Each multiplied by $a_i * a_j = 1 * 1 = 1 * 20$
- All multiplied by -1/2 = -10

Let's step away from the trained pattern and see what happens to energy…

*To a first approximation. As you'll see, the truth is more complicated than this. But bear with me…

10

Training one Pattern Creates an Energy Well*



Recall this training pattern, and let the black lines be connection weights of 1 (all others are zero).

The energy of this trained pattern is -10:

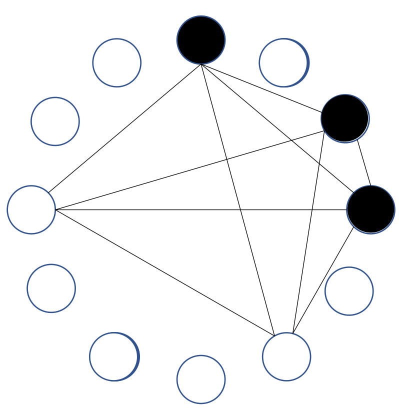
- 20 weights ($5^2 - 5$), each with a value of 1
- Each multiplied by $a_i * a_j = 1 * 1 = 1 * 20$
- All multiplied by -1/2 = -10

Let's step away from the trained pattern and see what happens to energy…

Energy = 12 weights ($4^2 - 4$) * -1/2 = -6

11

Training one Pattern Creates an Energy Well*



Recall this training pattern, and let the black lines be connection weights of 1 (all others are zero).

The energy of this trained pattern is -10:

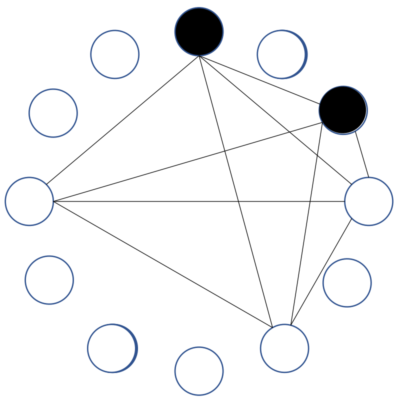
- 20 weights ($5^2 - 5$), each with a value of 1
- Each multiplied by $a_i * a_j = 1 * 1 = 1 * 20$
- All multiplied by -1/2 = -10

Let's step away from the trained pattern and see what happens to energy…

Energy = 6 weights ($3^2 - 3$) * -1/2 = -3

12

Training one Pattern Creates an Energy Well*



Recall this training pattern, and let the black lines be connection weights of 1 (all others are zero).

The energy of this trained pattern is -10:

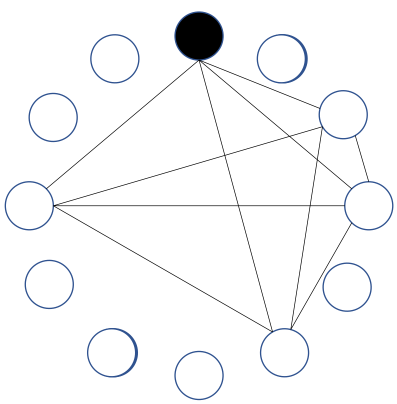
- 20 weights ($5^2 – 5$), each with a value of 1
- Each multiplied by $a_i * a_j = 1 * 1 = 1 * 20$
- All multiplied by -1/2 = -10

Let's step away from the trained pattern and see what happens to energy…

Energy = 2 weights ($2^2 – 2$) * -1/2 = -1

13

Training one Pattern Creates an Energy Well*



Recall this training pattern, and let the black lines be connection weights of 1 (all others are zero).

The energy of this trained pattern is -10:

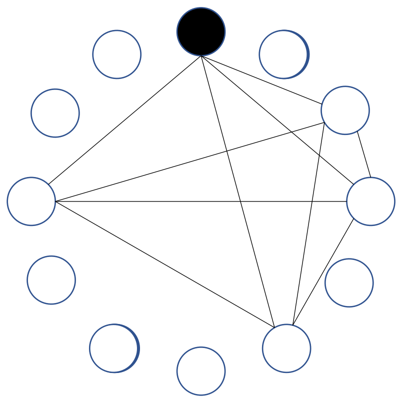
- 20 weights ($5^2 – 5$), each with a value of 1
- Each multiplied by $a_i * a_j = 1 * 1 = 1 * 20$
- All multiplied by -1/2 = -10

Let's step away from the trained pattern and see what happens to energy…

Energy = 0 weights * -1/2 = 0

14

Training one Pattern Creates an Energy Well*



Recall this training pattern, and let the black lines be connection weights of 1 (all others are zero).

The energy of this trained pattern is -10:

- 20 weights $(5^2 - 5)$, each with a value of 1
- Each multiplied by $a_i * a_j = 1 * 1 = 1 * 20$
- All multiplied by -1/2 = -10

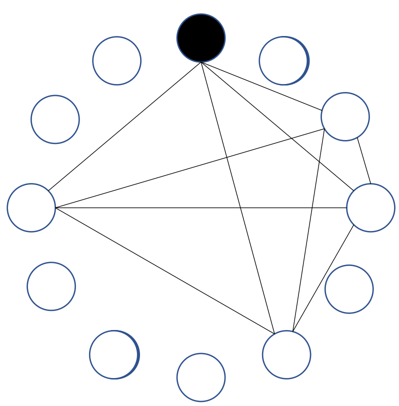
Let's step away from the trained pattern and see what happens to energy…

As we stepped further away from the trained pattern, the energy went up: -10, -6, -3, -1, 0

Energy well: The closer you are to the training pattern, the lower the energy gets.

15

Hopfield Network do Gradient Descent in the Energy Landscape

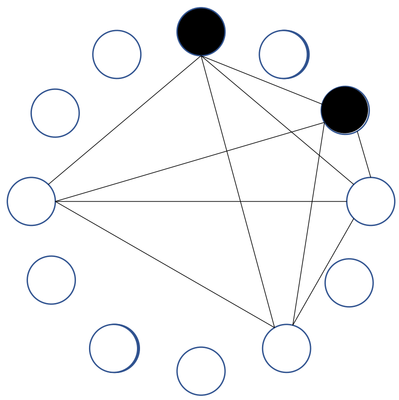


If we run this network with this starting pattern of activation and keep track of energy as we go…

Iteration	Energy
0	0

16

Hopfield Network do Gradient Descent in the Energy Landscape

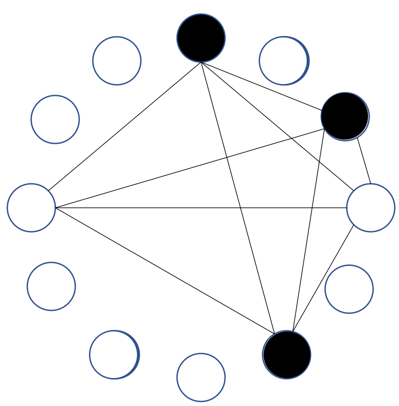


If we run this network with this starting pattern of activation and keep track of energy as we go…

Iteration	Energy
0	0
1	-1

17

Hopfield Network do Gradient Descent in the Energy Landscape

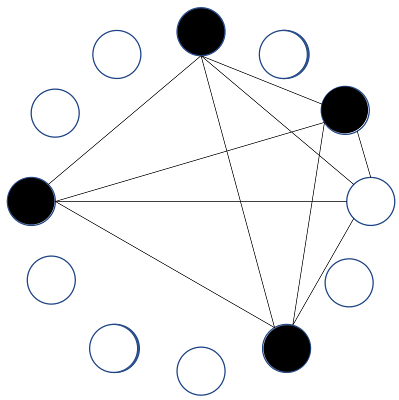


If we run this network with this starting pattern of activation and keep track of energy as we go…

Iteration	Energy
0	0
1	-1
2	-3

18

Hopfield Network do Gradient Descent in the Energy Landscape

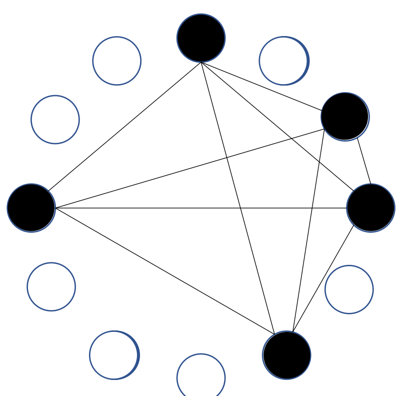


If we run this network with this starting pattern of activation and keep track of energy as we go…

Iteration	Energy
0	0
1	-1
2	-3
3	-6

19

Hopfield Network do Gradient Descent in the Energy Landscape



If we run this network with this starting pattern of activation and keep track of energy as we go…

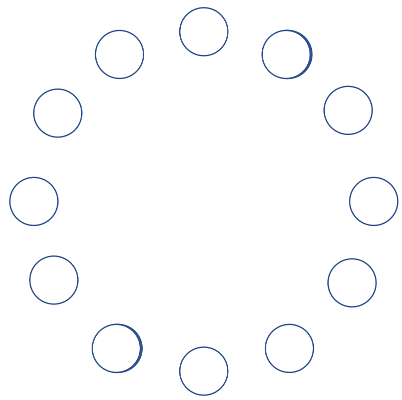
Iteration	Energy
0	0
1	-1
2	-3
3	-6
4	-10

Note that energy falls off in an accelerating fashion in spite of the "-1/2" correction: The number of weights, and thus the (negative of) energy, increases with the square of the number of units in the pattern.

20

The Ugly Reality of the Energy Landscape

Let's take a more realistic look at the energy landscape of a Hopfield network…



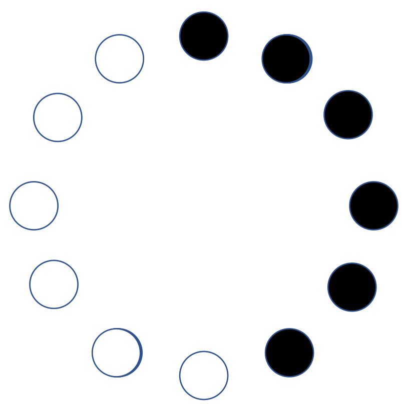
1) When you teach a Hopfield net a pattern, you are really teaching it *two* patterns:

$$\Delta w_{ij}^{p} = (2a_i^p - 1)(2a_j^p - 1)$$

21

The Ugly Reality of the Energy Landscape

Let's take a more realistic look at the energy landscape of a Hopfield network…



1) When you teach a Hopfield net a pattern, you are really teaching it *two* patterns:

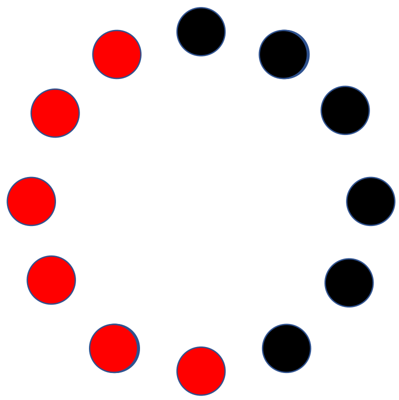
$$\Delta w_{ij}^{p} = (2a_i^p - 1)(2a_j^p - 1)$$

Teach it *this*…

22

The Ugly Reality of the Energy Landscape

Let's take a more realistic look at the energy landscape of a Hopfield network…



1) When you teach a Hopfield net a pattern, you are really teaching it *two* patterns:

$$\Delta w_{ij}^{p} = (2a_i^p - 1)(2a_j^p - 1)$$

Teach it *this*…

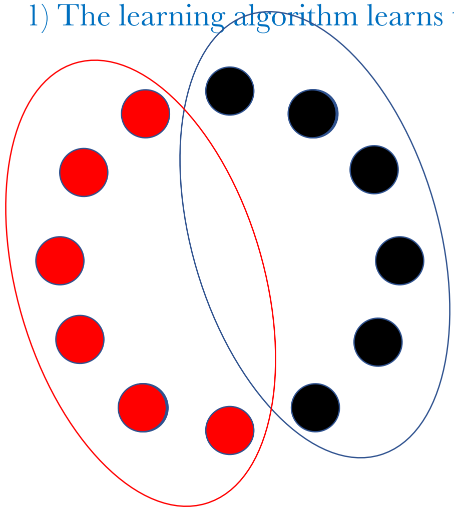
… and it will also learn *this*.

The algorithm increments w_{ij} for any a_i and a_j that are *the same*, even if they're both zero.

23

The Ugly Reality of the Energy Landscape

1) The learning algorithm learns two energy wells for each pattern…



… the pattern you *think* you've trained,

and its inverse.

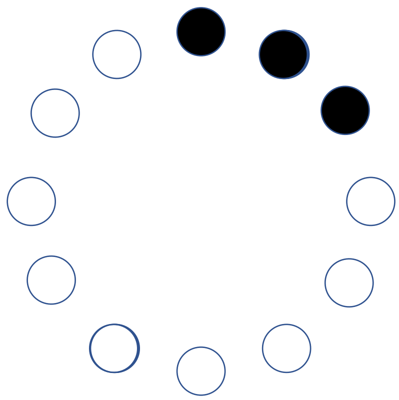
In this (pretty harmless) example, the patterns have the same number of units, so the energy wells are of equal depth.

(Remember that the depth of the energy well is a function of how many units are in the pattern.)

24

The Ugly Reality of the Energy Landscape

1) The learning algorithm learns two energy wells for each pattern…

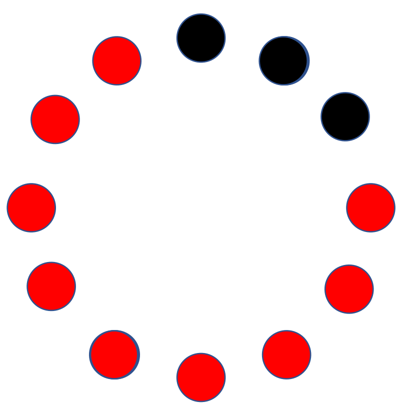


But if this is the pattern you *think* you've trained (energy = -3: $3^2 - 3 * -1/2 = -3$),

25

The Ugly Reality of the Energy Landscape

1) The learning algorithm learns two energy wells for each pattern…



But if this is the pattern you *think* you've trained (energy = -3),

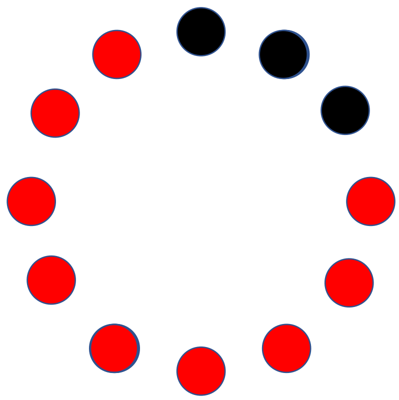
then *this* is the pattern you've *really* trained (energy = -36!: $9^2 - 9 = 72 * -1/2 = -36$)

The pattern you didn't mean to train is *12 times as powerful* as the one you did mean to.

The absurdity is painful!
(It's a good thing we thought carefully about the logical implications of what we're doing, eh?)

26

<u>Question</u>: "Why don't you just [do whatever] to *fix* this?"



<u>Answer</u>: **You could!** And I say, ***Go for it!***

But you have to recognize the problem first.

Only a modeler (a computer-aided logician) would even recognize the problem.

Would you have recognized this problem with "associative learning" 2,000 years ago?

No one did at the time.

And almost no one does now.

But this is what people who believe in "associative learning" have signed up for without even knowing it.
And *this* is why we do modeling: *To protect ourselves from believing stupid shit.*

27

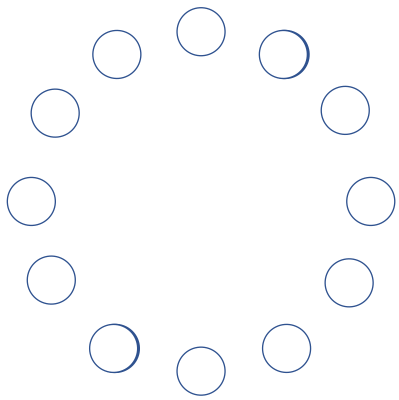
<u>Aside</u>: Responsible Modeling

- **Thinking like a modeler is a good way to protect yourself from bullshit.**
- *Being* a modeler is also a way to convince people (especially the math-timid) of bullshit.
  - Many modelers use math to intimidate non-modelers. *This is evil.*
  - ***Do. Not. Be. Intimidated.*** (If you don't understand the math, *call them on it.*)
  - *The best way to not fall for it is to understand it before they start spewing it.*
- Also, ***never use modeling as a weapon to intimidate others***.
  - It's a dishonest, asshole move. *Smart people will disrespect you instantly.*
  - **Worse: It hides the reality of your ideas from yourself**.
  - The whole point of modeling is to understand what your ideas *really* imply.
- Modeling is like a martial art
  - It can be used for good or evil
  - Use it to defend yourself against bad ideas (yours and others')
  - Never use it to intimidate others into believing bad ideas

28

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…

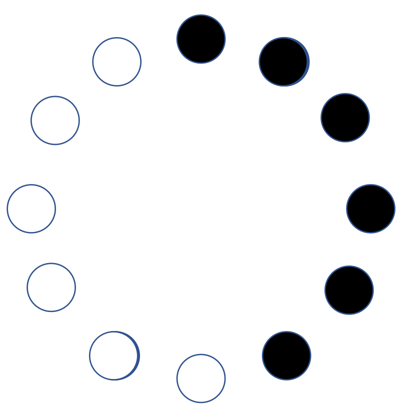


2.1) Energy wells are not small

29

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



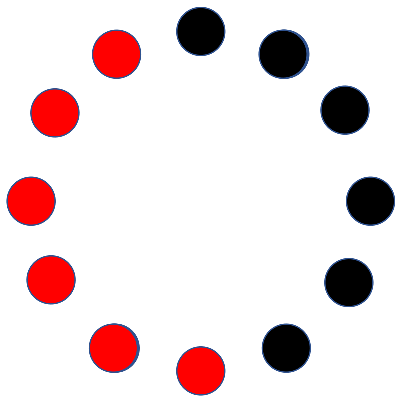
2.1) Energy wells are not small

- Say you train this…

30

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



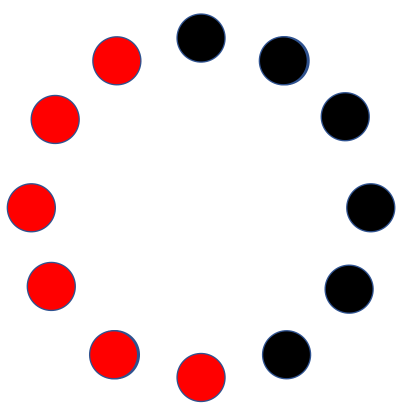
2.1) Energy wells are not small

- Say you train this…
- … (which means you also trained this)…

31

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



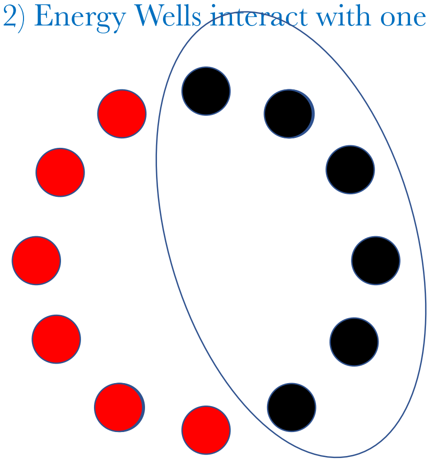
2.1) Energy wells are not small

- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*

32

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



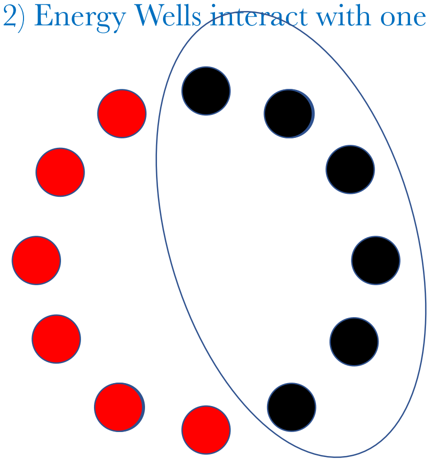
2.1) Energy wells are not small

- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *This one covers…*

33

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



2.1) Energy wells are not small

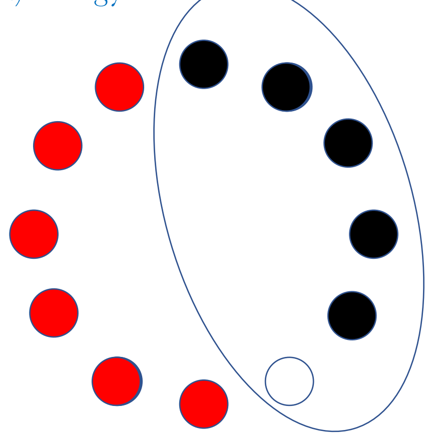
- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *This one covers…*

This: Energy = -15 = $(6^2 - 6) * -1/2$

34

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



2.1) Energy wells are not small

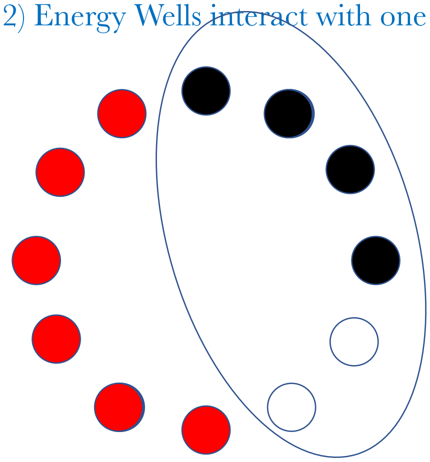
- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *This one covers…*

And this: Energy = -10 = $(5^2 - 5) * -1/2$

35

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



2.1) Energy wells are not small

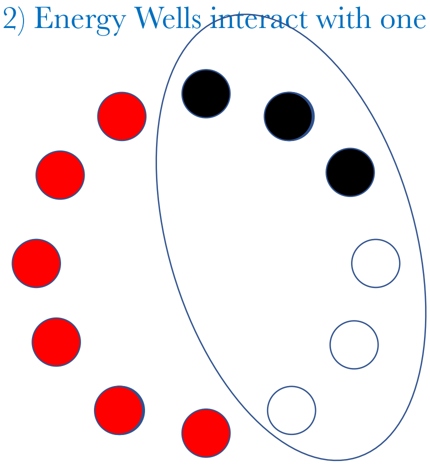
- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *This one covers…*

And this: Energy = -6 = $(4^2 - 4) * -1/2$

36

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



2.1) Energy wells are not small

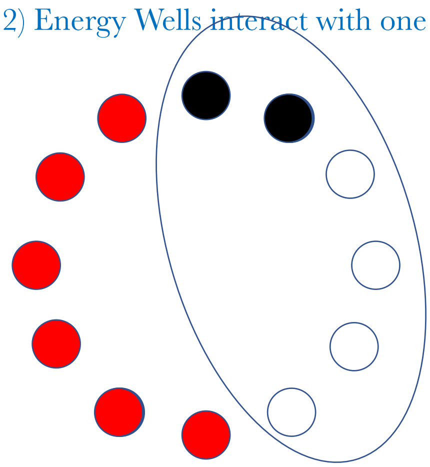
- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *This one covers…*

And this: Energy = -3 = $(3^2 - 3) * -1/2$

37

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



2.1) Energy wells are not small

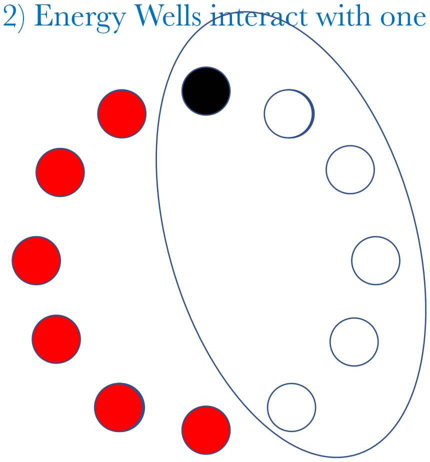
- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *This one covers…*

And this: Energy = -1 = $(2^2 - 2) * -1/2$

38

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



2.1) Energy wells are not small

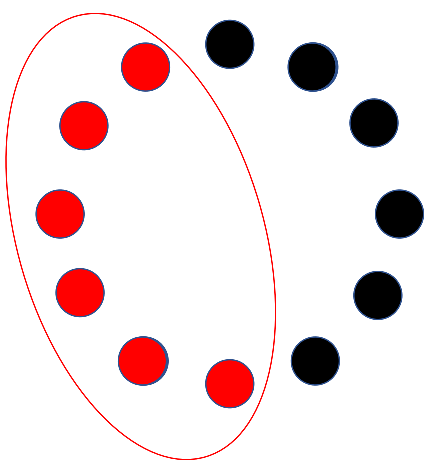
- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *This one covers…*

And this: Energy = 0

39

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



2.1) Energy wells are not small

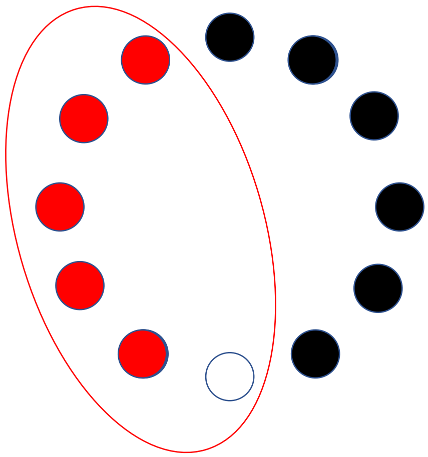
- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *And This one covers…*

This: Energy = -15

40

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



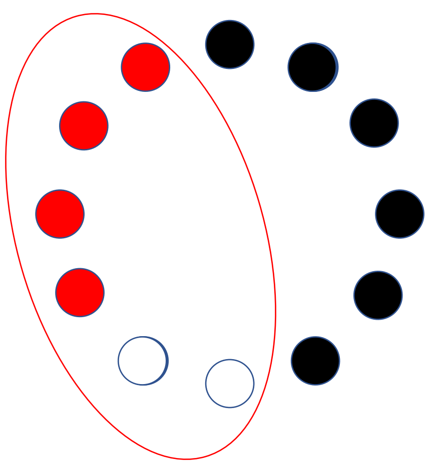
2.1) Energy wells are not small
- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *And This one covers…*

And this: Energy = -10

41

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



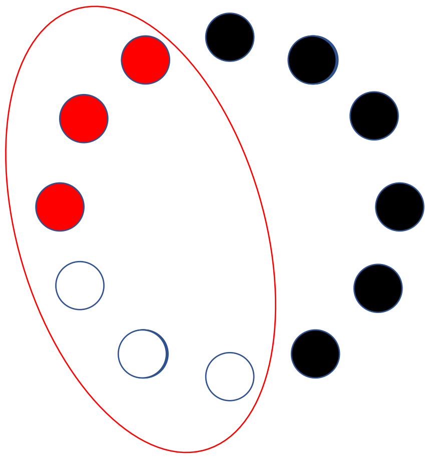
2.1) Energy wells are not small
- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *And This one covers…*

And this: Energy = -6

42

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



2.1) Energy wells are not small

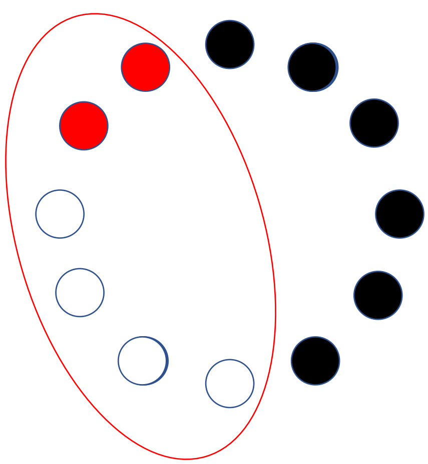
- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *And This one covers…*

And this: Energy = -3

43

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



2.1) Energy wells are not small

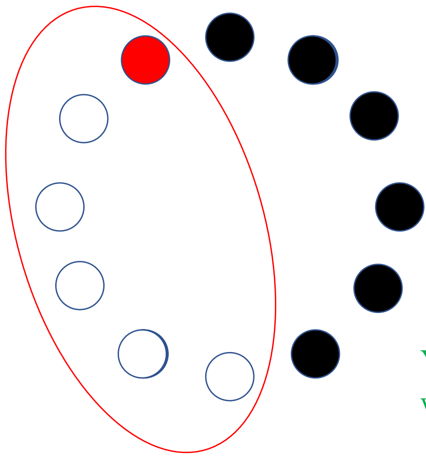
- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *And This one covers…*

And this: Energy = -1

44

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



2.1) Energy wells are not small

- Say you train this…
- … (which means you also trained this)…
- Then the two energy wells you have created *span the entire space of the Hopfield network!*
- *And This one covers…*

And this: Energy = 0

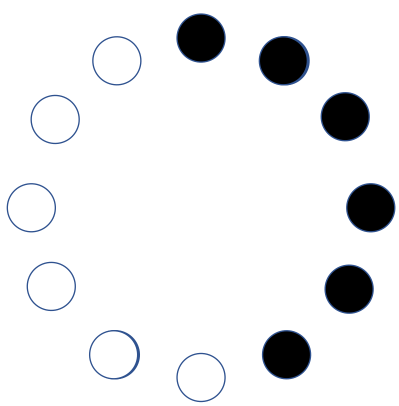
You trained *one* pattern and it created two energy wells that cover *all* the units in the network.

Every training pattern affects every unit in the network. (Oh my!)

45

And This Answers the Question, "Why Asynchronous Updating?"

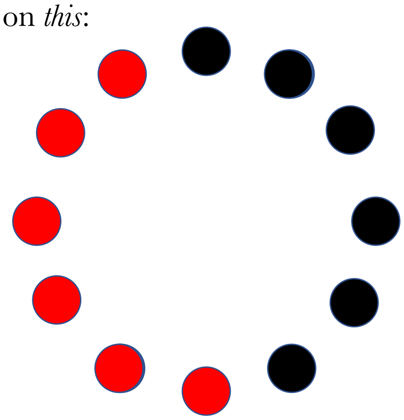
Say you've trained trained your network on this…



46

And This Answers the Question, "Why Asynchronous Updating?"

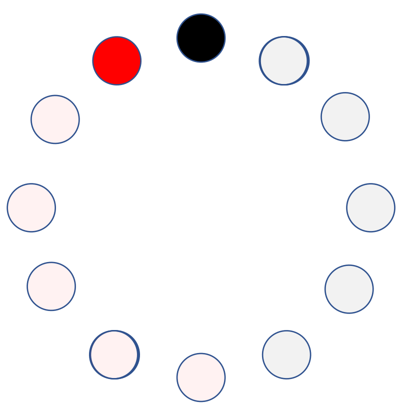
Say you've trained trained your network on this… which really means you've trained it on *this*:



47

And This Answers the Question, "Why Asynchronous Updating?"

And say you then test it with this starting pattern (training patterns are pale):



What is the network to do?

The "red route" and the "black route" are equally viable.

But it cannot take both.

It needs a way to choose.

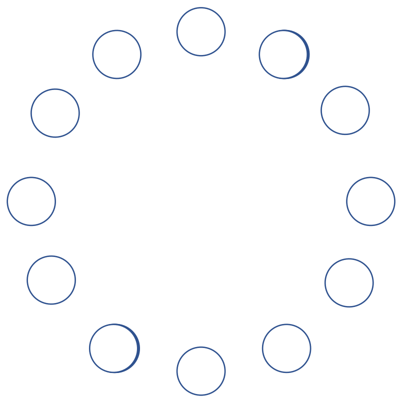
Randomly choosing which unit to update solves this problem.

Choose pink, you'll go red; choose gray, you'll go black.

48

Returning to the Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



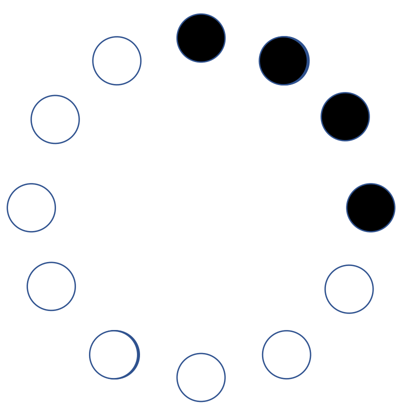
2.2) You haven't seen the half of it!

Say you train this…

49

The Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



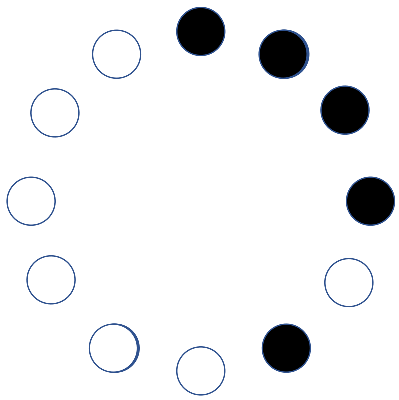
2.2) You haven't seen the half of it!

Say you train this…

50

The Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



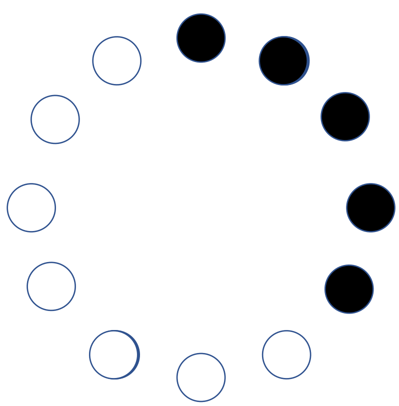
2.2) You haven't seen the half of it!

… and then this.

51

The Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



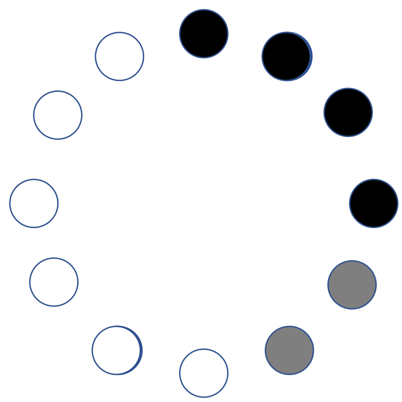
2.2) You haven't seen the half of it!

… and then this.

52

The Ugly Reality of the Energy Landscape

2) Energy Wells interact with one another in often unfortunate ways…



2.2) You haven't seen the half of it!

The network will learn this.

And worse.

53

Subtleties of the Hopfield Model

- **The model has stable limit points**
  - It tends to be attracted to specific states (energy wells)
  - These tend to reside close to trained states
  - Network finds stable states close to its starting state
- **Nonlinear activation function allow it to make choices**
  - Given a starting state that's 60% A and 40% B, it will tend to settle into A
- **Major emergent properties do not depend on the particulars**
  - **With asymmetric connections** (i.e., $w_{ij} \neq w_{ji}$):
    - Still settles into stable limit points
    - Probability of errors increases, but network still settles
    - Signal to noise ratio (correct to incorrect bits) decreases by about 1/root(2)
  - **With completely random connections**:
    - Network most often settled into a stable state
    - Sometimes entered limit cycle
    - Rarely entered chaotic wandering through small region of state space

54

Subtleties of the Hopfield Model

- **With random training patterns, storage capacity is about 12 – 15%** of the model's dimensionality
  - Training patterns interact in the energy landscape
  - Beyond about 12%, the interactions obscure the original patterns
    - Negative effect: Information loss
    - Positive effect: Generalization

- **With orthogonal training patterns, training capacity is higher**

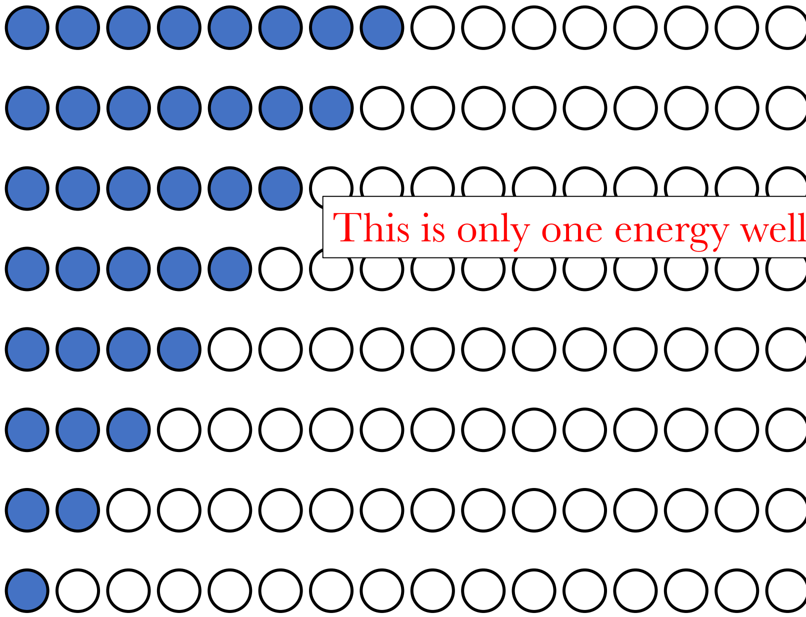
55

Training Capacity of the Hopfield Model

- **It's all about how energy wells interact**
- Recall that energy wells are *large*…

56

An energy well spans any pattern sharing at least one pair of active units with the vector at the center of the well



This is only one energy well!

Energy = - 28

Energy = - 21

Energy = - 15

Energy = - 10

Energy = - 6

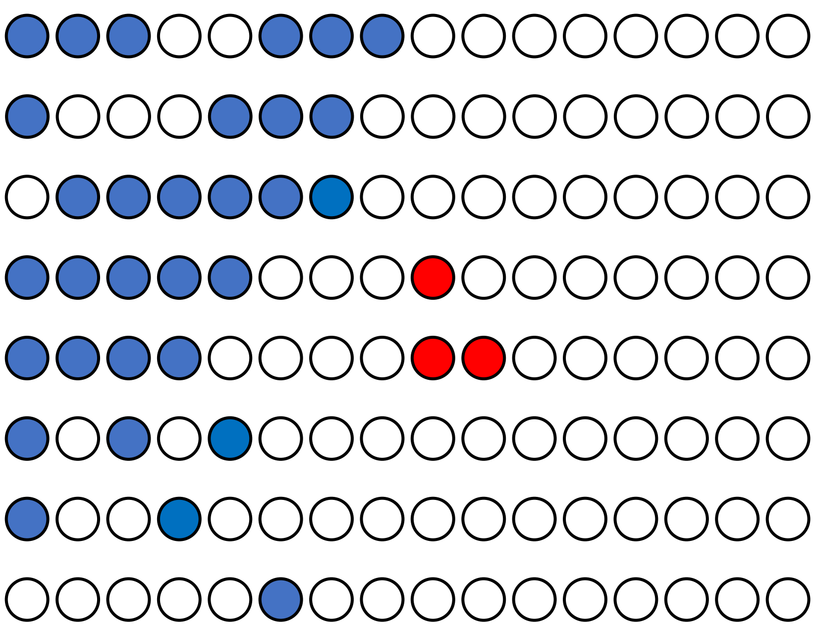
Energy = - 3

Energy = - 1

Energy = 0

57

It's even bigger than that. All the following are also in the well…



Energy = - 15

Energy = - 6

Energy = - 15

Energy = - 7.5

Energy = - 2

Energy = - 3

Energy = - 1

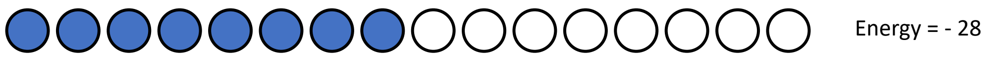
Energy = 0

58

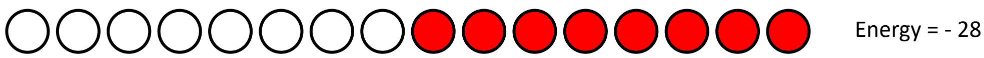
Training Capacity of the Hopfield Model

- **It's all about how energy wells interact**
- Recall that energy wells are *large*…
- … and recall that training "one" energy well is really training ***two***:

If you think you're training this:



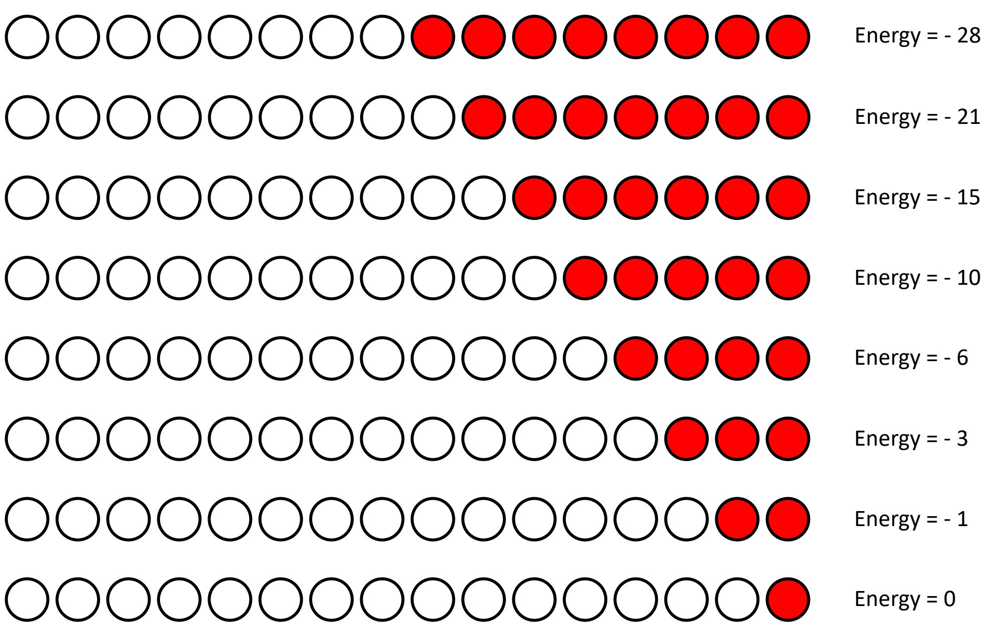
Then you're also training this:



With its own equally enormous energy well.

59

An energy well spans any pattern sharing at least one pair of active units with the vector at the center of the well

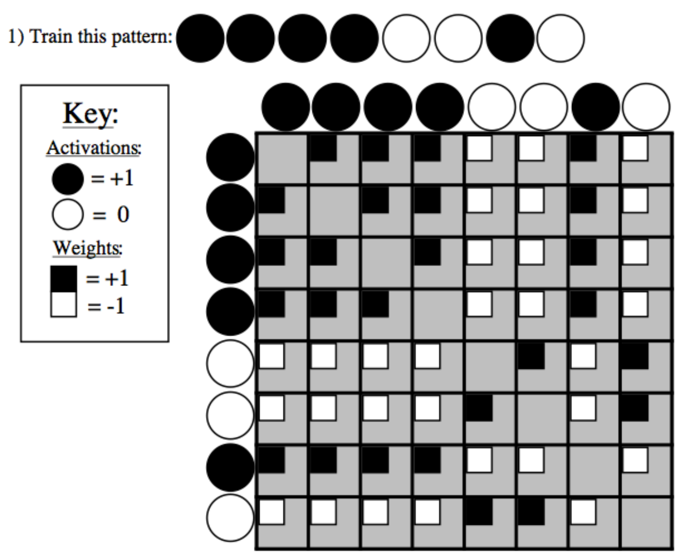


60

A slightly different look at energy wells
and
how training patterns interact

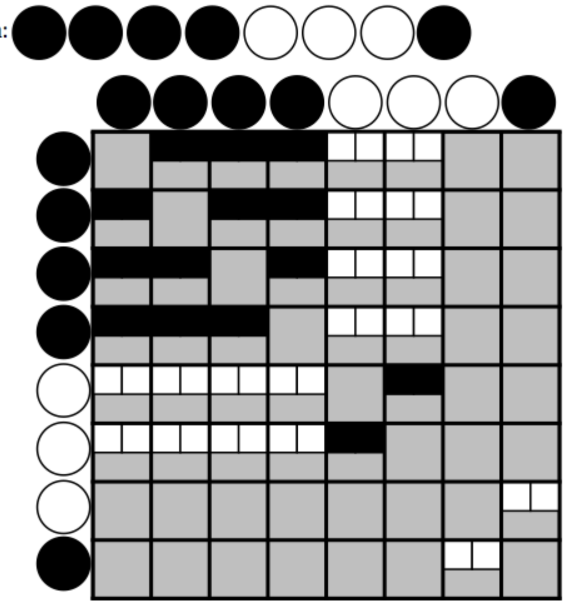
61

Interactions among training patterns in a Hopfield Network:
I. Prototype Learning



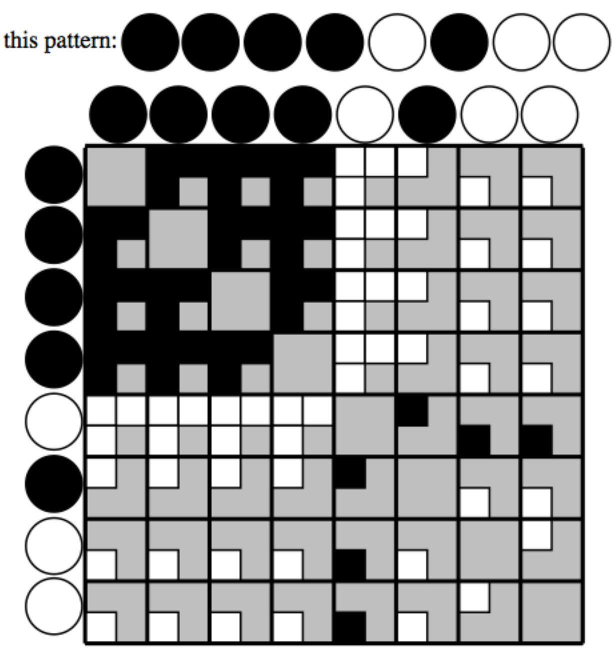
62

2) Train this pattern:



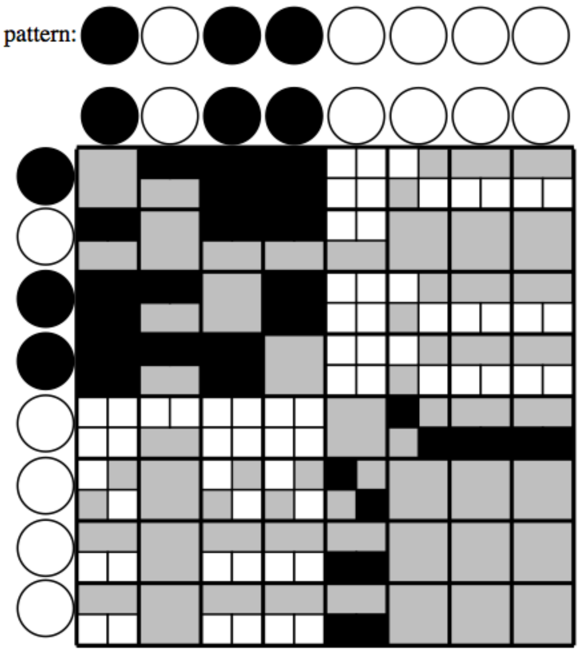
63

3) Train this pattern:



64

4) Train this pattern:

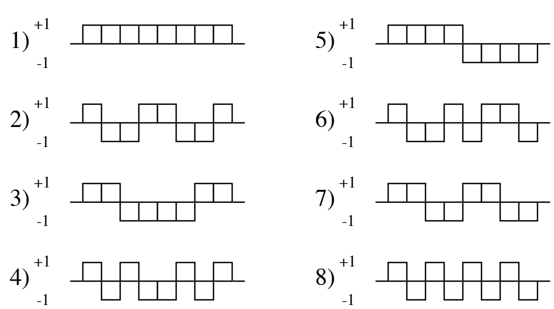


Note that units 1-4 all mutually excite one another and inhibit units 5-8: Any time one of 1-4 becomes active it will activate the rest.

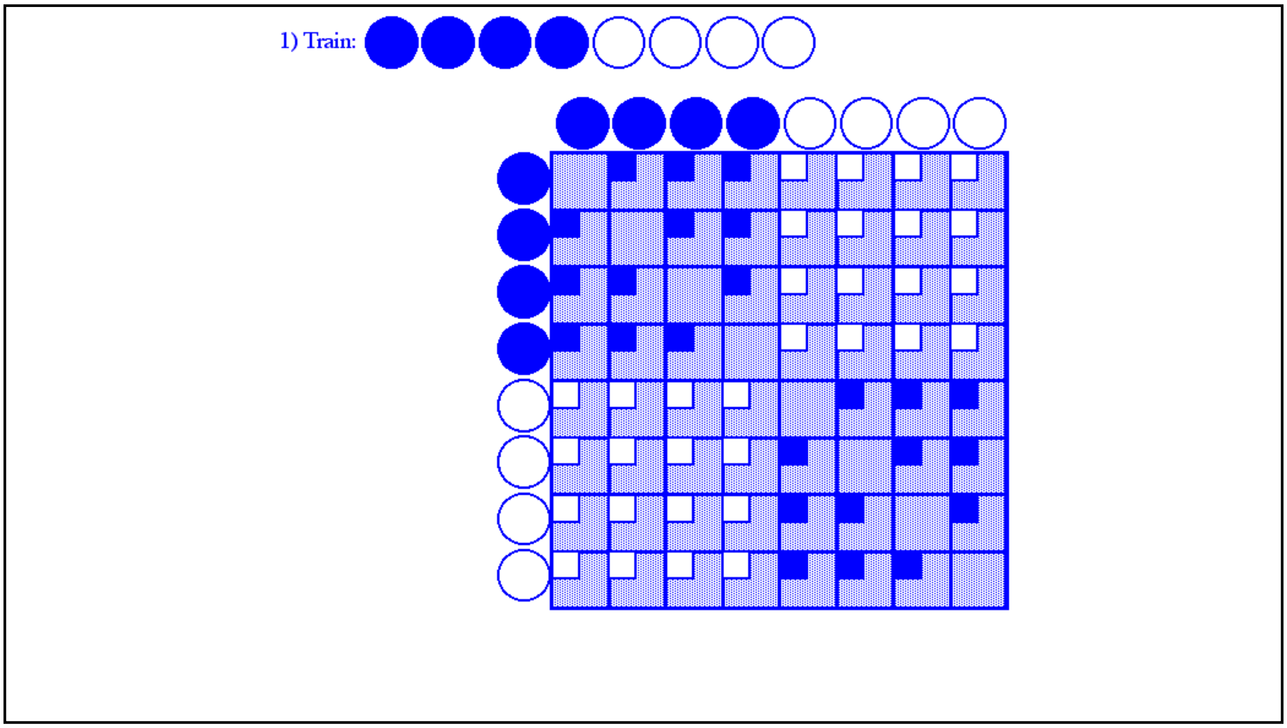
Note also that unit five excites and is excited by units 6-8. Whenever 5 becomes active, it will tend to activate 6-8. Likewise, whenever 6, 7 or 8 gets active, they wil activate 5, which will activate the rest.

65

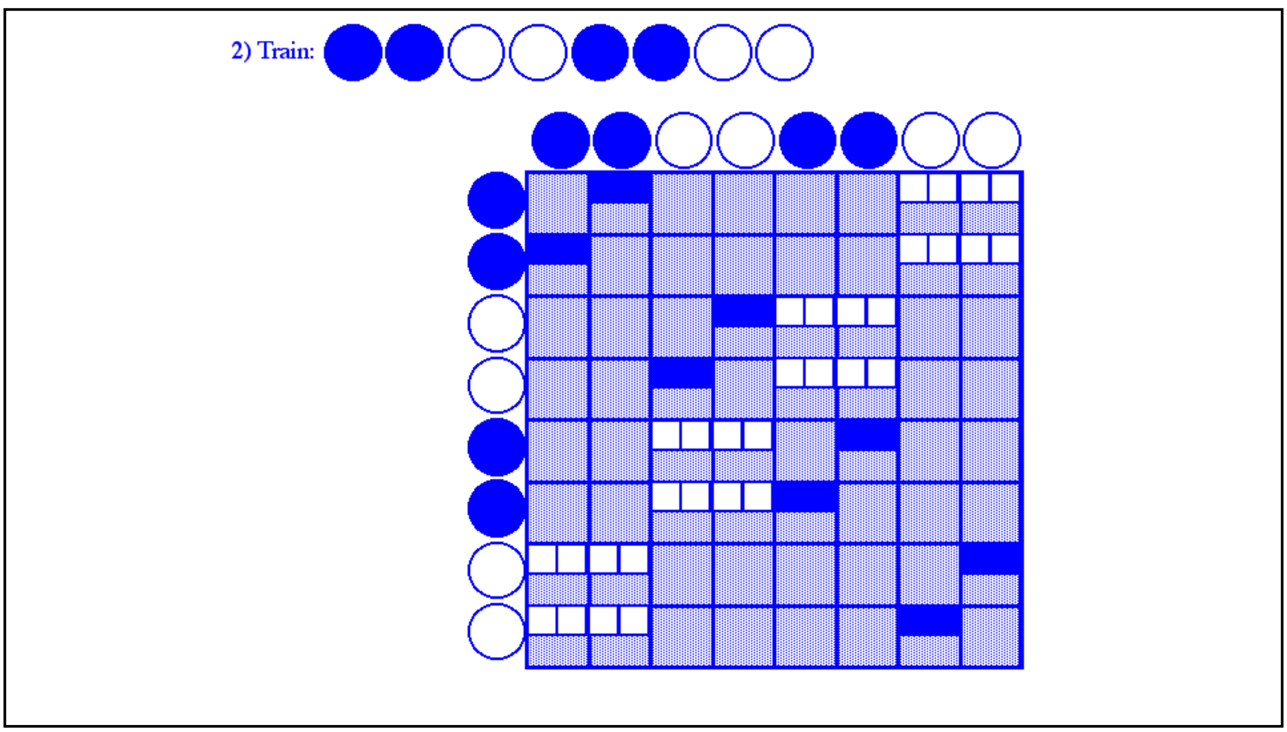
Walsh Functions



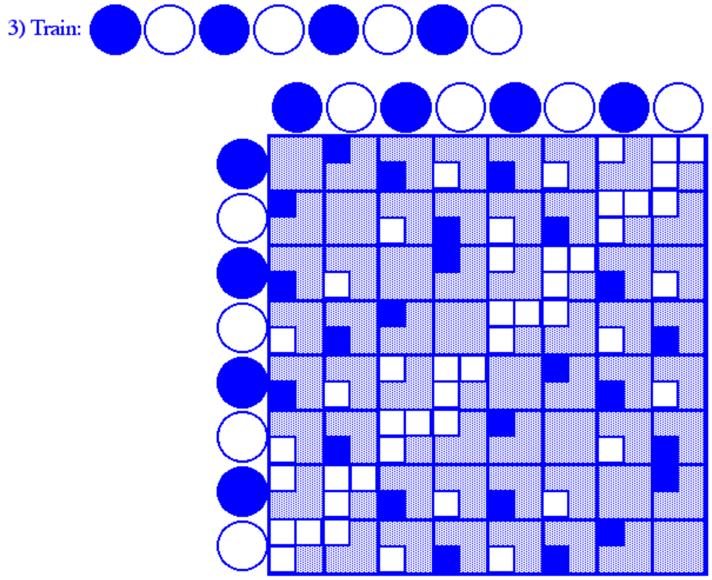
66



67

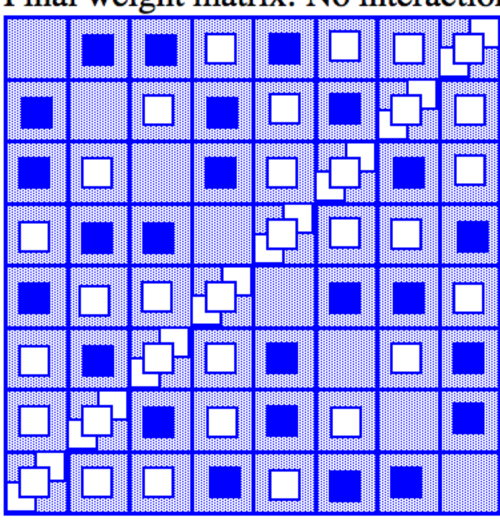


68



69

Final weight matrix: No interaction between patterns.



Total support from any one unit to any trained pattern is the same for all units and all patterns: for any unit, *u*, and any pattern *p*, five of u's connections support p (have positive values where p is active and negative values where p is inactive) and two oppose it. For example, **consider the first unit**:

For p= 11110000,
weights 2, 3, 6,7, 8 support p;
weights 4 and 5 opposes it.

For p = 11001100,
weights 2, 4, 5, 7 and 8 support p;
weights 3 and 6 opposes it.

For p= 10101010,
weights 3, 4, 5, 6, and 8 support p;
weights 2 and 7 oppose it.

A similar pattern holds for each and every unit. Based on the weights alone, every unit supports every pattern equally: This is minimal interaction between patterns.

70

Implementing a Hopfield Net

1. Create the network:
   - Create *n* units and connect every unit to every other unit but not itself
2. Train the network:
   - Create a set of training patterns. For each pattern in the set:
   - Present the pattern and update the weights
3. Test the network. For each test:
   - Create a *starting pattern*, e.g., by systematically distorting a training pattern
   - Impose the starting pattern on the network
   - Repeatedly update the network with asynchronous updating. On each iteration:
     - Update one unit, randomly chosen
     - Record the energy of the network
     - Check to see whether the network as settled
   - When settled: record the Hamming distance between final state and trained state

71

How Can You Tell When it Has Settled

- Updating is stochastic, so how can you know when it's done changing?
- **The hard way**:
  - Record whether it changes on each iteration
  - When it has gone unchanged for *s* iterations, "declare" it settled
  - Obvious, but time consuming and error prone
- **The easy way**:
  - Give each unit a *target activation* in addition to its *actual activation*
  - On each iteration:
    - Update *every* unit's input
    - Based on that, update every unit's *target* activation (this is *synchronous*)
    - Randomly choose one unit. Set *that* unit's activation to its target activation
  - When *every* unit's activation equals its target activation, the network has settled

72