1 Hopfield Network

1.1 Discrete-time Hopfield Network

1.1.1 Definition

Definition 1. *[Discrete-time Hopfield Network]*

Let $t \in \mathbb{N}$ *and* $x \in \{-1, +1\}^d$, $W \in \mathbb{R}^d \times \mathbb{R}^d$ *with* $W_{\alpha\beta} = W_{\beta\alpha}$ *and* $W_{\alpha\alpha} = 0$, *and* $b \in \mathbb{R}^d$. *Define discrete-time dynamics*

$$x^\alpha(t+1) = \mathrm{sign}(W^\alpha{}_\beta x^\beta(t) + b^\alpha).$$

The (W, b) *is called a discrete-time Hopfield network.*

1.1.2 Convergence

Lemma 2. *Let* (W, b) *a discrete-time Hopfield network. Define* $\mathcal{E}(x) := -(1/2) W_{\alpha\beta} x^\alpha x^\beta - b_\alpha x^\alpha$. *Then* $\mathcal{E}(x(t+1)) - \mathcal{E}(x(t)) \leqslant 0$.

Proof. Consider async-updation of Hopfield network, that is, change the component at dimension $\hat{\alpha}$, i.e. $x'_{\hat{\alpha}} = \mathrm{sign}[W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}}]$, then

$$\begin{aligned}\mathcal{E}(x') - \mathcal{E}(x) &= -\frac{1}{2} W_{\alpha\beta} x'^\alpha x'^\beta - b_\alpha x'^\alpha + \frac{1}{2} W_{\alpha\beta} x^\alpha x^\beta + b_\alpha x^\alpha \\ &= -2\,(x'^{\hat{\alpha}} - x^{\hat{\alpha}})\,(W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}}),\end{aligned}$$

which employs conditions $W_{\alpha\beta} = W_{\beta\alpha}$ and $W_{\alpha\alpha} = 0$. Next, we prove that, combining with $x'_{\hat{\alpha}} = \mathrm{sign}[W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}}]$, this implies $\mathcal{E}(x') - \mathcal{E}(x) \leqslant 0$.

If $(x'^{\hat{\alpha}} - x^{\hat{\alpha}}) > 0$, then $x'^{\hat{\alpha}} = 1$ and $x^{\hat{\alpha}} = -1$. Since $x'_{\hat{\alpha}} = \mathrm{sign}[W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}}]$, $W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}} > 0$. Then $\mathcal{E}(x') - \mathcal{E}(x) < 0$. Contrarily, if $(x'^{\hat{\alpha}} - x^{\hat{\alpha}}) < 0$, then $x'^{\hat{\alpha}} = -1$ and $x^{\hat{\alpha}} = 1$, implying $W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}} < 0$. Also $\mathcal{E}(x') - \mathcal{E}(x) < 0$. Otherwise, $\mathcal{E}(x') - \mathcal{E}(x) = 0$. So, we conclude $\mathcal{E}(x') - \mathcal{E}(x) \leqslant 0$. □

Theorem 3. *[Convergene of Discrete-time Hopfield Network] Let* (W, b) *a discrete-time Hopfield network. Then any trajectory obeying the update rule will converge either to a fixed point or a limit circle.*

Proof. Since the states of the network are finite, the $\mathcal{E}$ is lower bounded. □

1.1.3 Learning Rule

Let (W, b) a discrete-time Hopfield network. And $D := \{x_n | x_n \in \{-1, +1\}^d, n = 1, \ldots, N\}$ a dataset[1]. We can train the Hopfield nework by seeking a proper parameters (W, b), s.t. its stable points cover the dataset as much as possible, by[2]

Algorithm 1

```
W, b = init_W, init_b  # e.g. by Glorot initializer
for step in range(max_step):
    for x in dataset:
        y = f(W @ x + b)
```

1. We use Greek alphabet for component in $\mathbb{R}^d$ and Lattin alphabet for element in dataset.
2. This algorithm is the algorithm 42.9 of Mackay.

```
loss = norm(x - y)
optimizer.minimize(objective=loss, variables=(W, b))
W = set_zero_diag(symmetrize(W))
```

1.2 Continuous-time Hopfield Network

1.2.1 Definition

Definition 4. *[Continuous-time Hopfield Network]*

Let $t \in [0, +\infty)$ and $x \in [-1, +1]^d$, $W \in \mathbb{R}^d \times \mathbb{R}^d$ with $W_{\alpha\beta} = W_{\beta\alpha}$, and $b \in \mathbb{R}^d$. Define dynamics

$$\tau \frac{\mathrm{d}x^\alpha}{\mathrm{d}t}(t) = -x^\alpha(t) + f(W^\alpha{}_\beta x^\beta(t) + b^\alpha),$$

where $\tau \in (0, +\infty)$ a constant and $f: \mathbb{R} \to [-1, 1]$ being increasing. The $(W, b; \tau, f)$ is called a continuous-time Hopfield network.

Remark 5. With

$$\tau \frac{x^\alpha(t + \Delta t) - x^\alpha(t)}{\Delta t} = -x^\alpha(t) + f(W^\alpha{}_\beta x^\beta(t) + b^\alpha)).$$

Setting $\Delta t = \tau$ gives and $f(.) = \mathrm{sign}(.)$ gives

$$x^\alpha(t + \tau) = \mathrm{sign}(W^\alpha{}_\beta x^\beta(t) + b^\alpha),$$

which is the same as the discrete-time Hopfield network.

1.2.2 Convergence

Lemma 6. *Let $(W, b; \tau, f)$ a continous-time Hopfield network. Define $a^\alpha := W^\alpha{}_\beta x^\beta + b^\alpha$ and $y^\alpha := f(a^\alpha)$, then*

$$\mathcal{E}(y) := -\frac{1}{2} W_{\alpha\beta} y^\alpha y^\beta - b_\alpha y^\alpha + \sum_\alpha \int^{y^\alpha} f^{-1}(y^\alpha) \mathrm{d}y^\alpha.$$

Then $\mathcal{E}(y(x(t + \mathrm{d}t))) - \mathcal{E}(y(x(t))) \leqslant 0$.

Proof. The dynamics of a^α is

$$\begin{aligned}
\tau \frac{\mathrm{d}a^\alpha}{\mathrm{d}t} =& \tau W^\alpha{}_\beta \frac{\mathrm{d}x^\beta}{\mathrm{d}t} \\
=& W^\alpha{}_\beta [-x^\beta(t) + f(a^\beta)] \\
=& -(W^\alpha{}_\beta x^\beta(t) + b^\alpha) + b^\alpha + W^\alpha{}_\beta y^\beta \\
=& W^\alpha{}_\beta y^\beta + b^\alpha - a^\alpha.
\end{aligned}$$

Since W is symmetric, we have $\partial \mathcal{E} / \partial y^\alpha = -W_{\alpha\beta} y^\beta - b_\alpha + f^{-1}(y_\alpha)$. Then

$$\begin{aligned}
\frac{\mathrm{d}\mathcal{E}}{\mathrm{d}t} =& \frac{\mathrm{d}y^\alpha}{\mathrm{d}t} (-W_{\alpha\beta} y^\beta - b_\alpha + f^{-1}(y_\alpha)) \\
=& \frac{\mathrm{d}y^\alpha}{\mathrm{d}t} (-W_{\alpha\beta} y^\beta - b_\alpha + a_\alpha) \\
=& -\frac{\mathrm{d}y^\alpha}{\mathrm{d}t} (W_{\alpha\beta} y^\beta + b_\alpha - a_\alpha)
\end{aligned}$$

Notice that, the second term of rhs is exactly the dynamics of a_α, then

$$\begin{aligned}\frac{\mathrm{d}\mathcal{E}}{\mathrm{d}t} &= -\tau \frac{\mathrm{d}y^\alpha}{\mathrm{d}t}\frac{\mathrm{d}a_\alpha}{\mathrm{d}t} \\ &= -\tau \frac{\mathrm{d}y^\alpha}{\mathrm{d}a^\alpha}\left(\frac{\mathrm{d}a^\alpha}{\mathrm{d}t}\frac{\mathrm{d}a_\alpha}{\mathrm{d}t}\right) \\ &= -\tau f'(a^\alpha)\left(\frac{\mathrm{d}a^\alpha}{\mathrm{d}t}\frac{\mathrm{d}a_\alpha}{\mathrm{d}t}\right).\end{aligned}$$

Since f is increasing and $\tau > 0$, $\mathrm{d}\mathcal{E}/\mathrm{d}t \leqslant 0$. □

Remark 7. The condition $W_{\alpha\alpha} = 0$ for $\forall \alpha$ is not essential for this lemma. Indeed, this condition is absent in the proof. This differs from the case of discrete-time.

Theorem 8. *[Convergene of Continuous-time Hopfield Network] Let $(W, b; \tau, f)$ a continous-time Hopfield network. Then any trajectory along the dynamics will converge either to a fixed point or a limit circle.*

Proof. The function $E := \mathcal{E} \circ y$ is lower bounded since y, i.e. function $f: \mathbb{R} \to [-1, 1]$, is bounded. This E is a Lyapunov function for the continous-time Hopfield network. □

1.2.3 Learning Rule

Corollary 9. *Let $(W, b; \tau, f)$ a continous-time Hopfield network. And $D := \{x_n | x_n \in \mathbb{R}^d, n = 1, \ldots, N\}$ a dataset. If add constraint $W_{\alpha\alpha} = 0$ for $\forall \alpha$, then we can train the Hopfield nework by seeking a proper parameters (W, b), s.t. its stable points cover the dataset as much as possible, by*[3]

***Algorithm* 2**

```
W, b = init_W, init_b  # e.g. by Glorot initializer
for step in range(max_step):
    for x in dataset:
        y = f(W @ x + b)
        loss = norm(x - y)
        optimizer.minimize(objective=loss, variables=(W, b))
        W = set_zero_diag(symmetrize(W))
```

Proof. For $\forall x_n \in D$, we try to find (W, b), s.t. $\mathrm{d}x/\mathrm{d}t = 0$ at x_n, i.e.

$$x_n^\alpha = f(W^\alpha{}_\beta x_n^\beta + b^\alpha).$$

When $W_{\alpha\alpha} = 0$ for $\forall \alpha$, $f(W^\alpha{}_\beta x^\beta + b^\alpha)$ thus has no information of x^α, it has to predict the x^α by the interaction between x^α and the other x's components. □

Remark 10. This algorithm is equivalent to

Algorithm 3

```
dt = ...  # e.g. 0.1
W, b = init_W, init_b
```

3. This algorithm generalizes the algorithm 42.9 of Mackay.

```
for step in range(max_step):
    for x in dataset:
        # that is, compute x(dt), with x(0) = x
        y = ode_solve(f=lambda t, x: -x + f(W @ x + b), t0=0, t1=dt, x0=x)
        loss = norm(x - y)
        optimizer.minimize(objective=loss, variables=(W, b))
        W = set_zero_diag(symmetrize(W))
```

Indeed, trying to reach $y = x$ within a small interval will force x to be a fixed point.

1.2.4 Relation to Auto-encoder

Notice that at fixed point $x_\star$, $x_\star^\alpha = f(W^\alpha{}_\beta x_\star^\beta + b^\alpha)$, which is a single-layer auto-encoder. The learning rule is also simply the learning rule of single-layer auto-encoder.

1.2.5 Stability of Fixed Points

We study the stability of fixed points. Let $z^\alpha := W^\alpha{}_\beta x^\beta + b^\alpha$. Jacobian

$$\begin{aligned} J^\alpha{}_\beta =& \frac{\partial}{\partial x^\beta}(-x^\alpha + f(z^\alpha)) \\ =& -\delta^\alpha{}_\beta + f'(z^\alpha) W^\alpha{}_\beta. \end{aligned}$$

If $f(x) = \tanh(x)$, and at fixed point,

$$\begin{aligned} J^\alpha{}_\beta =& -\delta^\alpha{}_\beta + \frac{1}{2}(1 - f^2(z^\alpha)) W^\alpha{}_\beta \\ =& -\delta^\alpha{}_\beta + \frac{1}{2}(1 - x_\star^\alpha)(1 + x_\star^\alpha) W^\alpha{}_\beta. \end{aligned}$$

The eigen-value of J, $\lambda_J =: -1 + \lambda$, have

$$\det\left(\frac{1}{2}(1 - x_\star^\alpha)(1 + x_\star^\alpha) W^\alpha{}_\beta - \lambda \delta^\alpha{}_\beta\right) = 0$$

For instance, if $x_\star^\alpha \to \pm 1$ for $\forall \alpha$, that is $\|x_\star^2 - 1\| \ll 1$, then, because of the linearity of this equation, we will have $\lambda \ll 1$. In this case, $\lambda_J \approx -1 < 0$, indicating the stability of the fixed point $x_\star$.

2 Variations

2.1 Dense Associative Memories

Theorem 11. *Let $v \in \mathbb{R}^d$, $F \in C^1(\mathbb{R}^n, \mathbb{R})$, $W \in \mathbb{R}^n \times \mathbb{R}^d$, $b \in \mathbb{R}^n$, and $\tau > 0$. Define the dynamics*

$$\tau \frac{\mathrm{d}x}{\mathrm{d}t} = -\nabla E(x) = -x + W^T \cdot \nabla F(W \cdot x + b) + v.$$

If $\nabla F(.)$ is bounded, i.e. $\exists K > 0$ s.t. $\max_{x \in \mathbb{R}^n} \{\nabla F(x)\} < K$, then any trajectory along the dynamics will converge either to a fixed point or a limit circle.

Proof. Let $E(x) := \frac{1}{2} x_\alpha x^\alpha - v_\alpha x^\alpha - F(W^\alpha{}_\beta x^\beta + b^\alpha)$, then $\tau \mathrm{d}x / \mathrm{d}t = -\nabla E(x)$. The $-x$ term will dominate the $W^T \cdot \nabla F(W \cdot x + b)$ term for $\|x\| > K \|W\|$, thus converges. So E is a Lyapunov function of the dynamics. ☐

Example 12. Let $F(x) := \sum_\alpha \int^{x^\alpha} \sigma(s)\mathrm{d}s$, where σ is sigmoid function. Then

$$\tau \frac{\mathrm{d}x}{\mathrm{d}t} = -x + W^T \cdot \sigma(W \cdot x + b) + v.$$

This coincides with the form in ref 1.

Example 13. Let $F(x) := \beta^{-1} \ln(\beta \sum_\alpha \mathrm{e}^{x^\alpha})$, $b = 0$, and $v = 0$, then

$$\tau \frac{\mathrm{d}x}{\mathrm{d}t} = -x + W^T \cdot \mathrm{softmax}(\beta W \cdot x).$$

This coincides with the form in ref 2.

Example 14. Let $v_i := W_{i,\cdot}$, i.e. the ith row of the matrix W. Assume $\|v_i\| = 1$ for $\forall i = 1, \ldots, n$. Let $F(x) := \beta^{-1} \ln(\beta \sum_\alpha \mathrm{e}^{x^\alpha})$, and $v = 0$, then

$$\tau \frac{\mathrm{d}x^\alpha}{\mathrm{d}t} = -x^\alpha + \sum_i p_i v_i^\alpha,$$

where $z_i := v_i \cdot x + b_i$ and then $p^i := \exp(\beta z^i) / \sum_j \exp(\beta z^j)$. The $\{(p_i, v_i) | i = 1, \ldots, n\}$ forms a categorical distribution.

Lemma 15. *Assume example 14. The Jacobian of the dynamics is*

$$J^{\alpha\beta}(x) = -\delta^{\alpha\beta} + \mathrm{Cov}_{p(x)}(v^\alpha, v^\beta),$$

where $\mathrm{Cov}_p(\cdot, \cdot)$ *denotes the covariance given distribution* p.

Proof. Directly,

$$\begin{aligned}
J^{\alpha\beta} \equiv & \frac{\partial}{\partial x_\beta} \left(-x^\alpha + \sum_i v_i^\alpha p_i \right) \\
= & -\delta^{\alpha\beta} + \sum_{i,j} v_i^\alpha \frac{\partial p_i}{\partial z^j} \frac{\partial z^j}{\partial x_\beta} \\
= & -\delta^{\alpha\beta} + \beta \sum_{i,j} v_i^\alpha v_j^\beta (p_i \delta_{i,j} - p_i p_j) \\
= & -\delta^{\alpha\beta} + \beta \sum_i p_i v_i^\alpha v_i^\beta - \beta \left(\sum_i p_i v_i^\alpha \right) \left(\sum_j p_j v_j^\beta \right) \\
= & -\delta^{\alpha\beta} + \beta \mathbb{E}(v^\alpha v^\beta) - \beta \mathbb{E}(v^\alpha) \mathbb{E}(v^\beta) \\
= & -\delta^{\alpha\beta} + \beta \mathrm{Cov}_p(v^\alpha, v^\beta).
\end{aligned}$$

And notice that the only variable that depends on x is p. So we insert x and gain the result. $\square$

For instance, at fixed point $x = v_1$, $p = (1, 0, \ldots, 0)$. $\mathrm{Cov}_{p(v_1)}(v^\alpha, v^\beta) = v_1^\alpha v_1^\beta - v_1^\alpha v_1^\beta = 0$. So $J^{\alpha\beta} = -\delta^{\alpha\beta}$ is negative defined, indicating that the fixed point is stable.

2.2 Cellular Automa

TODO

2.3 Relation to Restricted Boltzmann Machine

Definition 16. *[Ristricted Boltzmann Machine] Let $W \in \mathbb{R}^L \times \mathbb{R}^A$, with $L < A$, $b \in \mathbb{R}^A$, and $v \in \mathbb{R}^L$. For $\forall x \in \{-1, +1\}^A$, define updation rule*

$$
\begin{aligned}
z_{t+1} &= \operatorname{sign}[W \cdot x_t + b];\\
x_{t+1} &= \operatorname{sign}[W^T \cdot z_{t+1} + v].
\end{aligned}
$$

We call (W, b, v) with this updation rule a ristricted Boltzmann machine.

Theorem 17. *Restricted Boltzmann machine is a special case of discrete-time Hopfield network. This, thus, ensures the convergence of restricted Boltzmann machine.*

Proof. Define $y := (z_1, ..., z_L, x_1, ..., x_A) \in \mathbb{R}^{L+A}$, $h := (b_1, ..., b_L, v_1, ..., v_A)$, and

$$U := \begin{pmatrix} \mathbb{0}_1 & W^T \\ W & \mathbb{0}_2 \end{pmatrix},$$

where $\mathbb{0}_1 \in \mathbb{R}^A \times \mathbb{R}^A$, $\mathbb{0}_2 \in \mathbb{R}^L \times \mathbb{R}^L$ are zero matrices. Then we have $U_{\alpha\beta} = U_{\beta\alpha}$ and $U_{\alpha\alpha} = 0$ for $\forall \alpha$, β, and the updation rule can be viewed as an async-updation of Hopfield network (U, h), which updates the first L components at each step of updation. □

2.3.1 Learning Rule

Let (W, b, v) a restricted Boltzmann machine. And $D := \{x_n | x_n \in \{-1, +1\}^d, n = 1, ..., N\}$ a dataset. We can train the restricted Boltzmann machine by seeking a proper parameters (W, b), s.t. its stable points cover the dataset as much as possible, by[4]

Algorithm 4

```
W, b, v = init_W, init_b, init_v  # e.g. by Glorot initializer
for step in range(max_step):
    for x in dataset:
        next_z = softsign(W @ x + b)
        next_x = softsign(transpose(W) @ next_z + v)
        loss = norm(x - next_x)
        optimizer.minimize(objective=loss, variables=(W, b, v))

@custom_gradient
def softsign(x, T=1e-0):
    y = sign(x)
    grad_fn = lambda x: (1 - tanh(x)) ** 2 / T
    return y, grad_fn
```

Remark 18. In this algorithm, we use a specially designed softsign function instead of using tanh. The reason is that the output y is binary in this case, which then ceasing the difficulty of learning. Indeed, when $x = 1$, $y = 1$ using softsign is much simpler than $y = 0.1$ using tanh, reflected in the loss. This improves the capacity of re-construction with the same capacity of network. Numerical experiments confirm this remark.

4. This algorithm generalizes the algorithm 42.9 of Mackay.

3 References

1. On autoencoder scoring.

2. Hopfield networks is All You Need.

3. Information Theory, Inference, and Learning Algorithms, D. Mackay.