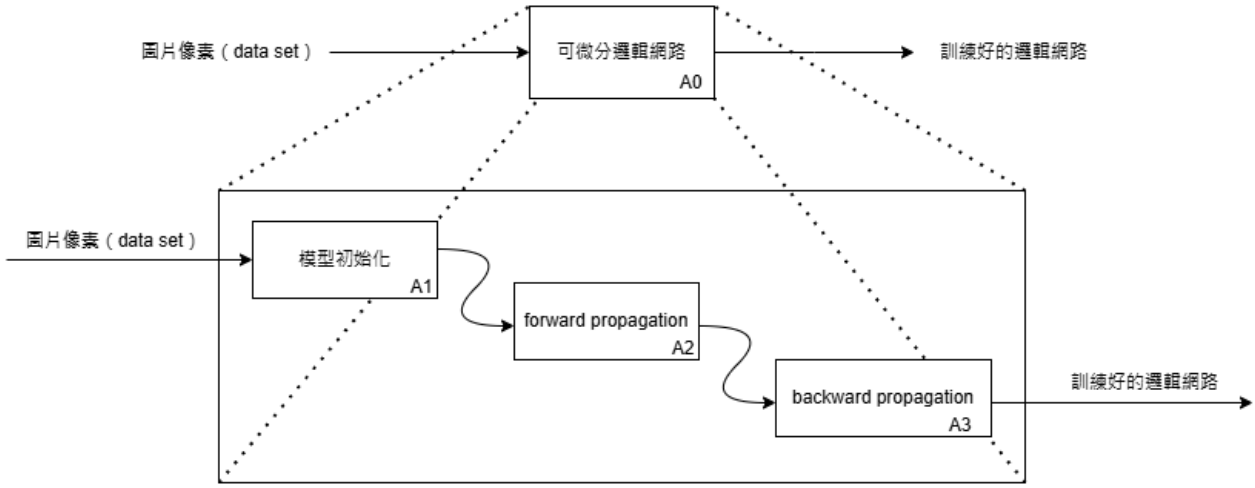
Differentiable Logic Network

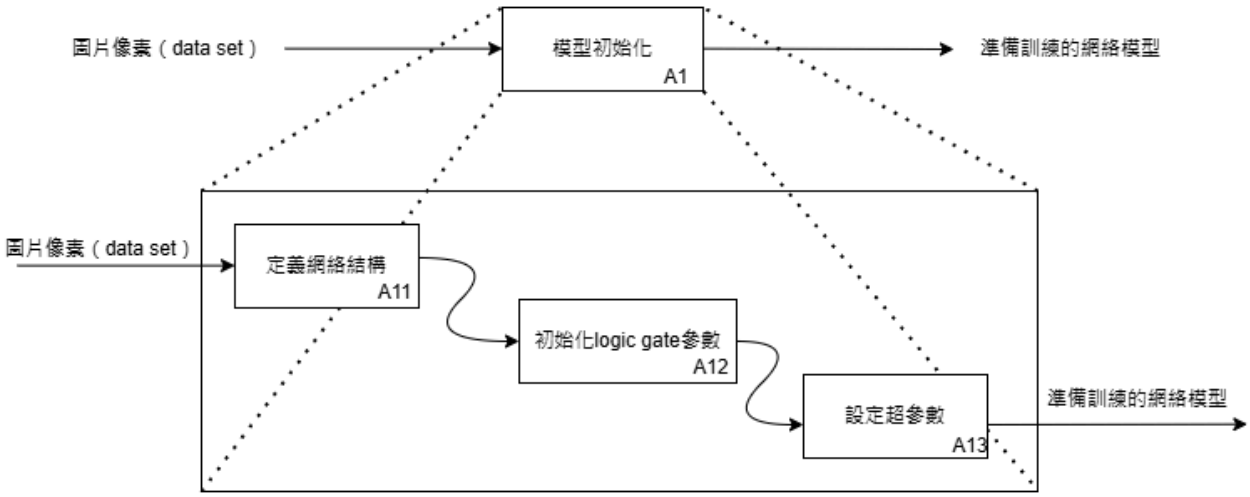
Differentiable Logic Network 深入研究: hackmd (https://hackmd.io/@gary7102/BJX5IjTzkg)

IDEF0 可微分邏輯網路 系統階層式模組化架構

基本設計：一共包含三大部分，分別是「模型初始化」、「fordward propagation」和「backward propagation」，以圖片分類為例，
輸入：圖片像素；輸出：訓練好的邏輯網路
注意，A2和A3為訓練過程，通常會經過多次的迭代，而非單一過程



模型初始化(A1):



定義網路結構(A11):

- 確定網絡的總層數 L（例如 4 到 8 層）
- 固定每層的神經元數量，通常相等
- 每層的每個神經元與前一層的**兩個輸入**隨機連接

初始化Logic gate 參數(A12):

- 每個神經元都對應 16 種邏輯操作（如 AND、OR、XOR 等）
- 使用 Softmax 對 w_i 初始化每個logic gate被選擇機率(p_i)：

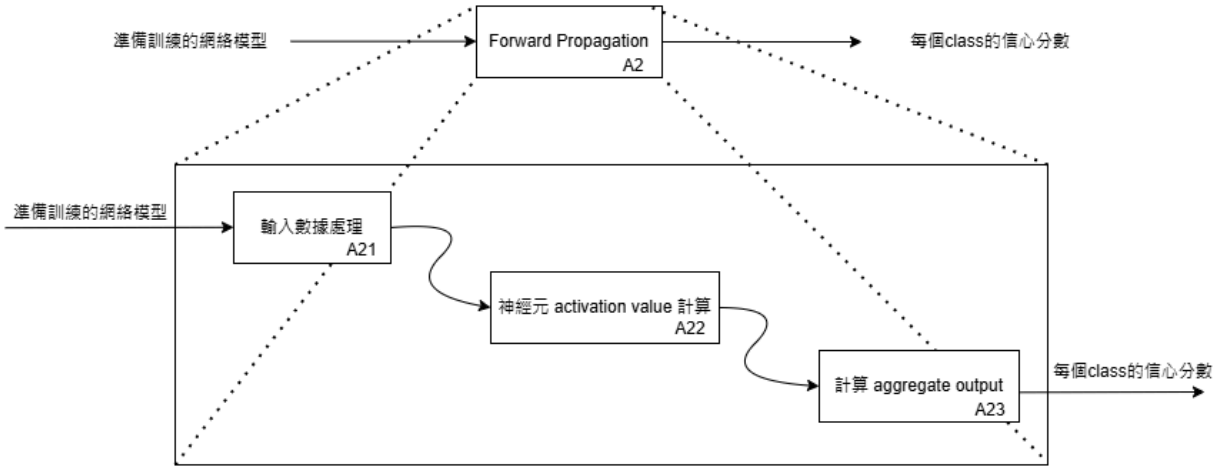
$$p_i = \frac{e^{w_i}}{\sum_{j=0}^{15} e^{w_j}}$$

w_i 是每個logic operation被選擇的優先程度，初始為常態分布隨機抽樣，代表每個logic gate的機率分布是均勻的，w_i 會在Backward Propagation中被更新，目的就是要改變每個神經元對Logic ate的選擇機率

定義超參數(A13):

- 初始化output layer之分組：將輸出層神經元分為 k 組(if k classes)，會在forward propagation 的aggregate output中使用到
- 定義learning rate, loss function (Softmax Cross-Entropy Loss)

Forward Propagation(A2):



輸入數據處理(A21):
若輸入是連續數據（如圖像像素值 $[0, 255]$），則進行nomilization，使$a \in [0, 1]$:

$$a = \frac{\text{輸入像素值}}{255}$$

神經元activation value 計算(A22):
每個神經元接受兩個輸入，假設為 a_1, a_2 (如上提到的a)，並計算所有logic gate的加權期望值：

$$a' = \sum_{i=0}^{15} \boldsymbol{p}_i \cdot f_i(a_1, a_2) = \sum_{i=0}^{15} \frac{e^{\boldsymbol{w}_i}}{\sum_j e^{\boldsymbol{w}_j}} \cdot f_i(a_1, a_2)\,.$$

p_i 即是上面提到的每個logic gate被選擇機率，使用softmax算得

$f_i(a_1, a_2)$ 為輸入a_1, a_2，第i個logic operation 的輸出

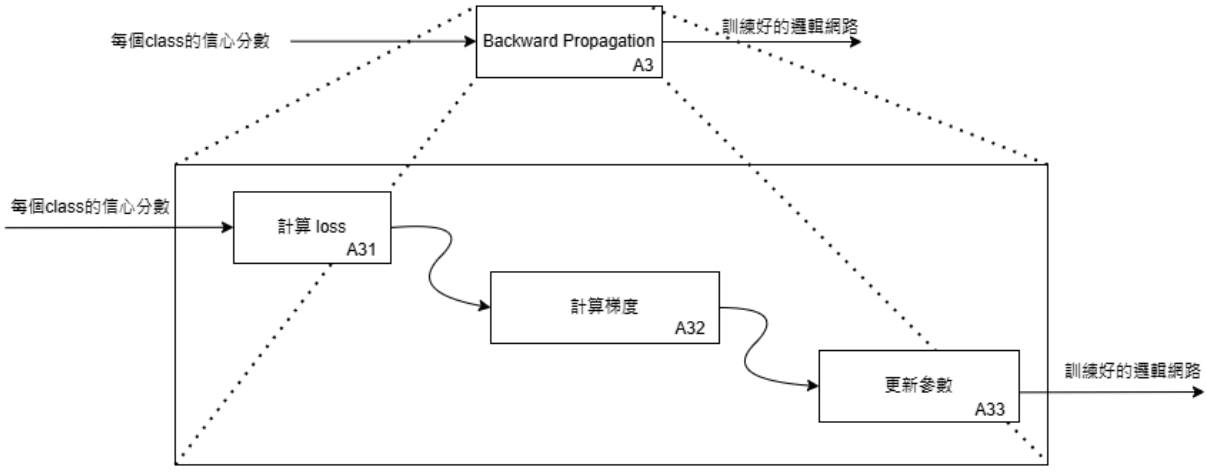
計算aggregate output(A23):

將output layer的 n 個神經元分成 k 組(if k classes)，每組 $\frac{n}{k}$ 個神經元，計算每個class的aggregate output:

$$\hat{y}_i = \sum_{j=i\cdot n/k+1}^{(i+1)\cdot n/k} a_j \,/\, \tau + \beta$$

- $\hat{y}_i$: class i 的信心分數
- a_j : output layer中第 j 個神經元的activation value
- τ 及 β : normalization value 及 offset

Backward Propagation(A3):



計算loss(A31):

Loss function: Softmax Cross-Entropy Loss，計算模型的的預測機率(q_i)相對於真實目標(t_i)的loss:

先對每個class 的aggregate output ($\hat{y}_i$) 求softmax，得 q_i:

$$q_i = \frac{e^{\hat{y}_i}}{\sum_i^k e^{\hat{y}_i}}$$

再把 q_i 代入cross entropy loss，得 L:

$$L = -\sum_i t_i * log(q_i)$$

計算梯度(A32):

計算loss對logic gate參數 w_i 的梯度:

$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial x_1} * \ldots * \frac{\partial x_i}{\partial w_i}$$

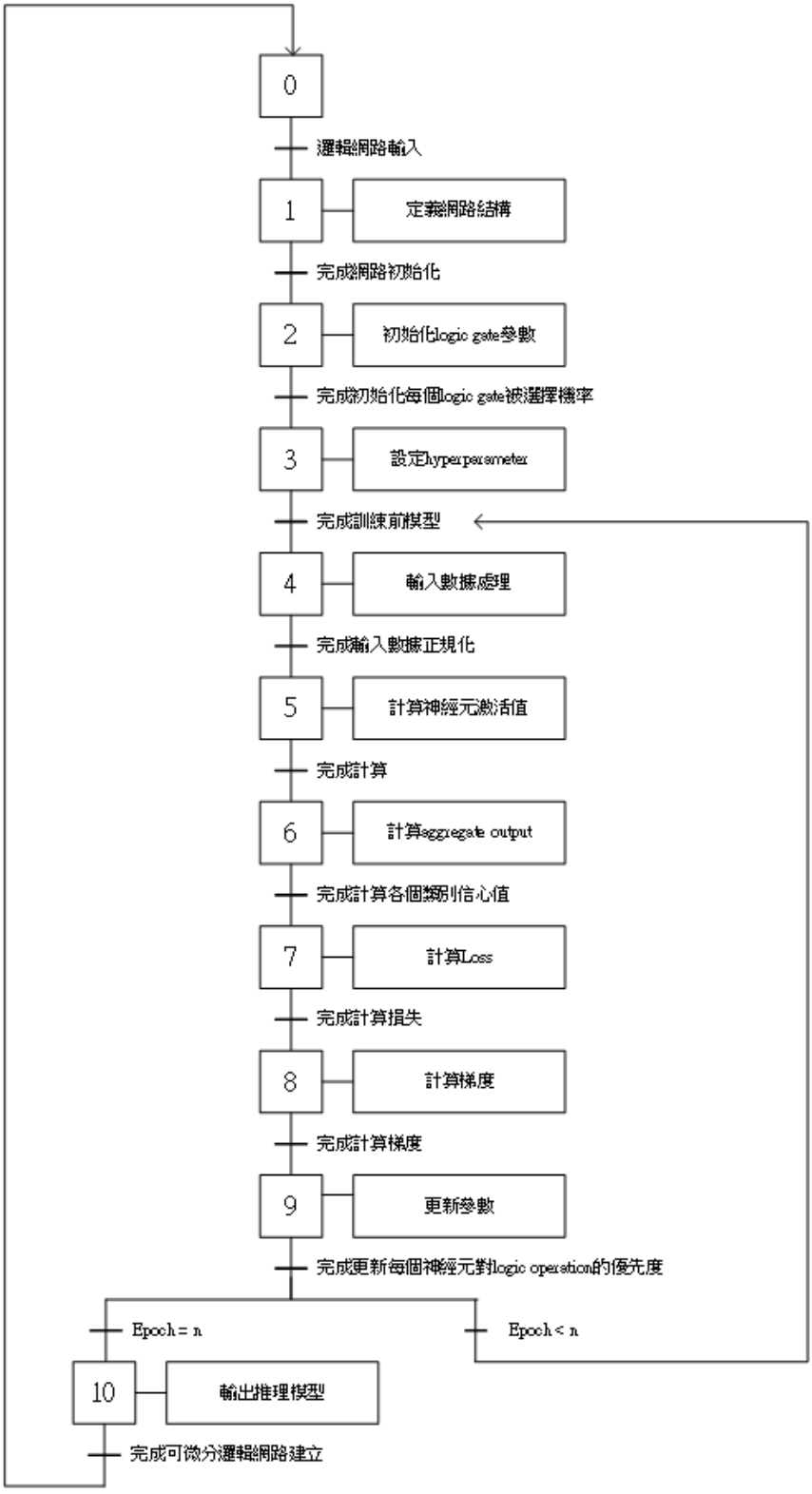
更新參數(A33):

更新 w_i，$\forall i$ = ith logic operation:

$$w_i = w_i - \eta * \frac{\partial L}{\partial w_i}$$

訓練階段多次迭代Foward Propagationr及Backward Prapagation 來更新 w_i ，並且更新後的 w_i 透過Softmax來更新 p_i(選取每個logic opeartion 的機率)，
使最適合的 p_i 最大化，進而在推理階段時讓每個神經元選擇最適合的logic operation

Grafcet 離散事件模型


0
邏輯網路輸入
1
定義網路結構
完成網路初始化
2
初始化logic gate參數
完成初始化每個logic gate被選擇機率
3
設定hyperparameter
完成訓練前模型
4
輸入數據處理
完成輸入數據正規化
5
計算神經元激活值
完成計算
6
計算aggregate output
完成計算各個類別信心值
7
計算Loss
完成計算損失
8
計算梯度
完成計算梯度
9
更新參數
完成更新每個神經元對logic operation的優先度
Epoch = n
Epoch < n
10
輸出推理模型
完成可微分邏輯網路建立