Chapter 4: The Hypernetwork Model of Memory

4.1 Hypergraphs
4.2 Hypernetwork Memory
4.3 Encoding and Decoding
4.4 Discussion: Localist vs. Globalist Models

4.3 Encoding and Decoding: Basic Algorithms

하이퍼망을 연상메모리로 사용하기 위해서는 정보를 저장(\textit{encoding})하고 저장된 정보를 인출(\textit{decoding})하는 과정이 필요하다. 정보를 저장하는 과정은 일반적으로 학습에 의해서 이루어지며 이를 위한 구체적인 알고리즘은 6장에서 살펴보기로 한다. 여기서는 기본적인 코딩과 디코딩 방법에 대해서 논한다.

\begin{align*}
D &= \{(x, y)\} \quad (1.1) \\
\text{하이퍼망 코딩의 목적은} \ D \text{를 이용하여 다음과 같은 함수} \ f \text{를 집합하는 파라매터 집합} \ \theta \text{를 가지는 하이퍼망} \ H = (X, E, W) \text{를 구성하는 것이다.}
\end{align*}
앞에서도 언급한 바와 같이 $\theta$는 하이퍼망의 구조, 예산에 대한 potential function의 모양, 결합 가중치의 값 등을 포함한다. 여기서 $\theta$의 값은 미세 조정하는 과정을 포함하지 않은 기본적인 코딩 절차를 기술한다. 보다 세련된 학습 알고리즘은 다음 장에서 고려한다. 간단한 코딩 절차만을 이용한 정보저장 과정도 product potential function $\varphi_{E_t}(x) = \prod_{i,j \in E_t} x_i x_j$와 같이 미세한 가중치 조정을 필요로 하지 않는 하이퍼망 모델의 경우에는 유용하게 사용될 수 있다. 코딩 절차를 요약하면 다음과 같다.

디코딩(저장) 알고리즘
0. Initialization: $E \leftarrow \emptyset, W \leftarrow \emptyset, H_0 \leftarrow (X, \emptyset, \emptyset)$
1. Get a training example $(x, y) \in D$
2. Generate hyperedges $E_i$ from $(x, y)$ to make $(x, y)$ and $t(t)$ $E_t W_t$
3. Update the edge set: $E \leftarrow E \cup E_t, W \leftarrow W + W_t$
   This results in a new hypernetwork $H_t(X, E, W)$
4. Go to Step 1.

하이퍼망 디코딩의 목적은 주어진 질의 입력 $x'$에 대해 대응되는 출력값 $f_{\theta}(x') = y'$를 생성하는 것이다. 이를 위해 먼저 partial hypernetwork의 개념을 도입한다. 즉 하이퍼망 $H = (X, E, W)$에 대해 패밀리 $F \subset E$에 의해서 생성된 하이퍼망 $(X_F, E_F, W_F)$를 $H$의 일부(partial) 하이퍼망이라고 한다. 여기서 $X_F$와 $W_F$는 다음과 같다.

$$X_F = \bigcup_{E_i \in F} E_i$$

$$W_F = \bigcup_{W_i \in F} W_i$$

디코딩 절차는 두 개의 하이퍼망을 매칭하는 것으로 볼 수 있다. 즉 질의 입력 $x'$로부터 앞의 코딩절차를 사용하여 질의 하이퍼망 $H' = (X', E', W')$를 생성한 후 이를 저장된 하이퍼망 $H = (X, E, W)$와의 매칭되는 일부 하이퍼망(partial hypernetwork) $H^*$를 다음과 같이 생성한다.

$$H^* = (X', E', W')$$

$$X^* = X' \cap X$$

$$E^* = E' \cap E$$

$$W^* = W' \cap W$$

이제 $H^*$로부터 대응하는 출력 $y'$을 생성한다. 하이퍼망이 $x$와 $y$의 결합 확률분포 $P(x,y)$를 표현하고 있기 때문에 $H^*$로부터 만들어진 확률분포 $P_{H^*}(x,y)$로부터 다음과 같이 $P(y|x)$를 계산할 수 있다.
$P(y \mid x) = \frac{P(x, y)}{P(x)} \approx \frac{P_{H^*}(x, y)}{P(x)} = f_{H^*}(x)$ \hspace{1cm} (1.6)

디코딩 질문을 요약하면 다음과 같다.

디코딩(인출) 알고리즘

0. Let $H$ be a learned hypernetwork.

Start with an empty (query) hypernet $H^* = (X', E', W') = (\emptyset, \emptyset, \emptyset)$.

1. Get a query example $(x', y')$.

2. Generate hyperedges $E'_i$ and associated $W'_i$ from $(x', y')$ to make $E'$ and $W'$.

3. This results in a query hypernetwork $H^* = (X', E', W')$, where $X' = \bigcup_{E_i \subseteq E} E'_i$.

4. Construct the partial hypernetwork $H^*$ from $H$ and $H^*$ as follows:

$$H^* = H \cap H^* = (X, E, W) \cap (X', E', W')$$.

5. Return $y^*$ as the predicted value of $y'$ by $f_{H^*}(x')$ using hypernetwork $H^*$.

Example: Movie Transcript Data

Statistics for Movie Transcripts

- 총 문장 수: 149663, 평균 문장 길이(단어): 6.24, 총 단어 수: 23719

Logical-conjunction potential function:
\[
\varphi_F(x) = \prod_{x_j \in E} x_j = x_1 x_2 \cdots x_{|E|}
\]

**Completion (Recall) Model (Unsupervised Learning)**

\[
P(x) = \frac{1}{Z} \exp\left( - \sum_{i=1}^{|E|} w_i \varphi_{E_i}(x) \right) = \frac{\exp\left( - \sum_{i=1}^{|E|} w_i \varphi_{E_i}(x) \right)}{\sum_{x'} \exp\left( - \sum_{i=1}^{|E|} w_i \varphi_{E_i}(x') \right)}
\]

**Classification (Recognition) Model (Supervised Learning)**

\[
f(x) = \sum_{i=1}^{|E|} w_i \varphi_{E_i}(x) = \sum_{i=1}^{|E|} w_i \prod_{x_j \in E} x_j
\]

**The probabilistic library as a hypernetwork**

- The whole library = \(H = (X, E, W)\)
- Vocabulary (i.e., words) = \(X\)
- Set of hyperedges = \(E\)
- Frequency of hyperedges = \(W\)
- Cardinality of hyperedges = \(\varphi_E()\)

**Example:** Space complexity (the size of hypernetwork) for binary variables

다음과 같은 10-bit 이진 함수 \(F\)를 생각하자.

\[
\begin{array}{c|c}
X & Y \\
\hline
x^{(0)} = 0000000000 & y^{(0)} = 0 \\
x^{(1)} = 0000000001 & y^{(1)} = 1 \\
\vdots & \vdots \\
x^{(1023)} = 1111111111 & y^{(1023)} = 1 \\
\end{array}
\]

총 \(2^n\) 개

\[
N = \sum_{k=0}^{n} \binom{n}{k} = 2^n
\]

\(n=10\)인 경우 \(N = 2^{10} = 1024\)이다.

위의 함수를 학습하는 한가지 방법은 10 bit 길이의 예제들을 모두 저장하는 것이다. 그러나
이 방법은 관측한 예제들을 그대로 저장할 뿐 예제들간의 일반적인 특성을 파악하지는 못한
다.
하이퍼망은 10 차원의 예제들을 저장하는 대신 이를 저차원의 조각으로 나누어 이들의 조합으로 목표함수를 나타내려 한다. 이 저차원의 조각이 하나의 에지가 되며 이들의 결합은 하나의 하이퍼망을 구성한다. 형식화하면, 우 원소(이진 변수)의 수가 \( n \)인 집합 \( X \)는 다음과 같이 구성된다.

\[
H = (X, E, W) \\
X = \{x_1, x_2, \ldots, x_n\} \\
x_i \in \{0, 1\} \\
n = 10
\]

예지의 집합은 집합 \( X \)의 멱집합이 되며 그 크기는 다음과 같다.

\[
E = Pow(X) \\
|E| = \sum_{k=0}^{n} \binom{n}{k} = 2^n = 2^{|X|}
\]

예지의 종류 및 그 개수를 구체적으로 나열해 보면 다음과 같다.

\[
\emptyset \} \{n\} = 1 \\
x_1 \} \{n\} = n = 10 \\
x_2 \} \{n\} = 2 \cdot 9 = 18 \\
\vdots \\
x_{n-1}x_n \} \{n\} = \frac{n!}{2(n-2)!} = \frac{n(n-1)}{2} = 10 \cdot 9 = 90 \\
x_{n-2}x_{n-1}x_n \} \{n\} = \frac{n!}{3(n-3)!} = \frac{n(n-1)(n-2)}{3 \cdot 2 \cdot 1} = 10 \cdot 9 \cdot 8 = 720 \\
\vdots \\
x_1x_2x_3\ldots x_n \} \{n\} = 1
\]

특정한 하나의 함수는 집합 \( X \)의 멱집합에 대한 멱집합의 한 원소로 정의된다. 따라서 정의할 수 있는 모든 함수의 종류는 다음과 같다.

\[
L = \sum_{k=0}^{n} \binom{2^n}{k} = 2^{2^n}
\]

\( n=10 \)인 경우 \( L = 2^{1024} \)가 된다. 주어진 훈련 데이터로부터 올바른 목표 함수를 저장하기 위해서는 크기 \( L \)인 공간을 탐색해야 한다. 하이퍼망은 \( X \)의 하이퍼예지의 조합 즉 멱집합의 원소들의 조합을 통해서 이 공간을 탐색하는 한 가지 방법이다.

**Example:** Space complexity (hypernetwork size) for movie transcript data
영화 자막 문장의 경우 각각의 변수 $x_i$는 어휘집합 $V$로부터 하나의 단어를 선택한다. 알맞게의 크기 $|V|=30000$이고 문장의 길이가 $n=10$이라고 하자. 즉

$$X = \{x_1, x_2, \ldots, x_n\}$$

$$x_i \in V$$

$$n=10$$

$$|V|=30000$$

10단어로 구성된 문장을 생성하기 위해서는 10개의 점을 갖고 각 점이 $|V|=30000$ 중 하나의 단어를 갖는 하이퍼äche만을 구성할 수 있다. 이 망에 포함될 수 있는 가능한 하이퍼에지의 수는 다음과 같다.

$$|E| = |V^{|X|} = 30000^{10} = 3 \cdot 10^{60}$$

이 숫자는 인간의 뇌에 존재하는 뉴런의 개수 10^11개나 시냅스의 개수 10^14개와 비교해 볼 때 어마어마하게 큰 수이다. 그러나 언어 사용자들은 이 정도의 어휘와 문장에 쉽게 구사한다. 효율적인 메모리가 되기 위해서는 이 수보다도 높은 수의 에지를 가지 하이퍼면이 구성되어야 할 것이다. 즉 메모리를 표현하는 코드가 상당히 sparse하게 될 것이다.

### 4.4 Discussion: Localist vs. Globalist Models

사람의 얼굴 사진을 적은 $N$개의 이진값 이미지 데이터 $D = \{I_1, I_2, \ldots, I_N\}$가 주어졌다고 하자. 여기서 $I_i = \{x_1, x_2, \ldots, x_n\}$이다. 이를 저장하였다면 연상기억하는 메모리를 생각해 보자. 한 가지 방법은 지금의 컴퓨터 메모리처럼 $N$개의 메모리를 할당하여 각 $i$번에 이미지 $I_i$를 그대로 저장하는 것이다.

$$M = \{I_1, I_2, \ldots, I_N\}$$

그러나 이러한 방식은 관측된 메모리를 정확히 재현하는 것은 가능하나 주소를 주어야 하며 따라서 연상작용이나 내용기반의 검색(검색하고자 하는 사진과 비슷한 사진을 입력으로 제공)을 수행할 수 없다. 이러한 극단적인 localist적 방안은 정확도는 높지만 압축률이 저조하다. 실제로 뇌에서는 이러한 localist적 정보 저장 방식을 사용하지 않는다.\(^1\)

위와 같은 극단적인 localist적 기억 모델의 단점을 극복하는 한 가지 방법은 데이터를 명시적으로 저장하지 않고 그 데이터의 개별값만을 저장하는 것이다. 예를 들어, 10000개의 화소에 대응되는 메모리 요소를 할당한 후 각 화소에 대한 통계적인 정보 예를 들어 $N$개의 이미지에 대한 그 화소의 1의 개수의 평균값을 저장하는 것이다.

$$M = \{\bar{I}_1, \bar{I}_2, \ldots, \bar{I}_n\}$$

그러나 이러한 방식은 모든 이미지에 대해 하나의 대표 이미지를 저장하는 것과 같으며 각각의 화소점이 상호 독립적이 경우에는 잘 동작하나 화소간에 상호 의존적인 얼굴 인식을

---

\(^1\) 아마도 물리적으로 뇌가 이러한 데이터베이스 방식의 정확한 정보 저장 및 검색에 적합하지 않은 장치인 것도 한 가지 이유일 것이다. 이것은 다시 뇌의 전화 과정과 관계가 있을 것이다.
위에서의 적합하지 않다. 이러한 극단적인 globalist적인 모델은 압축률이 높지만 정보 손실이 너무 많다.

위의 두 가지 극단적인 방법의 중간에 여러가지 기역 모델이 존재할 수 있다. 최근에 기계학습 분야에서 많이 연구되고 있는 커널 방법은 그 예 중의 하나이다(이에 대한 보다 상세한 논의는 7장 참조). 즉 각각의 데이터 항목을 따로 기억하는 대신 데이터 항목간의 상호관계를 저장하는 것이다.

\[ M = [K(I_i, I_j)] = [K_y] = \begin{bmatrix} K_{11} & K_{12} & \cdots & K_{1N} \\ K_{21} & K_{22} & \cdots & K_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ K_{N1} & K_{N2} & \cdots & K_{NN} \end{bmatrix} \]

내적 커널을 사용할 경우 \( K_y = K(I_i, I_j) = <k(I_i), k(I_j)> = k(I_i) \cdot k(I_j) \)이다. 그러나 이 방법은 \( N \times N \) 크기의 매트릭스를 저장하여야 하며 \( N \)이 클 경우 \( (N > 5000) \) 매트릭스 계산이 실제적으로 불가능해 진다. 또한 이 방법은 학습 데이터가 늘거나 줄 때마다 커널 매트릭스를 전체적으로 다시 계산해야 한다. 네의 정보처리 능력이 이러한 커다란 매트릭스 연산을 정확하고 빠르게 수행할 수 있는지도 의문이다.

하이퍼망 기억구조에서는 위의 방법들과는 달리 \( n \)차원 이미지 전체를 모두 저장하지 않는다. 대신 많은 수의 저차원 이미지로 나뉘어 저장한다. 모든 가능한 저차원의 조각 이미지의 수는 \( n \)개의 변수들로 구성된 집합 \( X \)에 대한 멱집합의 크기 만큼 가능하다.

\[ Pw(X) = \{ E_i | E_i \subseteq X \} = 2^k \]

\[ |Pw(X)| = 2^k \] (1.16)

이 크기는 급격히 증가한다. 그러나 많은 실세계의 데이터가 규칙성을 가지고 있으며 이러한 데이터에 대해서는 \( 2^k \)보다 월등히 작은 수의 메모리 요소를 사용하고도 전체 데이터를 대별할 수 있는 모델이 존재한다. 확률이 높거나 정보량이 많은 작은 조각 이미지를 잘 결합한다면 데이터집합의 크기에 두관하게 전체 데이터의 정보를 잘 보존하되 compact한 code를 생성할 수 있다. 이는 커널 방법이 학습 예제집합의 크기에 비례하여 복잡도가 증가하는 것과 대조적이다. 실제로 2장에서 살펴본 바와 같이 인간의 기억현상에 대한 연구 결과들이 이러한 가설을 뒷받침하고 있다. 또한 숫자 인식 및 얼굴 인식 문제나 영화자막 데이터에 대한 실험 결과에서도 cardinality가 작은 하이퍼메시지를 사용하고도 좋은 메모리 성능을 낼 수 있음을 보여준다.

기억 저장에 관한 localists vs. globalists debate 관점에서 하이퍼망은 이 두 가지 측면을 모두 가지고 있다. 조각 조각의 하이퍼메시지는 단편적인 기억 단위를 표현하는 점에 있어서 localist적인 요소를 가지고 있다. 한편 하이퍼메시지는 따로 존재하고 있고 인접한 하이퍼메시지들과 링을 형성하면서 결합되어 있기 때문에 전체적으로는 globalist 적인 요소를 다분히 지니고 있다. 또한 하나의 하이퍼메시지는 다양한 cardinality를 가지는 하이퍼메시지들이 공존한다. Cardinality \( k \)가 작은 에제는 좀더 일반적인 규칙을 표현하고 \( k \)가 큰 에제는 예외적인 규
칙들은 표현한다. 전체 맏은 이들의 mixture이므로 이점에 있어서도 localist적인 요소와
globalist 적인 요소가 함께 하나의 모델에 포함되는 것으로 볼 수 있다. 하이퍼망은 localist-
globalist의 전체 스펙트럼 상에서 주어진 데이터에 대한 적합한 mixture를 찾는 것을 목적으로
한다.

6장에서는 하이퍼망을 통해서 compact code를 학습하는 알고리즘에 관하여 논의한다. 그에 앞서서 다음의 5장에서는 먼저 하이퍼망을 분자컴퓨팅으로 구현하기 위한 기술에 대해
서 논의한다.