Developing a Photo-enabled Carbon Alkane Nomenclature Software

H30312 Seungmin Lee
H30417 Jaewook Lee
H30416 Yujun Lee

Introduction(English)

Since the emergence of the novel coronavirus, mankind has faced a new type of crisis that has not existed before. Not only was the quarantine industry activated to prevent the spread of the virus, but the demand for vaccines and various medicines exploded. Among them, vaccines played a major role in reversing the game. There is something that has been of great help in the development of these vaccines. It is artificial intelligence; especially, "Alpha Fold 2," developed by Google DeepMind, predicts the protein structure of the novel coronavirus among countless protein structures and additionally allows the development of inhibitory substances and vaccines based on the results. In particular, it still seems to linger the awe that AlphaFold felt when it produced the highest performance and accuracy ever at the 2020 CASP, a protein structure prediction competition. In this aspect, predicting and confirming the protein structure of various substances and the form of the chemical structure were considered essential in various studies. Prior to the prediction, many researchers or related experts started this study since the use of alkane molecules would increase if they could be easily classified and named.

Introduction (연구 동기)

신종 코로나 바이러스의 출현 이후 인류는 이전에는 없던 새로운 양상의 위기를 맞았다. 바이러스의 확산 방지를 위한 방역 산업이 활성화된 것은 물론, 백신 및 각종 의약품의 수요가 폭발적으로 급증하였다. 그중에서도 판도를 뒤집는 데 큰 역할을 한 것은 바로 백신이다. 이런 백신의 개발 과정에서 큰 도움이 된 것이 있다. 바로 인공지능인데, 구글 딥마인드사가 개발한 '알파폴드2'는 무수한 단백질 구조 중에서 신종 코로나 바이러스의 단백질 구조를 예측하여 이를 기반으로 억제 물질과 백신을 개발할 수 있도록 하였다. 특히 단백질 구조 예측 대회인 2020년 CASP 대회에서 알파폴드가 사상 최대의 성적과 정확성을 내었을 때 느꼈던 경외심을 아직까지도 여운을 주는듯 하다. 이처럼 여러 물질의 단백질 구조, 그 기반의 화학적 구조의 형태를 예측하고 확인하는 것은 여러 연구에 있어서 필수적이라고 생각하였다. 이에 예측에 앞서 많은 연구자들, 또는 관련 전문가들이 알케인 분자들을 쉽게 분류하고 명명할 수 있게된다면 그 활용성이 높아질 것이라고 생각하여 본 연구에 착수하게 되었다.

Purpose (연구 목적)

본 연구의 목적은 알케인 구조를 가지고 있는 형태의 분자의 이름을 쉽게 확인할 수 있는 명명 소프트웨어를 개발하는 것이다. 알케인 분자들은 상대적으로 간단한 명명법을 가지고 있지만, 탄소의 최외각 전자가 4개인 것을 고려하면 단순히 명명하기 복잡한 안케인 분자들도 존재할 것이라고 예측할 수 있다. 그러므로, 복잡한 알케인 분자를 신속하게 명명하는 소프트웨어를 개발한다면 연구 환경이 더 효율적일 것이다. 분류를 위한 프로그램의 경우 기존의 알파폴드와 같이 무수한 가능성으로부터 인공신경망을 기반으로 추론해나가는 과정보다는 직접 통계 코드를 설정하여 IUPAC 명명법 (IUPAC system of nomenclature)에 기반하여 명명해나가는 과정이므로 더욱 간단한 과정을 통해 프로그래밍할 수 있을 것으로 기대된다.

Background Research (선행 연구/이론적 배경)

IUPAC 명명법(IUPAC system of nomenclature)을 토대로 소프트웨어 설계를 하였는데, McMurry의 "Organic Chemistry, 8th Edition"에서 기재되어 있는 알케인 명명법을 사용하였다.

Nomenclature of Alkane (Locant—Prefix—Parent—Suffix)

1. Finding the parent hydrocarbon
    a. Longest continuous chain of carbon atoms
    b. If equal length, choose one with larger number of branch points as the parent
2. Number the atoms in the longest chain
    a. Nearer the first branch point
    b. If same distance, begin number at the end nearer the second branch point
3. Identify and number the substituents
4. Write the name as a single word
5. Name a branched substituent as though it were itself a compound

명명법 5번 문항에서 Branched substituent를 따로 compound로 취급하라는 부분에서 IUPAC 명명법을 준수하기로 하였다. 이 뜻은 보통 과학계에서 흔히 "Isopropyl (*i*-Pr)"이라고 명명하는 알킬 그룹을 "1-Methylethyl)이라고 표기하겠다는 것이다.

Methods (연구 방법)

본 연구 모델 제작 환경은 Google Colaboratory이었으며, easyOCR과 Pytesseract 모듈을 사용하였다. 이미지 resize를 위해서는 opencv를 사용하였고, 기존 OCR은 화학 원소 사이의 선을 인식할 수 없었기에 텐서플로우에서 제공하는 오픈소스인 Keras를 이용하여 Text localization과 text identification을 위한 CNN 모델을 생성해 학습시켰다.

먼저 좌표가 흔들리게 된다면 이차원 배열로 옮길 때 문제가 발생할 수 있으므로 이미지 크기를 고정시켜 주기 위하여 opencv를 사용하여 크기를 고정해주었다. 아래 그림 1과 그림 2에서 보여지듯 각 원소를 (0, 0)을 기준으로 오른쪽 밑으로 내려가는 것을 알 수 있다. Pytesseract와 easyOCR을 사용하여 문자를 인식하면 그림 2과 같이 좌표가 tuple형식으로 입력되는데, 이때 원소 사이를 잇는 선을 착각하여 빨간 박스처럼 "-"를 인식할 수 있다. 이 때문에 영어와 숫자를 제외한 모든 공백 및 기호를 제거해줬다. 그 후 따로 제작한 CNN모델을 사용하여 원소 사이의 가로선과 세로선을 인식하여 그 좌표들을 이용해 2차원 배열에 집어넣었다.

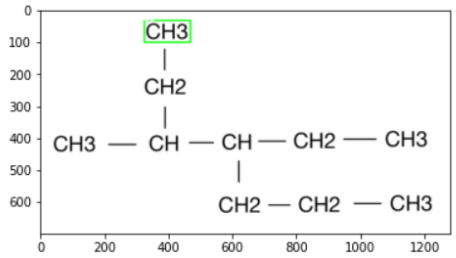


그림 1

```
[([[325, 31], [469, 31], [469, 99], [325, 99]], 'CH3', 0.8517652950481003),
 ([[321, 205], [463, 205], [463, 273], [321, 273]], 'CH2', 0.8846434770369346),
 ([[37, 383], [181, 383], [181, 451], [37, 451]], 'CH3', 0.8040857592683385),
 ([[337, 383], [439, 383], [439, 449], [337, 449]], 'CH', 0.9998014098258808),
 ([[565, 381], [721, 381], [721, 445], [565, 445]],
  'CH -',
  0.6815206829873797),
 ([[787, 375], [931, 375], [931, 443], [787, 443]], 'CH2', 0.7685347239499876),
 ([[1072, 368], [1216, 368], [1216, 436], [1072, 436]],
  'CH3',
  0.9117160439491272),
 ([[553, 575], [697, 575], [697, 643], [553, 643]], 'CH2', 0.8545180940812478),
 ([[801, 573], [945, 573], [945, 641], [801, 641]], 'CH2', 0.6702746977879699),
 ([[1087, 569], [1233, 569], [1233, 637], [1087, 637]],
  'CH3',
  0.9245463748310206)]
```

그림 2

30X30의 빈 배열을 생성한 후 가장 왼쪽에 있는 원소를 위 좌표를 이용하여 구한다. 그 후 (15, 0)에서부터 차례로 집어넣는다(좌표는 (열, 행)으로 표시하겠다). 각 원소를 이차원 배열에 삽입하였고, 편의를 위하여 원소를 연결하는 선은 숫자 1로 표시하였다.[그림 3]

```
12 :  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
13 :  [0, 0, 0, 0, 'CH3', 0, 0, 0, 'CH3', 0, 0, 0, 0, 0, 0,
14 :  [0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0
15 :  ['CH3', 1, 'CH2', 1, 'C', 1, 'CH2', 1, 'CH', 1, 'CH3'
16 :  [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
17 :  [0, 0, 0, 0, 'CH2', 1, 'CH3', 0, 0, 0, 0, 0, 0, 0, 0,
18 :  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

그림 3

위에서 언급한 nomenclature을 찾기 위한 가장 첫번째 방법은 parent hydrocarbon을 찾는 것인데, 가장 긴 chain을 찾기 위해서는 탐색 알고리즘 중 하나인 DFS를 사용하기로 하였다. DFS(깊이 우선 탐색법)에서는 깊은 부분을 먼저 탐색하기에 하나의 시작 노드를 기준으로 잡고 끝 노드에 도착할 때까지 계속 나아가다며 방문 처리를 하다 도착하면 최상단의 노드를 하나씩 꺼내며 돌아오는 탐색 방법이며, 이는 모든 경로를 한번씩 방문할 수 있기에 가장 긴 길이를 확인하기에 최적의 알고리즘이라 판단했다.

```
for i in range(len(vec)):
  x=vec[i][0]
  y=vec[i][1]
  if(a+x>=0 or a+y>=0):
    if(arr[a+x][b+y]==1 and arr[a][b]!=1 and visited[a+x][b+y]==0):
      visited[a+x][b+y]=1
      dfs(a+x, b+y, count+1)
      visited[a+x][b+y]=0
    elif(arr[a+x][b+y]!=0 and arr[a][b]==1 and arr[a+x][b+y]!=1 and visited[a+x][b+y]==0):
      visited[a+x][b+y]=1
      dfs(a+x, b+y, count)
      visited[a+x][b+y]=0
```

DFS를 하기 위해서는 먼저 시작 노드와 끝 노드가 필요한데, 이는 먼저 vec이라는 배열을 만들어준 후 상, 하, 좌, 우로 움직일 수 있도록 [[0, 1], [1, 0], [0, -1], [-1, 0]]
들을 삽입하였다. 그림 4와 같이 상, 하, 좌, 우에 0이 3개, 그리고 1이 하나라면 시작 노드 또는 끝 노드이다.
DFS를 실행한 후 가장 긴 chain의 개수를 출력하면 그림 5와 같고, 이차원 배열로 구현하였다[그림 6]. 그리고 그림 7처럼 Branch들을 표현한 후 Branch의 위치에 2를 표현하였고, 최종적으로 nomenclature 을 위해 번호를 매기고 Branch의 시작점을 'B'로, 연결선을 '-' 표현하면 그림 8과 같다.

```
12 :  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
13 :  [0, 0, 0, 0, 'CH3', 0, 0, 0, 'CH3', 0, 0, 0, 0, 0, 0,
14 :  [0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0
15 :  ['CH3', 1, 'CH2', 1, 'C', 1, 'CH2', 1, 'CH', 1, 'CH3'
16 :  [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
17 :  [0, 0, 0, 0, 'CH2', 1, 'CH3', 0, 0, 0, 0, 0, 0, 0, 0,
18 :  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

그림 4

```
visited from [11, 2]
6
7
visited from [15, 0]
visited from [15, 8]
visited from [17, 8]
```

그림 5

```
11 :  [0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
12 :  [0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
13 :  [0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
14 :  [0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
15 :  [0, 0, 1, 1, 1, 0, 0, 0, 0, 0,
16 :  [0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
17 :  [0, 0, 0, 0, 1, 1, 1, 1, 1, 0,
```

그림 6

```
0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0,
2, 1, 1, 1, 2, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 1, 1, 1, 1, 1, 0,
```

그림 7

```
0, 0, 0, 0, 0, 0, 0, 0, 0, 0
0, 0, 0, 0, 0, 1, 0, 0, 0, 0
0, 'B', 0, 0, 0, '-', 0, 0,
'B', 4, '-', 3, '-', 2, 'B',
0, '-', 0, 0, 0, 0, 0, 0, 0,
0, 5, '-', 6, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0
0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

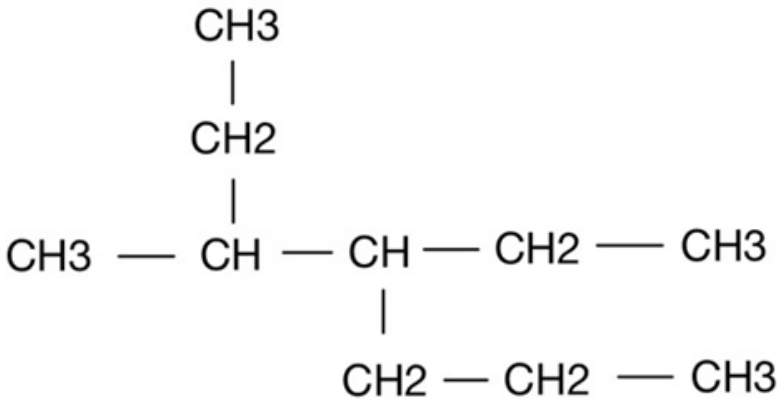
그림 8

위에서 구했던 길이를 suffix라는 변수에 담고 시작점부터 한칸씩 움직이며 Branch를 한번씩 방문하며 C의 개수, H의 개수를 세주며 배열에 담는다. 그 후 정렬해준 후 양식에 맞춰 출력하면 그림 9와 같은 결과가 나온다. 만약 branched substituent이 있다면 위의 DFS를 한번 더 돌려줘 나온 값을 바탕으로 nomenclature을 완성하면 된다.

```
7.0
Heptane
Methyl
Ethyl
[['Methyl', 3], ['Ethyl', 4]]
[['Ethyl', 4], ['Methyl', 3]]
4-Ethyl 3-MethylHeptane
```

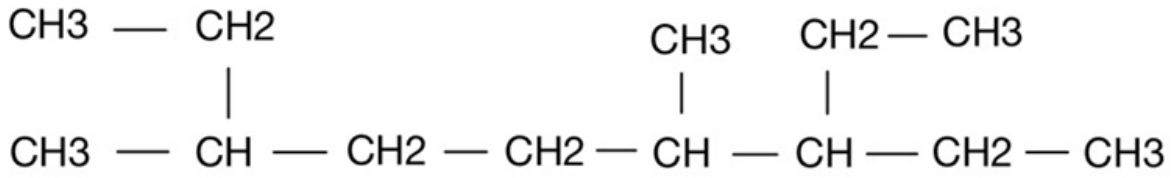
그림 9

Results (연구 결과)




```
[0, 0, 'CH3', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 'CH2', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
['CH3', 1, 'CH', 1, 'CH', 1, 'CH2', 1, 'CH3', 0,
[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 0, 0, 'CH2', 1, 'CH2', 1, 'CH3', 0, 0, 0,
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, '-', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, '-', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 'B', 3, '-', 4, 'B', 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 0, 0, '-', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 0, 0, 5, '-', 6, '-', 7, 0, 0, 0, 0, 0, 0,
```

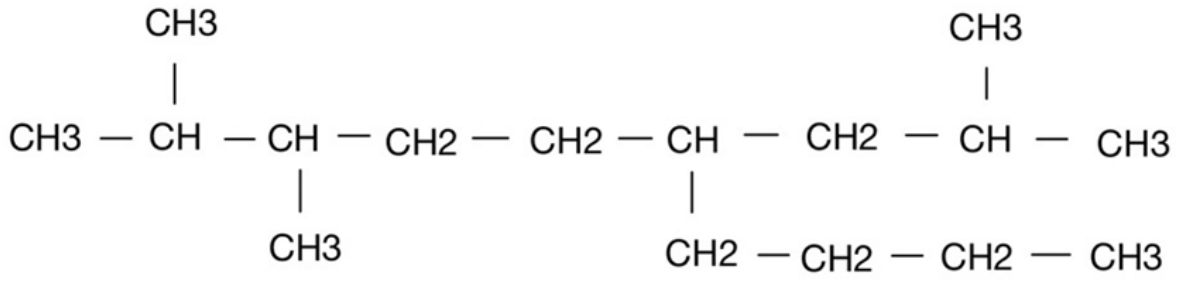
4-Ethyl 3-MethylHeptane



```
['CH3', 1, 'CH2', 0, 0, 0, 0, 0, 'CH3', 0, 'CH2', 1, 'CH3', 0, 0, 0, 0, 0,
[0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
['CH3', 1, 'CH', 1, 'CH2', 1, 'CH2', 1, 'CH', 1, 'CH', 1, 'CH2', 1, 'CH3',
```

```
[9, '-', 8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, (
[0, 0, '-', 0, 0, 0, 0, 0, 'B', 0, 'B', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 'B', 7, '-', 6, '-', 5, '-', 4, '-', 3, '-', 2, '-', 1, 0, 0, (
```

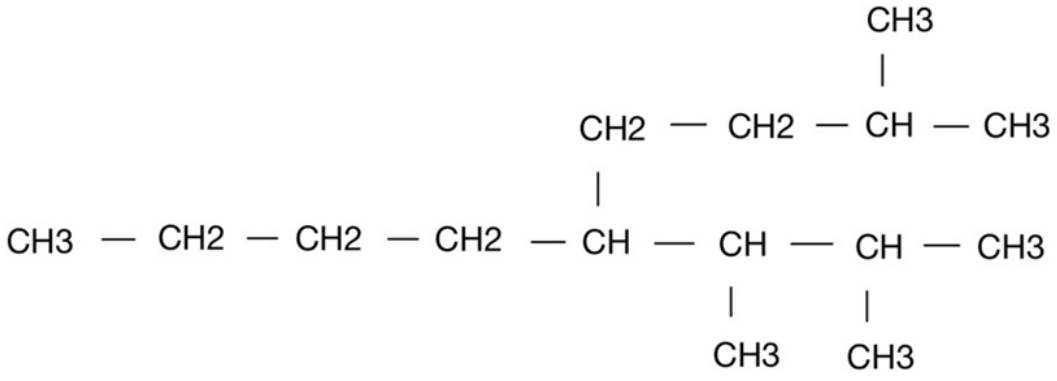
3-Ethyl 4,7-diMethylNonane



```
[0, 0, 'CH3', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 'CH3', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
['CH3', 1, 'CH', 1, 'CH', 1, 'CH2', 1, 'CH2', 1, 'CH', 1, 'CH2', 1, 'CH3', 0, 0, 0,
[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 0, 0, 'CH3', 0, 0, 0, 0, 0, 'CH2', 1, 'CH2', 1, 'CH2', 1, 'CH3', 0, 0, 0, 0,
```

```
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, '-', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 'B', 2, '-', 3, '-', 4, '-', 5, '-', 6, 'B', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 0, 0, 'B', 0, 0, 0, 0, 0, '-', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 7, '-', 8, '-', 9, '-', 10, 0, 0, 0, 0, 0, 0, 0, 0
```

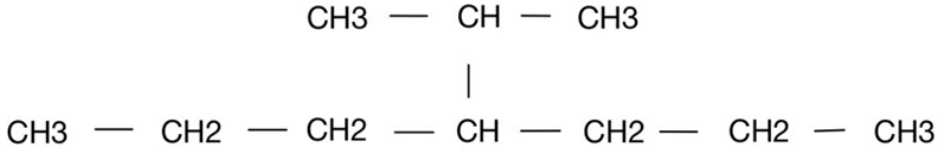
2,3-diMethyl 6-(2-MethylPropyl) Decane




```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 'CH3', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, (
[0, 0, 0, 0, 0, 0, 0, 0, 'CH2', 1, 'CH2', 1, 'CH', 1, 'CH3', 0, 0, 0, 0, (
[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, (
['CH3', 1, 'CH2', 1, 'CH2', 1, 'CH2', 1, 'CH', 1, 'CH', 1, 'CH', 1, 'CH3',
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, (
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 'CH3', 0, 'CH3', 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, (
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, (
[0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 'B',
[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, (
[1, 1, 1, 1, 1, 1, 1, 1, 1, 'B', 0, 0, 0, 0,
```

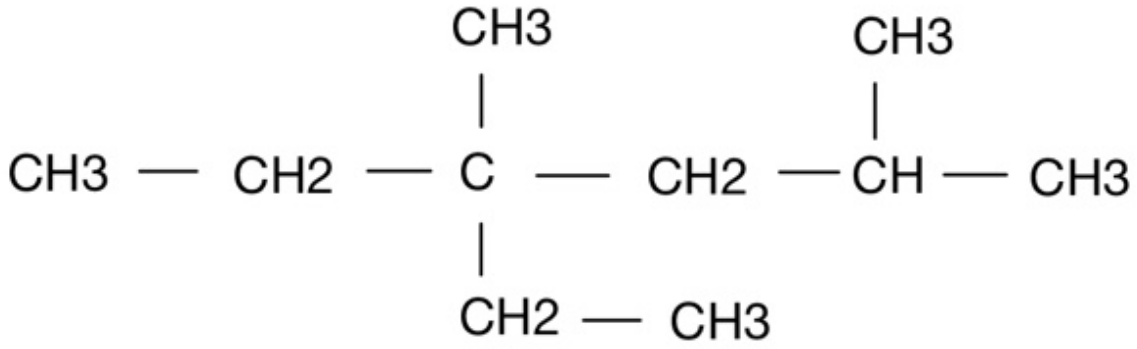
5-(1,2-DimethylPropyl) 2-MethylNonane



```
[0, 0, 0, 0, 'CH3', 1, 'CH', 1, 'CH3', 0, 0, 0, 0, 0, 0, 0, 0, 0
[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
['CH3', 1, 'CH2', 1, 'CH2', 1, 'CH', 1, 'CH2', 1, 'CH2', 1, 'CH3',
```

```
[0, 0, 0, 0, 0, 0, 'B', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[1, '-', 2, '-', 3, '-', 4, '-', 5, '-', 6, '-', 7, 0,
```

4-(1-MethylEthyl) Heptane




```
[0, 0, 0, 0, 'CH3', 0, 0, 0, 'CH3', 0, 0, 0, 0, 0, 0,
[0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
['CH3', 1, 'CH2', 1, 'C', 1, 'CH2', 1, 'CH', 1, 'CH3',
[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 0, 0, 'CH2', 1, 'CH3', 0, 0, 0, 0, 0, 0, 0, 0,
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
[0, 0, 0, 0, 'B', 0, 0, 0, '-', 0, 0, 0, C
[0, 0, 0, 'B', 4, '-', 3, '-', 2, 'B', 0,
[0, 0, 0, 0, '-', 0, 0, 0, 0, 0, 0, 0, 0,
[0, 0, 0, 0, 5, '-', 6, 0, 0, 0, 0, 0, 0,
```

4-Ethyl 2,4-diMethylHexane

Conclusion & Discussion (결론 및 토의)

결과적으로 자동화 과정을 도입하여 여러 알케인 분자를 더 빠르게 명명할 수 있게된다는 점은 화학 분자를 활용하는 연구에 있어 높은 효율성을 가져다줄 것으로 예상된다. 현재는 분류에 초점을 둔 프로그램을 개발하였다면, 향후에는 분류를 넘어 과학계에서 흔히 사용되는 이름도 함께 출력하고, 사람의 글씨체를 읽을 수 있게하는 등 효율성을 더욱 높이는 방안도 가능할 것이라고 생각된다.

그럼에도 사용자의 활용성 및 분류 기준 자체에 더욱 초점을 두는 것은 도구적 수단으로 사용하기 위해 개발하는 것이기 때문이다. 만일 알파폴드를 넘어 대부분의 연구자들의 연구 절차가 대체되어버린다면, 그때는 연구의 본질과 더불어 우리의 역할에 관해 다시 한 번 생각해보게될 수도 있으므로 인간의 창의성을 기반으로 과정을 좀 더 효율적으로 시행할 수 있는 도구적 수단 정도에 머무르는 정도로 개발되어야할 것 같다. 물론 알파폴드와 같은 기술은 높은 활용성을 지녔고 스스로 판단도 가능하나, 역시 많은 가능성 중 일부를 산출하면 그때부터는 인간이 백신 개발 등 연구를 시작해야한다는 점은 여전히 도구적 수단으로서 남아있다는 것을 뜻하기도한다.

Reference

1. Mcmurry J. Organic chemistry. 9th ed. Boston: Cengage Learning; 2016.

2. Janet,M. T. *et al.* (2021) Alpha Fold heralds a data-driven revolution in biology and medicine

3. Senior,A.W. *et al.* (2019) Protein structure prediction using multiple deep neural networks in the 13th Critical Assessment of Protein Structure Prediction (CASP13)

4. Smith, Ray. "An overview of the Tesseract OCR engine." Ninth international conference on document analysis and recognition (ICDAR 2007). Vol. 2. IEEE, 2007.

5. Tarjan, Robert. "Depth-first search and linear graph algorithms." SIAM journal on computing 1.2 (1972): 146-160.