단백질 서열과 텍스트 정보 기반 오토마타 종 분류기

박준형[O], 이현정[1], 양지훈[1], 김선호[1]
다이퀘스트 연구소[O], 서강대학교 컴퓨터학과[1]
pustar@gmail.com, luckyhj777@naver.com, yangjh@sogang.ac.kr, shkim@lex.yonsei.ac.kr

Automata Species Classifier based on Protein Sequences and Text Information

요약

단백질 분류는 현대 생물학의 큰 도전과제이다. 현재 여러 단체에 의해 잘 관리되는 상세한 주석이 달린 많은 양의 단백질 정보들이 존재한다. 이러한 데이터베이스의 덕분으로 다양한 물리 화학적 특성과 주석들에 기반하고 있는 분류 기법들이 연구되고 있다. 특히, 아미노산들로 이루어진 단백질 서열이 해당 단백질의 분류에 중요한 역할을 하는 진화적 기록들의 단서가 되기 때문에 단백질 서열들에 대한 연구가 활성화되고 있다. 비록 단백질 서열이 단백질 분류 문제의 중요한 특징이 된다고 해도 단순한 단백질 서열만으론 해당 단백질에 대한 충분한 정보를 얻을 수 없으며, 타 종 간에도 기능상 유사성 때문에 서로 비슷하게 판별될 수 있다. 이러한 문제점에 착안해서 우리는 오토마타 종 분류기라고 부르는 새로운 시스템적인 종 분류 접근 방법을 제안한다. 이 시스템의 클러스터링과 종 분류 판별 성능에 대한 평가 실험을 수행해본 결과, 상대적으로 좋은 성능을 얻을 수 있었다.

1. 서론

생명과학의 급격한 발전으로 생명과학 분야에 관련된 수많은 데이터가 쏟아져 나오고 있다. 이러한 상황에서, 특히 단백질 서열데이터는 양이 특히 방대하고 직관적으로 의미를 알 수 없기 때문에 컴퓨터학의 알고리즘을 이용한 연구가 활발하게 진행되어왔다. 단백질 서열 데이터를 처리할 때 가장 중요시 되는 것은 각 단백질 서열간의 유사도이다. 계통발생학에 따르면, 단백질 서열은 진화에 따라 특정 부분의 아미노산이나 염기서열이 변화하게 된다. 두 단백질이 매우 작은 부분만을 제외하고 대부분 일치한다면 이 단백질들은 서로 매우 유사하며, 반대의 경우라면 이 단백질들은 서로 다른 진화과정을 거쳤다고 간주할 수 있다. 따라서 단백질 서열을 비교해보면 해당 단백질들이 어떤 관계를 가지는가를 추측해 볼 수 있다. 본 논문에서는 텍스트 마이닝 기법을 이용해서 한 종의 단백질 서열들을 클러스터링 해보고 성능을 평가하였다. 그리고 클러스터링 된 단백질 서열들에 대해서 각각 MSA기법을 이용해서 일치하는 부분들을 찾은 후, 일반화시켜 오토마타를 사용한 모델을 제작, 종 단위의 분류를 시도하였다. 그리고 이 모델을 이용, 병합해서 한 계층 더 위의 종을 분류하는 방법을 제안한다.

2. 오토마타 종 분류기의 제작

2.1 단백질 서열들의 기능 단위 클러스터링

단백질 데이터는 해당 단백질이 존재하는 기질이나 생화학적 기능 등에 따라서 구별된다. 다른 종들 간의 단백질이라고 해도 그 위치나 기능이 유사하면 매우 비슷한 형태를 가진다. 따라서 어떤 단백질 서열이 주어졌을 때 이 단백질이 어떤 종에 속하는가를 결정하려면 그 단백질을 단순히 다른 종의 단백질 서열들과 비교하는 것만으로는 충분하지 않다. 따라서 종 분류기를 제작하려면 우선, 그 종에 해당하는 단백질 서열들을 클러스터링해서 클러스터 별로 특성을 추출하는 작업이 필요하다. 단백질 서열의 클러스터링의 기준으로 텍스트와 서열 정보를 이용하는 두 가지 방법을 생각해볼 수 있다. 텍스트 기반 클러스터링은 일반적인 텍스트마이닝 기법을 통해 이루어진다. 즉, 워드 벡터를 만들어서 그것을 기반으로 각 단백질들의 유사도를 측정하는 것이다. 이를 위해서는 각 단백질을 대표하는 단어들이 필요하다. 많은 단백질 데이터들이 이런 데이터를 따로 필드를 두어 제공하고 있다. SwissProt[1] 단백질 데이터에는 DE 필드가 존재한다. 이 필드는 해당 단백질을 분류하는데 어느 정도 충분한 서술 정보를 포함한다. 그 밖에도 단백질을 분류하는데 유용하게 이용될 수 있는 텍스트 정보를 포함한 필드로는 CC 필드와 KW 필드가 있다. CC 필드는 해당 단백질에 대한 몇 가지 주제에 대한 정보를 담을 수 있다. KW 필드는 기능, 구

조 및 다른 속성에 따라 구별할 수 있는 정보를 제공하는 필드이며 선택적으로 존재한다. 이러한 주어진 단어들을 이용, 워드 벡터 행렬을 만들어 클러스터링을 수행할 수 있다. 대부분의 MSA기법은 여러 단백질들 간의 유사도를 서로 비교하고 비슷한 노드들을 인접 이웃들끼리 병합해서 계층발생학 트리를 만든다. 이 트리를 이용해서 MSA 알고리즘이 단백질을 어떻게 클러스터링 했는지를 알 수 있다. 즉, 하위 노드들이 어떻게 묶여서 전체를 표현하는 계층적 구조를 이루고 있는지를 알 수 있다. MSA의 결과 중에서 단순히 클러스터링 성능만이 텍스트 기반 클러스터링의 성능과 비교된다.

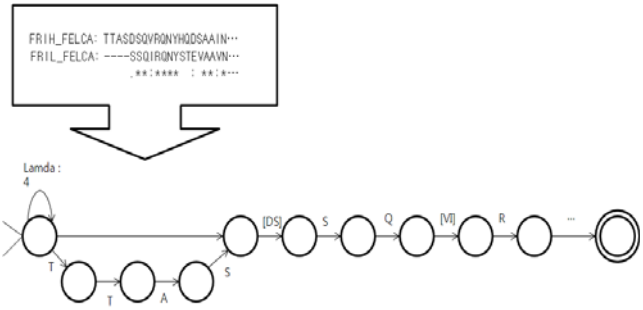
2.2 MSA 일반화 오토마타 제작

이전 장에서 설명한 클러스터링 방식으로 특정 종에 대한 단백질들을 기능별로 클러스터링한다. 그리고 이렇게 얻어진 클러스터들에 대해서 MSA를 수행한다. MSA를 해주는 도구로는 여러 가지가 있지만 본 논문에서는 그중 널리 쓰이는 CLUSTALW를 사용한다. CLUSTALW를 통해 얻을 수 있는 결과 중에 본 논문에서 사용되는 것은 단백질 서열에서 완벽한 일치가 일어나는 부분과 끊김의 위치 정보이다. MSA 결과 예제는 (그림 1)과 같다.

```
COX1_FELCA      ...WLVPLMIGAPD...
COX2_FELCA      ...MAYPFQLGFQD...
COX3_FELCA      ...-PSPWPLTGAL...
                        *  :
```

(그림 1) MSA 결과 예제

(그림 1)은 COX1, COX2 그리고 COX3 단백질의 클러스터에 대해 MSA를 수행한 예제이다. '*' 로 표기된 부분이 완벽하게 일치하는 부분이고 '-' 로 표시된 부분이 끊김인 부분이다. ':' 이나 '.' 으로 표시된 부분은 높은 확률로 서로 치환 가능한 아미노산들이다. 위의 결과 파일들을 토대로 우선, 완전히 일치하는 부분을 기반으로 기본적인 오토마타를 제작한다. 완전히 일치하는 부분이 하나의 전이가 되며 끊김은 람다 전이로 정의된다. (그림 2)은 이에 대한 예제이다.



(그림 2) MSA 결과 예제

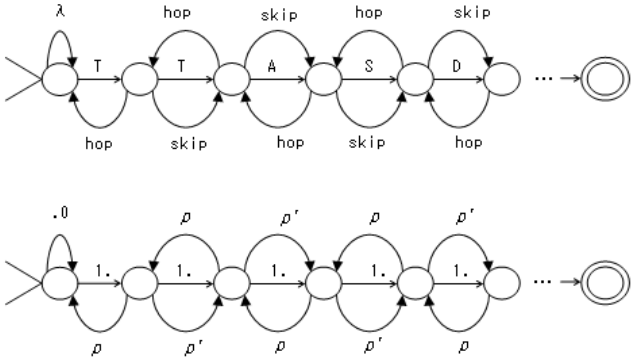
(그림 2)에서 [DS], [VI]처럼 대괄호로 표현된 노드들은 두 노드가 치환 가능한 노드일 경우 합병되어 생긴 노드들이다. 즉, 아미노산 D와 S가 서로 치환될 확률이 크기 때문에 유사한 노드로 판정, 합병하는 것이다. 유사한 노드로 판정하는 근거는 여러 가지가 있다. 위 예제에서 사용된 근거는 Taylor에 의해 제안된 Venn diagram[2]에 기초한다. 본 논문에서는 Kristine Yu가 제안한 방법[3]에 착안하여 치환 가능한 아미노산의 그룹을 좀 더 확장해 보기로 하였다. 이 방법은 가령 아미노산 F와 W가 단백질 서열 정렬시 같은 위치에 놓이게 되고 치환 가능한 그룹에 F, W, Y가 존재한다면, 그 위치에서의 오토마타 전이는 [FWY]가 되어 그 아미노산이 Y일 때도 F나 W일 때와 마찬가지로 동등한 전이를 허용하는 방식이다. 그러나 이렇게만 제작한 오토마타에 단백질 서열을 테스트하면 수많은 불일치들이 일어나게 된다. 그것은 단백질 서열을 비교할 때 아미노산의 삽입, 삭제 등의 여러 이유로 인해 끊김이 존재하기 때문이다. 본 논문에서는 간단한 확률 모델을 이용해서 오토마타 테스트를 반복해주는 방식을 사용해서 끊김을 처리한다.

일단, 불일치가 발생할 경우, 현재 테스트 중인 서열의 단백질을 무시하고 오토마타에 다음 단백질 서열을 계속 테스트하는 방법을 생각해 볼 수 있다. 이러한 방법을 hop이라고 정의한다. 그리고 다른 경우로서 현재 테스트중인 서열의 단백질을 다음 단계의 오토마타서부터 다시 테스트하는 경우를 생각할 수 있다. 이와 같은 경우를 skip이라고 정의한다. hop은 진화 단계에서 어느 한 아미노산이 삭제되었을 경우를 염두에 둔 정의이고 skip은 진화 단계에서 어느 한 아미노산이 추가된

경우를 염두에 둔 정의이다. 이와 같이 진화과정에서 자주 나타나는 삽입, 삭제 그리고 치환 현상을 각각 hop, skip 전이 그리고 위에 언급했던 오토마타 일반화를 통해 고려해준다. 매번 불일치를 만나게 될 때마다 그 위치로부터 다시 테스트를 수행하는 프로세스를 생성하고, 테스트를 수행하는 프로세스는 확률에 따라 hop을 할건지 skip을 할 것인지를 결정한다. 오토마타 $S_0S_1S_2...S_k...S_{n-1}S_n$와 $P_0P_1P_2...P_l...P_{m-1}P_m$라는 단백질 서열이 존재한다고 가정하자. S_k 에서 P_l 을 테스트하고 있을 때 hop, skip이 일어날 각각의 확률은 각각 다음 수식들과 같다.

$$P(hop|miss)=\frac{(n-k)}{(n-k)+(m-l)}$$
$$P(skip|miss)=\frac{(m-l)}{(n-k)+(m-l)} \quad (1)$$

이 식은 오토마타 테스트 과정이 global match 지향적인 작동을 하도록 해준다. 즉, 현재 테스트해야 할 단백질의 서열 길이가 남아있는 오토마타의 상태 수보다 길면 hop을 할 확률이 커진다. 반대의 경우에는 skip을 할 확률이 커지게 된다. (그림3)은 이렇게 구해진 확률값과 완전한 일치할 때 이루어지는 전이의 가중치 값을 표현한 오토마타 테스트를 보여주는 그림이다.



(그림 3) hop, skip이 적용된 오토마타 테스트 모델

현재 상태에 정의된 전이가 테스트중인 서열의 다음 아미노산인 경우, 1의 가중치를 전체 점수에 더하고 다음 상태로 전이한다. 람다 전이나 skip, hop에 의한 전이는 가중치 0을 전체 점수에 더하고 전이한다. 이런 식으로 테스트를 수행할 때 문제점은 이 방식이 전역 최대점에 도달하지 못할 수 있다는 점이다. 이러한 문제점을 해소하기 위한 노력으로서 수식 (1)에 상수 alpha 값을 더해줄 수 있다. 여기서 alpha 값은 매 시행마다 조금씩 감소하게 설정해주는 값으로서, 오토마타의 어느 한 상태에서 다음 전이로 hop이나 skip이 확정되어져 버리는 것을 막기 위한 상수이다. 이 값이 0.5이면 한 상태에서의 hop 혹은 skip을 할 확률은 정반대가 된다. 이 값은 0.5보다 작은 상수를 매 오토마타 테스트 시행마다 실험적으로 반복할 횟수로 나눠준 값으로 감소시켜주며 0일 때 실행 반복을 멈춘다. 이러한 방식의 실험 설계는 simulated annealing기법[4]에서 주로 사용되는 방법이며 휴리스틱이지만 효과적으로 전역 최대점을 찾는 것으로 알려져 있다.

아무런 제약 조건 없이 지금까지 제안한 방식으로 오토마타 테스트를 진행하다보면 연산의 부담이 너무 커져서 계산에 굉장히 많은 시간이 필요하다. 따라서 몇 가지 제약을 두어서 연속된 에러 발생으로 인한 계산 부담을 줄여야한다. 단, 이러한 방식을 사용하면 오토마타 테스트 과정에서 앞부분에 많은 에러를 가지고 있는 단백질 서열이 고르게 에러가 분포한 서열과 같은 에러 수를 가지고 있다고 해도 테스트에서 배제되어버리는 단점이 있다. 본 논문에서는 연속된 일치가 발견될 경우 연속된 불일치 발견 카운터 숫자를 줄여주는 보상을 통해 이러한 문제를 줄이고 있다.

3. 실험 및 결과

3.1 실험 방법

본 논문에서는 SwissProt의 단백질 데이터 중 felis, silurarna, equus, gorilla의 4개 종에 대한 단백질 서열 데이터에 대해 실험하였다. SwissProt의 단백질 데이터는 태그를 가진 필드들로 구성되어 있으며 그 중 실험에 쓰인 필드는 ID, SQ, DE, CC, KW이다.

텍스트 기준의 클러스터링에서는 단백질 데이터에서 각 단백질마다 가지고 있는 DE, CC, KW 필드의 단어들을 이용해서 클러스터링한다. KW 필드는 단백질에 따라 없는 경우도 있을 수 있다. 따라서 DE 필드만을 이용한 경우와 DE, CC 필드를 이용한 경우, 그리고 KW 필드를 포함한 세 필드를 모두 이용한 경우로 나

누어 실험해 보았다. 성능 평가의 지표로 삼은 척도는 널리 이용되는 클러스터링의 성능 평가 지표인 purity와 entropy값들을 이용하였다. 일반적으로 잘 클러스터링 된 클러스터일수록 purity의 값이 크게 나타나고 entropy의 값이 작게 나타난다.

텍스트 기반 클러스터링을 위해 문서 데이터를 워드 벡터화 하는 작업이 필요하다. 이 작업은 doc2mat[1]을 이용해서 수행하였다. 클러스터링 작업은 이렇게 만들어진 워드 벡터를 이용, CLUTO[2]로 수행하였으며 유사도 측정을 위한 함수로는 코사인 유사도(cosine similarity)를 이용해서 측정하였다. 클러스터링시 사용한 알고리즘으로는 agglomerative 방식을 이용하였고 최적화 함수로는 *I2* 함수를 사용하였다. *I2* 최적화 함수의 식은 다음과 같다.

$$maximize \sum_{i=1}^{k} \frac{1}{n_i} \left(\sum_{v,u \in S_i} sim(v,u) \right) \quad (5)$$

n은 전체 단어의 개수이며 v와 u는 각각의 단어를 말한다. 이 식은 $O(n^3)$ 의 복잡도를 가지며 *H3* 최적화 함수와 더불어 일반적인 경우에 좋은 성능을 보이는 걸로 알려져 있다.

서열을 이용한 클러스터링을 수행하기 위해서 한 종에 대한 모든 단백질을 ClustaW을 이용해서 MSA를 수행한다. 수행시 사용한 옵션은 <표 1>과 같다.

<표 1> CLUSTALW 실행 설정값들

Options	Values
Protein Gap Open Penalty	10.0
Protein Gap Extension Penalty	0.2
Protein Matrix	GONNET
Protein/DNA ENDGAP	-1
Protein/DNA GAPDIST	4

오토마타 성능을 평가하기 위해 비교 실험으로 MUSCLE[3][5]을 사용한다. MUSCLE은 MSA를 통해 분류한 단백질들을 phenogram으로 표현할 수 있다. 이것을 이용해서 MUSCLE이 각 단백질을 어떻게 분류하는지를 볼 수 있다. 오토마타 성능 평가를 위해, 주어진 트레이닝 세트의 일부를 테스트 데이터로 이용하는 cross-validation 방법을 이용하였다. 즉, 주어진 silurana, gorilla, felis 그리고 equus의 단백질들을 각각 5분할로 나누어서 각각의 분할을 모은 5개의 집합을 만든다. 이 집합들을 MUSCLE이 서열 기반으로 어느 정도로 분류를 할 수 있는지 알아본다.

오토마타 성능 평가로는 한 종에 대해서 위에서 분할한 5개의 데이터 집합 중 하나를 학습 데이터로 사용하고 나머지 4개를 테스트 데이터로 사용한다. 4개의 분류기에 대해서 위의 과정을 반복하여 실험을 수행하고, 그 때 얻어진 점수를 통해 재현율, 정확률 그리고 f-measure값을 계산하여 성능을 측정한다.

3.2 실험 결과

텍스트를 이용한 클러스터링 실험에서는 4개의 종들을 분류하되, 사용하는 단어 벡터의 종류를 다르게 해서 실험하였다. 이것은 단백질에 대해서 서술하는 텍스트 데이터가 많아질 때 클러스터링의 성능이 어떻게 영향을 받는지에 대해 알아보기 위한 실험이다. 실험 결과는 다음과 같다.

<표 2> 각 텍스트 필드를 조합한 단백질 클러스터링 결과

	Dataset	#	*k*	Entropy	Purity
Silurana (frog)	DE	231	139	**0.12**	**0.85**
	DE+CC		150	0.14	0.79
	DE+CC+KW		150	0.14	0.78
Gorilla (gorilla)	DE	174	53	**0.12**	**0.83**
	DE+CC		53	0.14	0.78
	DE+CC+KW		53	0.14	0.76
Felis (cat)	DE	199	109	**0.07**	**0.91**
	DE+CC		119	0.09	0.83
	DE+CC+KW		119	0.10	0.81
Equus (horse)	DE	243	146	**0.05**	**0.95**
	DE+CC		146	0.08	0.88
	DE+CC+KW		146	0.08	0.86

단백질 서열만으로 클러스터링 한 결과에서는 말단 계층, 즉 k값이 클 경우에 성능이 좋게 나타났고 상위

1)http://glaros.dtc.umn.edu/gkhome/files/fs/sw/cluto/doc2mat.html
2) http://www.cs.umn.edu/~karypis/cluto
3) http://www.ebi.ac.uk/muscle/

계층으로 올라갈수록 좋지 않은 성능을 나타내었다. 텍스트 정보만을 이용해서 클러스터링 했을 때는 DE필드만을 이용해서 얻은 클러스터링 결과가 가장 좋고 CC, KW를 더할수록 성능이 떨어짐을 확인할 수 있었다. 이것은 CC와 KW필드가 단백질에 대한 풍부한 정보를 제공해 주지만 그로 인해서 해당 단백질에 대해 서술하는 단어의 양이 많아져 오히려 특성이 일반화 되어버리는 현상이 일어나기 때문이다. 실험 결과는 <표3>와 같다.

종 분류 실험에서는 주어진 단백질 데이터를 이용해서 최대한 신뢰도 있는 성능 평가를 하기 위해서 5-cross-validation 방법을 사용하였다. 즉, 5개의 종들에게 같은 비율만큼의 데이터를 취해서 합한 데이터 집합을 5개 만든 다음 돌아가면서 이 데이터 집합 중 하나는 테스트에 사용하고 나머지는 학습에 사용하는 방식이다. MUSCLE의 성능 평가를 하기 위해서 위에서 언급한 데이터 집합을 이용하였다. MUSCLE을 이용한 단백질 서열 분류 실험 결과는 <표 4>와 같다.

<표 3> 단백질 서열을 이용한 클러스터링 결과

	Depth	#	*k*	Entropy	Purity
Silurana (frog)	1	231	**136**	**0.21**	**0.75**
	2		88	0.25	0.61
	3		63	0.28	0.58
Gorilla (gorilla)	1	174	**106**	**0.15**	**0.77**
	2		71	0.20	0.65
	3		52	0.23	0.62
Felis (cat)	1	199	**121**	**0.26**	**0.87**
	2		82	0.32	0.54
	3		45	0.34	0.52
Equus (horse)	1	243	**147**	**0.24**	**0.89**
	2		93	0.33	0.78
	3		53	0.35	0.72

<표 4> MUSCLE의 종 분류 결과 (purity, entropy순)

	Cluster1		Cluster2		Cluster3		Cluster4		Cluster5		Cluster6		Result	
0	.32	.48	.9	.49	.34	.48	-	-	-	-	-	-	.32	.33
1	.30	.49	.1	.49	.37	.48	-	-	-	-	-	-	.33	.49
2	.35	.48	.47	.37	.42	.45	.32	.48	.43	.47	-	-	.39	.45
3	.29	.50	.33	.49	.32	.49	-	-	-	-	-	-	.31	.49
4	.36	.49	.44	.44	.38	.48	.47	.45	.47	.45	.35	.48	.39	.46

y축은 4개의 종들의 단백질을 각각 5등분해서 모은 테스트 집합이다. MUSCLE을 이용해서 phenogram을 만들어서 가장 최적인 경우, 즉 phenogram에서 4개의 종들을 정확하게 맞춘 경우에 생기게 되는 클러스터 개수인 4에 가장 가까운 클러스터 수를 정해 각각의 클러스터의 purity와 entropy를 따져보면 MUSCLE의 분류 성능을 확인해 볼 수 있다. x축은 이렇게 정해진 4에 가장 가까웠던 클러스터의 개수이다. 클러스터 수 4개일 때 아무런 기준 없이 임의로 분류한 경우의 각각의 평균 purity, entropy 값은 0.25, 0.5에 가까워지게 된다. 따라서 위 실험 결과는 임의로 구분한 결과에 비해 그다지 크게 성능이 나아지지 않은 값들을 보여주고 있어서 서열 기반의 단순한 분류만으론 주어진 실험 조건에서 종 분류를 효과적으로 해낼 수 없음을 알 수 있다.

오토마타 종 분류기의 성능 테스트를 위해 각 종들에 대해서 분류기를 만들어서 분류 성능을 확인해 보았다. 분류 기준은 오토마타 종 분류기를 통과해 얻은 score가 0.5이상을 넘으면, 즉 반 이상 일치하면 해당 종으로 간주하는 문턱값이다. 아래 표는 위에 말했던 데이터 집합을 각 분류기에 넣어 평균 점수를 구한 표이다. y축은 각 분류기, x축은 각 데이터 집합을 나타낸다.

<표 5> 각 종 분류기의 각 종 단백질 서열에 대한 평균 점수

	Silurana	Gorilla	Felis	Equus
Silurana	**0.59**	0.39	0.37	0.38
Gorilla	0.35	**0.69**	0.45	0.43
Felis	0.36	0.43	**0.75**	0.51
Equus	0.36	0.44	0.48	**0.77**

이렇게 구한 점수들을 이용해서 널리 쓰이는 성능 평가의 척도인 재현율, 정확률 그리고 f-measure으로 평가한 오토마타 종 분류기의 성능은 다음 <표 6>과 같다.

<표 6> 오토마타 종 분류기의 성능평가

	Silurana	Gorilla	Felis	Equus
Recall	**1.0**	0.9	0.85	0.85
Precision	**0.95**	1.0	1.0	1.0
F-measure	**0.97**	0.95	0.92	0.92

위에서 확인해 볼 수 있듯이, 오토마타 종 분류기들이 위의 평가방법을 사용했을 때 높은 성능을 확인할 수 있었다. 큰 차이는 아니지만, Silurana의 성능이 다른 세 개의 종의 성능보다 좋은 이유는 Silurana가 다른 세 개의 종들과의 생물 분류상의 거리가 더 멀기 때문이다.

4. 결론 및 향후과제

본 논문에서 제시한 오토마타 종 분류기는 실험 결과 효과적으로 종을 분류할 수 있으며 기타 기존의 방식들에 비해 뒤쳐지지 않음을 확인하였다. 하지만 본 논문에서 제시한 오토마타의 설계 및 테스트 과정엔 많은 제약 조건이나 설정해줘야 하는 값 등이 있다. 이러한 값들이 많기 때문에 이들에 따라 성능이 많이 좌우된다는 단점이 있다. 따라서 지금은 휴리스틱하게 설정된 이러한 설정값들을 어떻게 결정해야 하는지에 대한 방안이 고려되어야 한다. 또한, 종 분류 오토마타는 global match를 고려하는 방식이기 때문에 빈번하게 일어나는 local match에 대한 필요성을 해결해주는 대안이 되지 못한다는 점도 있다. 비단 오토마타 설계, 테스트 상의 문제뿐만 아니라, 전처리 과정인 클러스터링 부분에서 고려해줘야 할 부분도 많다. 일단, 우리가 만들고자 하는 종의 단백질 정보가 단지 서열정보뿐이라고 하면 2차 정보를 이용해서 클러스터링을 할 수 없기 때문에 본 논문에서 제안한 방식을 사용할 수 없다는 한계가 있다. 향후 과제로는 위에서 언급된 문제점들에 대한 처리 및 계층적 분류기에 대한 고려가 있다. 또한 지금 제안한 방식은 종에 한정된 분류기였지만 이 분류기들을 통합해 이보다 상위 계층인 속, 과 등을 분류하는 분류기 제작을 생각해 볼 수 있다. 이를 위해서 지금의 모델을 효율적으로 통합하고 일반화하는 알고리즘을 고안해 볼 필요성이 있다.

참고문헌

[1] B. Boeckmann, A. Bairoch, R. Apweiler, M. C. Blatter, A. Estericher, E. Gasteiger, M. J. Martin, K. Michoud, C. O'Donovan, I. Phan, S. Pilbout and M. Schneider, "The Swiss-Prot Protein Knowledgebase and its supplement TrEMBL in 2003" Journal of Nucleic Acids Research 31, pp.365-370, 2003

[2] W. Taylor, "The Classification of Amino Acid Conservation", Journal of Theoretical Biology 119, pp. 205-218, 1986

[3] K. Yu, "Theoretical Determination of Amino Acid Substitution Groups based on Qualitative Physicochemical Properties", http://cmgm.stanford.edu/biochem218/Projects%202001/Yu.pdf

[4] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, "Optimization by Simulated Annealing", Science 13, pp.671-680, 1983

[5] R. C. Edgar, "MUSCLE: multiple sequence alignment with high accuracy and high throughput", Journal of Nucleic Acids Research 32(5), pp.1792-1797, 2004