**지능형 생물정보학 homework #2**

**노미나, 백은옥 교수님**

2013011424 컴퓨터공학부 컴퓨터전공 성예닮

제출일: 2015.11.28

**목차**

**1. 과제 내용**

**2. 실험 준비**

**3. 데이터 설명**

**4. 실험 과정 및 결과**

**5. 결론**

**- 부록**

**A. 참고 사항**

**B. 사용 코드**

**C. 참고 문헌 및 웹 사이트**

**1. 과제 내용**

We want to predict genes from the bacterial genomic sequences by using HMM.  
- HMM을 사용하여 박테리아의 유전자 서열을 예측하길 원합니다.

As a simple toy example, we will train HMM by using only one strain of *Staphylococcus aureus* and predict genes in another strain of *Staphylococcus epidermidis*  
- 간단한 예제로, 우리는 *Staphylococcus aureus*의 한 스트레인으로부터 HMM 학습을 하고, *Staphylococcus epidermidis*의 다른 스트레인에서의 유전자를 예측할 것입니다.

**2. 실험 준비**

이 실험에서는 주어진 데이터 파일들로부터 파일을 읽고, 데이터를 분석하여 텍스트 파일로 출력해야 합니다. 따라서 파일 입출력에 편리한 python 언어를 사용할 것입니다. 실험을 위한 훈련과 학습 프로그램은 python 3.4.3 버전을 기준으로 작성하였고 IDE는 PyCharm 5.0.1을 사용했습니다.

이번 과제에서는 HMM Viterbi 알고리즘을 사용할 것이고, 여러 order 중 1st order을 사용하여 분석 및 예측을 할 것입니다.

**3. 데이터 설명**

주어진 데이터 파일들은 훈련용 데이터 파일과 학습용 데이터 파일로 나눌 수 있습니다.

훈련용 데이터로 주어진 파일은 FASTA 포맷이고, rain\_genic.fasta, train\_intergenic.fasta 두 파일입니다.

두 파일 모두 형식은 아래와 같습니다.

1>  
ATGTCGGAAAAAGAAATTTGGGAAAAAGTG… 생략  
2>  
…

홀수 줄은 데이터 번호이고, 짝수 줄은 유전자 서열입니다.

유전자 서열의 길이는 제한이 없습니다.

테스트용 데이터로 주어진 파일은 FA 포맷이고, test.fasta 파일입니다. 전체 유전자 서열이 각 줄마다 70자로 분할되어 있습니다.

파일 형식은 다음과 같습니다.

gi|27466918|ref|NC\_004461.1| Staphylococcus epidermidis ATCC 12228 chromosome, complete genome  
AAGAAATTGTGACGCTTATTTGAAGTTATCCACTTATACACATAATTTCTCGCAAAAATTGTGGATAACA  
…  
생략

첫 줄에는 어떤 종류의 데이터인지에 대해 설명하고 있고, 두 번째 줄부터 마지막 줄까지 모두 유전자 서열을 나타내고 있습니다.

**4. 실험 과정 및 결과**

이번 과제에서의 전체적인 과정은 다음과 같습니다.

상태에 대한 부가적인 설명은 부록의 참고 사항-1.Transition State 에 적혀 있습니다.

1. Train 단계

2번 과정의 경우 Emission Probability, 3,4번의 과정의 경우 Transition Probability 입니다.

1.1. genic\_train.fasta, intergenic\_train.fasta 파일을 읽습니다.

1.2. 1st order HMM을 사용할 것이기 때문에 읽어들인 데이터 파일의 유전자를 2글자씩 읽어 각각의 상태 (G = gene, I = intergene, s = start1, S = start2, e = end1, E = end2) 에서의 발현 확률을 계산합니다. (Ex, A 뒤에 T가 나타날 확률 = P (T|A) = P (AT), C 뒤에 G가 나타날 확률 = P (G|C) = P (CG))

주어진 데이터는 가공되었기 때문에 start1과 start2, end1과 end2 상태의 경우 가질 수 있는 상태가 1가지입니다.

start1 = AT, start2 = TG, end1 = TG, end2 = GA 이므로

위의 상태들의 경우 가질 수 있는 상태의 확률을 0.9999985로 두고, 실제로 가질 수 없는 상태들 (start1의 경우 AA, CA, GA, TA등..)에 임의로 아주 작은 확률인 1e^(-7)을 주었습니다.

0으로 처리하지 않은 이유는 실제 계산 시 너무 작은 값들을 계산해야 하기 때문에 로그 값을 취하게 되는데, 로그 값의 경우 0은 계산할 수 없기 때문입니다.

1.3. 읽어들인 데이터로부터 gene의 개수와 총 길이, intergene의 개수와 총 길이를 각각 계산합니다. (순서대로 각각 gene의 개수, 길이, intergene의 개수, 길이 = geneCount, geneLength, intergeneCount, intergeneLength)

1.4. 3번 과정에서 계산한 geneCount, geneLength, intergeneCount, intergeneLength로부터 각각의 상태 전이 확률 (intergenic->intergenic = II, genic->start1 = Gs 등)을 계산합니다.

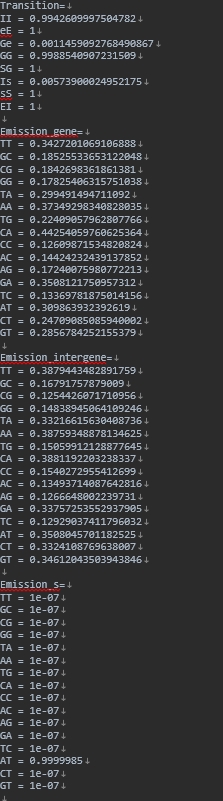
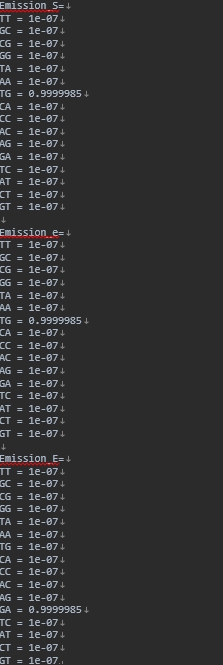
아무것도 고려하지 않는다면 6가지 상태들의 2개의 조합이기 때문에 총 36가지의 가지수가 나올 수 있으나, 없는 상태의 확률을 0 처리하지 않고, 존재하지 않는 상태로 취급했습니다.

따라서 최종적으로 가질 수 있는 Transition은 II, Is, sS, SG, GG, Ge, eE, EI 총 7가지입니다.

Emission Probability의 경우 Gene과 Intergenic 상태에 맞추기 위해 없는 값들을 일부러 임의의 작은 확률을 넣었으나, Transition의 경우 상태 자체를 존재하지 않는 경우로 처리하는 게 유리하기 때문에 이렇게 진행하였습니다.

1.5 위의 단계에서 진행한 상태와 값들을 Model.txt 파일로 출력합니다.

Model.txt 파일은 다음과 같습니다. (다음 장)



2. Test 단계입니다.

2.1 Model.txt 파일을 읽습니다.

2.2 읽어 들인 파일로부터 각각의 Transition, Emission probability를 저장합니다.

2.3 test.fasta 파일을 읽습니다.

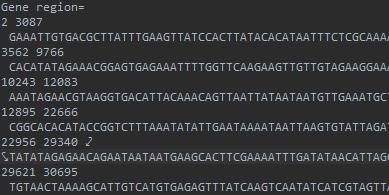
2.4 test.fasta 파일의 경우 2번째 줄부터 전체 유전자를 1줄당 70개씩 저장하고 있습니다. 여러 줄로 되어 있는 데이터 파일을 처리하기 편하도록 1줄로 변경합니다.

2.5 1줄로 처리된 전체 유전자를 읽습니다.

2.6 읽어 들인 전체 유전자를 읽으며 HMM Viterbi를 진행합니다.

2.7 Viterbi 알고리즘을 통해 만들어진 추론된 결과를 Result 파일에 출력합니다.

Result 파일은 다음과 같습니다. (파일이 너무 길어 다음 그림은 일부분입니다.)



**5. 결론**

1st order를 사용하였는데 예측 결과가 썩 좋지 않았습니다. 유전자라는 복잡한 구조물에 대해 여러 가지 상태를 정의하고, 고려해야 하는데 2자리의 유전자만 보고 예측하는 것은 한계가 있다고 보여집니다. 또한 각각의 상태 전이에서 단일 방향을 갖는 부분에서, 다른 방향에 대해서도 작은 확률이나마 처리해주었으면 조금 더 좋은 결과를 볼 수 있었을 것 같습니다.

2nd order나 3rd order 혹은 다른 알고리즘 (각각의 상태 및 위치에 대한 확률로 계산하는 등..)을 사용하면 어떤 결과가 나올 지 궁금해지는 사항입니다. 추후 과제가 아닌 개인적으로 방학 중 이나 다른 때에 한번 해보고 싶습니다.

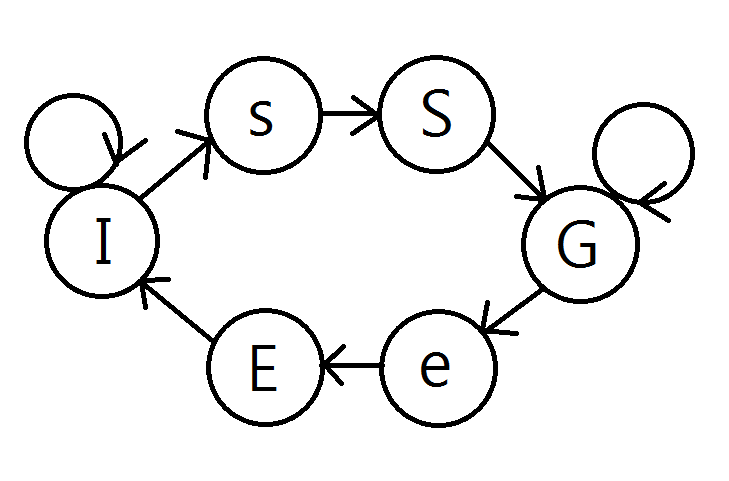
또한 Python의 경우 여러 가지 편리함은 있으나, 컴퓨터의 한계로 C/C++에 비해 아주 느려서 컴파일이 느려 테스팅에 시간을 많이 소요했습니다. 다음 시도에는 C/C++을 이용하여 실험을 해 보고 싶습니다.

생물학적 지식이 없는 (혹은 적은) 전산학도들에게도 생물정보학에 많은 기여를 할 수 있다는 것을 알게 해 주었고 마찬가지로 생물정보학에서 전산학도들이 많이 필요한 이유를 알게 되는 실험이었습니다.

**- 부록**

**A. 참고 사항**

1. Transition State



I == InterGenic

s == Start1 (Almost only AT!)

S == Start2 (Almost only TG)

G == Genic

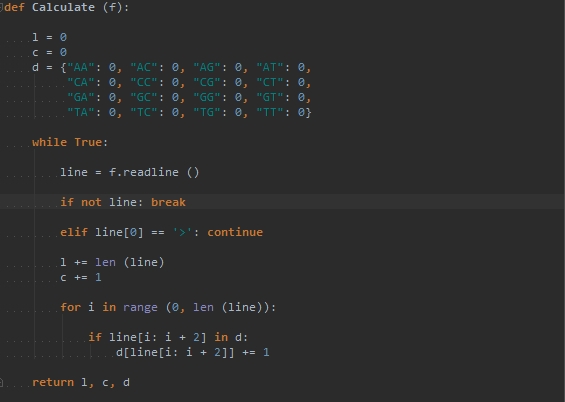
e == End1 (Almost only TG)

E == End1 (Almost only GA)

화살표가 없는 상태에 대해서는 고려하지 않았습니다. (Ex S->s, E->e 등..)

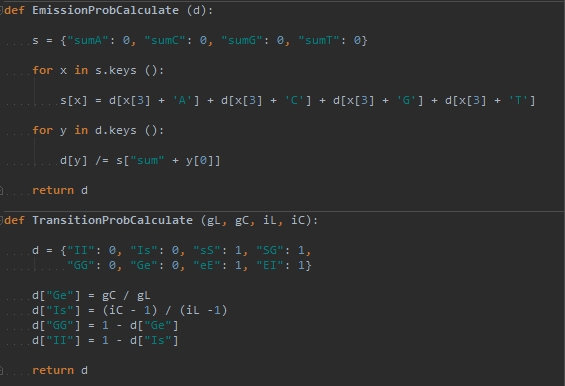
**B. 사용 코드**

train.py



파일을 인자로 받아 파일을 읽고 각각의 서열 (AA, AC, …, TG, TT)가 해당 파일에서 몇 번이나 나타나는지 계산하는 함수입니다.

파일에서의 유효한 (1>, 2> 등을 제외한) 문자열 (유전자의 길이)와, 유효한 문자열의 줄 수 (Genic, Intergenic 구간의 수), AA, AC, .., TG, TT의 발현 횟수를 저장하여 반환합니다.



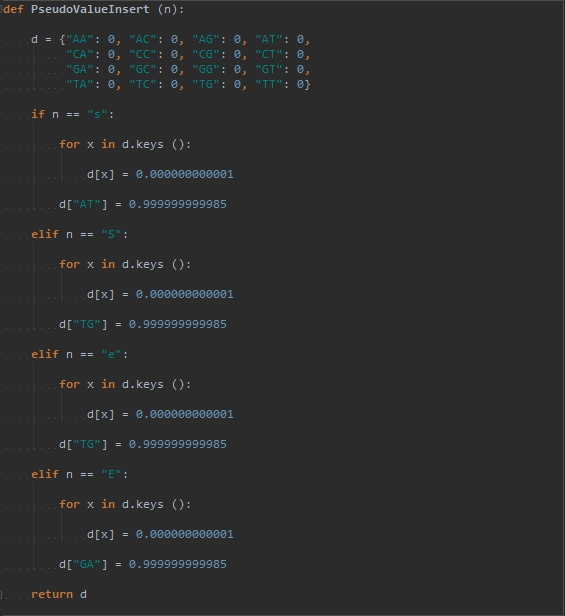
EmissionProbCalculate는

인자로 받은 상태 (G, I) 에서의 Emission Probability(P (A|A), P (A|C), …, P (T|G), P (T|T)) 를 계산하여 반환합니다.

나머지 상태 (s, S, e, E)에 대해서는 뒷장의 PseudoValueInsert 함수에서 계산합니다.

TransitionProbCalculate는

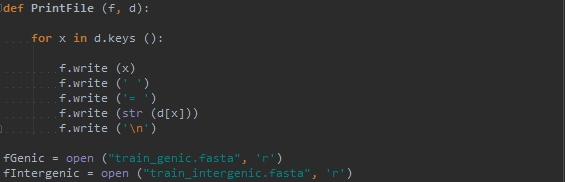
Genic->Genic, Genic->End1 등의 유효한 상태들에 대한 상태 전이 확률을 계산하여 반환합니다.



Start1, Start2, End1, End2의 경우, 실제로는 각각 AT, TG, TG, GA의 상태만 갖습니다.

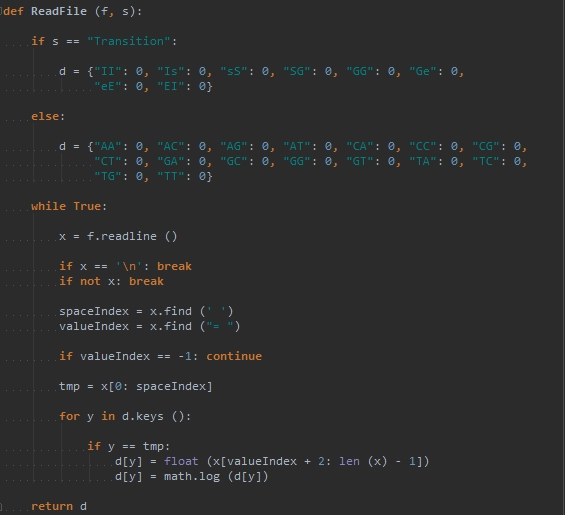
하지만 나머지 상태에 대한 확률을 0으로 처리해버리면, Log 계산시 오류가 나기 때문에 임의의 값 (Pseudo Value)를 설정해주는 함수입니다.

입력받은 상태에서의 Emission Probability를 계산하여 반환합니다.

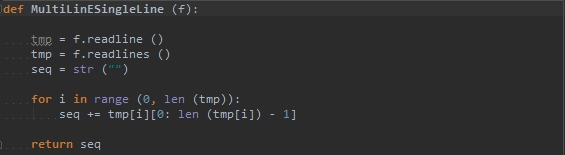


파일과 파일에 적힐 내용을 인자로 받아 파일에 출력합니다.

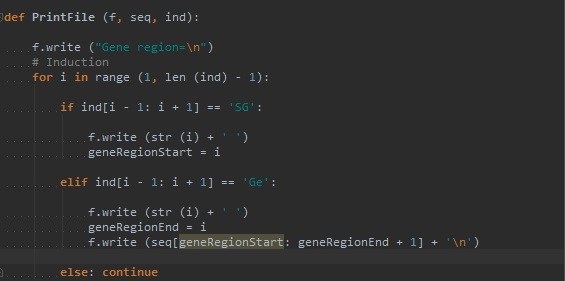
test.py



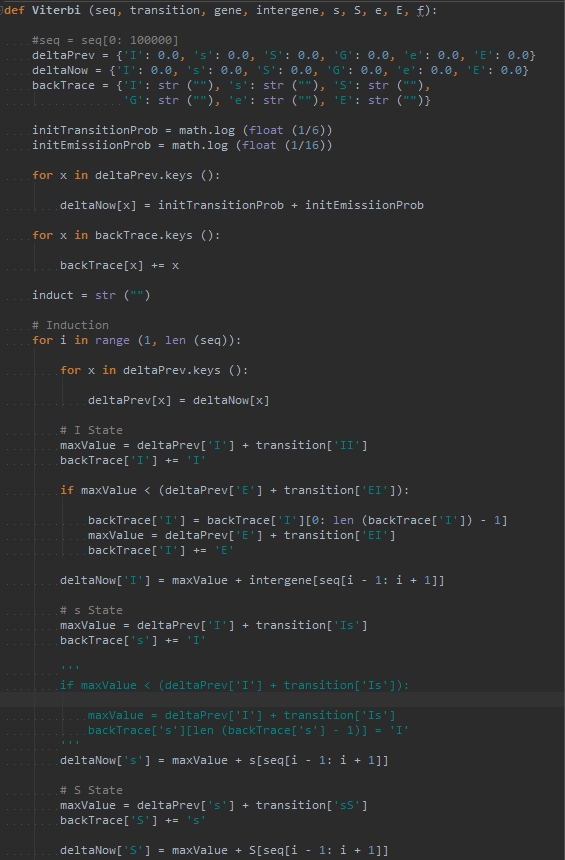
입력 받은 파일로부터 필요한 정보들을 읽고, 확률의 경우 자연로그를 취하여 반환합니다.



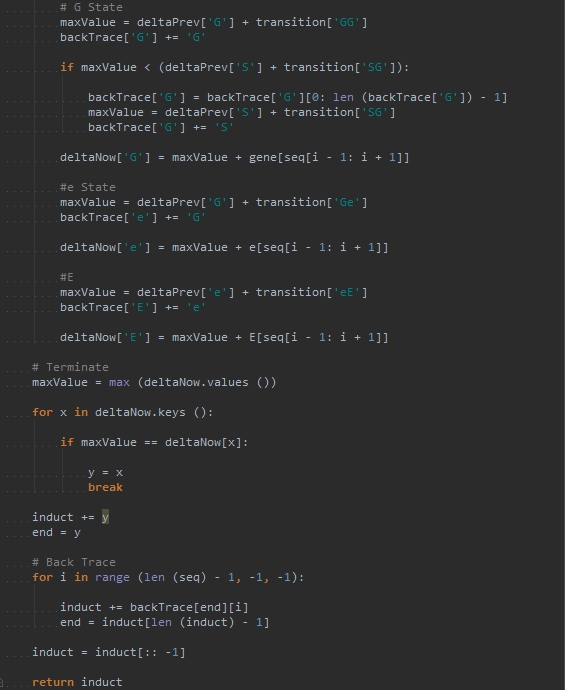
파일이 여러 줄인 경우 한 줄로 합쳐주는 함수입니다.



인자로 받은 내용들을 파일에 출력해주는 함수입니다.



내용이 다음 장에 이어져 있습니다.



Viterbi 함수입니다.

Transition Probability와 Emission Probability를 인자로 받아 계산합니다.

계산 후, Backtrace를 하여 추론 결과를 생성합니다.

추론 과정에서 서열의 뒤쪽부터 읽지만, 추론 결과를 앞에서부터 저장하기 때문에 계산 후 문자열을 뒤집는 연산 induct[:: -1]이 있습니다.

추론된 결과를 반환합니다.

**C. 참고 문헌 및 웹 사이트**

한국어 위키피디아 은닉 마르코프 모델

<https://ko.wikipedia.org/wiki/%EC%9D%80%EB%8B%89_%EB%A7%88%EB%A5%B4%EC%BD%94%ED%94%84_%EB%AA%A8%EB%8D%B8>

영어 위키피디아 Viterbi algorithm

<https://en.wikipedia.org/wiki/Viterbi_algorithm>

HMM: Viterbi algorithm - a toy example (리브르 드 브휵쎌르 대학 수업자료)

<http://homepages.ulb.ac.be/~dgonze/TEACHING/viterbi.pdf>