Reconnaissance de repliement d'Unité Protéique par apprentissage automatique des propriétés des acides aminés de familles alignées

Projet Long

Idée originale de M.Jean-Christophe Gelly

Keywords : alignement, repliement protéique, learning, arbre de décision, AAINDEX

Ghossoub Pierre

M2BI

Université Paris Diderot

08/01/2018

Introduction

Au fur et à mesure que le domaine de la bioinformatique se développe, la possibilité de prédire les comportements structuraux des protéines augmente. Cela permet d'obtenir des résultats avant d'initier des approches expérimentales, coûteuses en temps et en moyen. Ce développement est analogue à l'avancée des techniques à haut débit et à la progressive facilité d'utilisation de techniques algorithmiques comme le machine learning.

Le but de ce projet est le développement d'une méthode de reconnaissance de repliements sur des unités protéiques. Pour ce faire un apprentissage de la validité d'alignements de séquence de ces unités sera effectué : le programme sera capable de reconnaître un bon d'un mauvais alignement.

Le programme nécessitera une base de donnée d'alignements d'unité protéique (Orion) et des propriétés AAINDEX pour caractériser les séquences d'acides aminés des alignements, ce qui sera utile pour l'apprentissage.

Le projet contient les parties suivantes : la génération des alignements positifs et négatifs, le choix des AAINDEX représentatifs avec la transformation des alignements en vecteur AAINDEX, l'apprentissage par arbre de décision et le test de reconnaissance d'alignement étrangers à l'apprentissage.

Matériel et Méthodes

Génération des données

Alignements positifs

Les alignements positifs sont issus de la base de donnée Orion. Ces alignements sont des .fas générés par mafft. Un fichier « /data/posAlignUp25_35.txt » contient les noms des alignements de taille 25 à 35. Sur la même ligne nous avons des alignements présentant des similarités structurales, ce qui constitue ce que l'algorithme d'apprentissage doit repérer.

Alignements négatifs

Il faut générer les alignements négatifs. Ils sont générés par le logiciel mafft à partir de deux fichiers d'alignements issus de deux groupes de similarité différents. L'argument –keeplength permet de garder l'alignement de référence dit « positif » tel quel. L'argument --addfull permet d'enlever les gaps de l'alignement dit « négatif» (puisqu'il n'a pas de similarités structurales avec l'alignement de référence) pour l'utiliser comme nouvelles séquences à aligner. L'argument –thread -1 permet d'utiliser tous les cœurs disponibles de la machine. Puis les premières lignes du fichier sont retirées, puisqu'elles correspondent aux alignements positifs originels.

```
1   >/home/iena/sigaroudi/PDB_PIPELINE/Qsub_3DComb/fragments/1V25A/1V25A_13_289_317//1V25A_13_289_317_frag.pdb
    smoothed
2   ----------ESTRLV---VGSA----A-----------PRSLIARFEREV--RQG
3   >/home/iena/sigaroudi/PDB_PIPELINE/Qsub_3DComb/fragments/1V25A/1V25A_13_289_317//1V25A_13_289_317_frag.pdb
4   ARF-------ERMGVE---VRQG--YGLTETSPV-------VV-------------
5   >/home/wagram/gelly/CORNICHON/PROJECTS/DR/DR_2/PDBs_Stand/1V25A:_|PDBID|CHAIN|SEQUENCE
6   ARF-------ERMGVE---VRQG--YGLTETSPV-------VVQA-----------
7   >tr|A0A022L1P7|A0A022L1P7_9ACTO AMP-dependent synthetase OS=Dietzia sp. UCD-THP GN=H483_0118105 PE=4 SV=1
8   DTS-------TVLSETS--PVEV----|--------------EQ-------------
9   >tr|A0A0A1DPC3|A0A0A1DPC3_NOCSI Pimelobacter simplex strain VKM Ac-2033D, complete genome OS=Nocardioides
    simplex GN=KR76_25280 PE=4 SV=1
10  ------------AGSTD--RLAA-------PEGY-----TYPELSRLVTGSSGLPK
11  >tr|S0ANL1|S0ANL1_FERAC Uncharacterized protein OS=Ferroplasma acidarmanus fer1 GN=FACI_IFERC00001G0352 PE=4
    SV=1
12  --M-------KRASSL---PEKS----G------------SRSITYRKMLKNLAE
13  >tr|A8ZS17|A8ZS17_DESOH AMP-dependent synthetase and ligase OS=Desulfococcus oleovorans (strain DSM 6200 /
    Hxd3) GN=Dole_0225 PE=4 SV=1
14  SSG-------AIMEGYG--LSEM--------SPC-----THLI------------
```

Fig1 : Fichier .Fas d’un alignement négatif.

Il ne reste plus que des alignements de séquences qui ont peu de similarités entre elles. Les deux types d’alignements sont identifiés en tant que tel, ce qui constitue en la base de l’apprentissage.

Profil AAINDEX

Nous avons à disposition 545 propriétés AAINDEX, dans un fichier « selected_aaindex1_reformated ». A chaque propriété est associé une valeur pour chaque acide aminé. Dans notre projet, 58 propriétés AAINDEX sont extraites, car elles sont jugées plus pertinentes par l'étude de Van Westen et al. Le nouveau fichier est « selected_AAINDEX58_reformated ».

Le programme « fasta2 vector_wgap.pl » utilise un fichier de propriété AAINDEX tel que celui décrit ci-dessus pour transformer un alignement .fas en vecteur de propriétés AAINDEX. Chaque ligne correspond à une séquence, et pour chaque lettre les valeurs des 58 propriétés sont concaténées. Chaque vecteur pourra être utilisé pour la génération de l'arbre de décision.

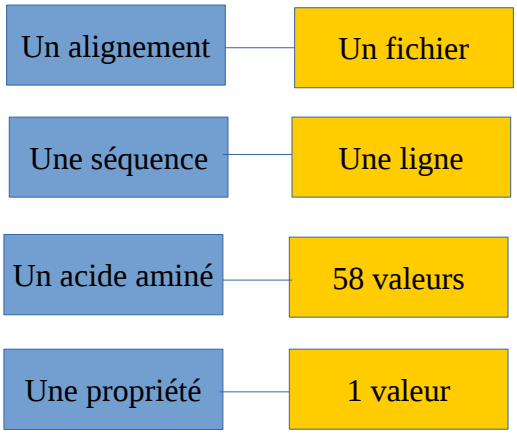

Fig2 : Schéma d'explication du contenu d'un vecteur aaindex.

```
#ID A R N D C Q E G H I L K M F P S T W Y V
ARGP820103 1.56 0.45 0.27 0.14 1.23 0.51 0.23 0.62 0.29 1.67 2.93 0.15 2.96 2.03 0.76 0.81 0.91 1.08 0.68 1.14
BHAR880101 0.357 0.529 0.463 0.511 0.346 0.493 0.497 0.544 0.323 0.462 0.365 0.466 0.295 0.314 0.509 0.507 0.
444 0.305 0.420 0.386
CHAM810101 0.52 0.68 0.76 0.76 0.62 0.68 0.68 0.00 0.70 1.02 0.98 0.68 0.78 0.70 0.36 0.53 0.50 0.70 0.70 0.76
CHAM820101 0.046 0.291 0.134 0.105 0.128 0.180 0.151 0.000 0.230 0.186 0.186 0.219 0.221 0.290 0.131 0.062 0.
108 0.409 0.298 0.140
CHAM830101 0.71 1.06 1.37 1.21 1.19 0.87 0.84 1.52 1.07 0.66 0.69 0.99 0.59 0.71 1.61 1.34 1.08 0.76 1.07 0.63
CHAM830107 0. 0. 1. 1. 0. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
CHAM830108 0. 1. 1. 0. 1. 1. 0. 0. 1. 0. 0. 1. 1. 1. 0. 0. 0. 1. 1. 0.
CHOP780201 1.42 0.98 0.67 1.01 0.70 1.11 1.51 0.57 1.00 1.08 1.21 1.16 1.45 1.13 0.57 0.77 0.83 1.08 0.69 1.06
CHOP780202 0.83 0.93 0.89 0.54 1.19 1.10 0.37 0.75 0.87 1.60 1.30 0.74 1.05 1.38 0.55 0.75 1.19 1.37 1.47 1.70
CHOP780203 0.74 1.01 1.46 1.52 0.96 0.96 0.95 1.56 0.95 0.47 0.50 1.19 0.60 0.66 1.56 1.43 0.98 0.60 1.14 0.59
CIDH920105 0.02 -0.42 -0.77 -1.04 0.77 -1.10 -1.14 -0.80 0.26 1.81 1.14 -0.41 1.00 1.35 -0.09 -0.97 -0.77 1.
71 1.11 1.13
FASG760101 89.09 174.20 132.12 133.10 121.15 146.15 147.13 75.07 155.16 131.17 131.17 146.19 149.21 165.19 115
```

Fig3 : Fichier de génération des vecteur AAINDEX à 58 propriétés.

Apprentissage par arbre de décision

Le but est que le programme sache à quoi « ressemble » un bon alignement et un mauvais alignement. Il sait déjà quels sont les bons et les mauvais alignements parmi ce qu'on lui a donné, et cette connaissance doit lui servir à classifier des alignements étrangers. Les 58 propriétés AAINDEX servent alors de descripteurs. En fonction de quelles valeurs sont présentes dans les alignements bons et mauvais, un pattern sera identifié par le programme.

Chaque descripteur est une variable prédictive : pour chaque, les valeurs des deux types d'alignements sont comparés. On peut croiser les différentes variables entre elles jusqu'à qu'on puisse déterminer une limite pour chaque valeur de descripteur à laquelle on passe d'un bon à un mauvais alignement. Ainsi les deux types d'alignements sont discriminés dans leurs valeurs de propriétés.

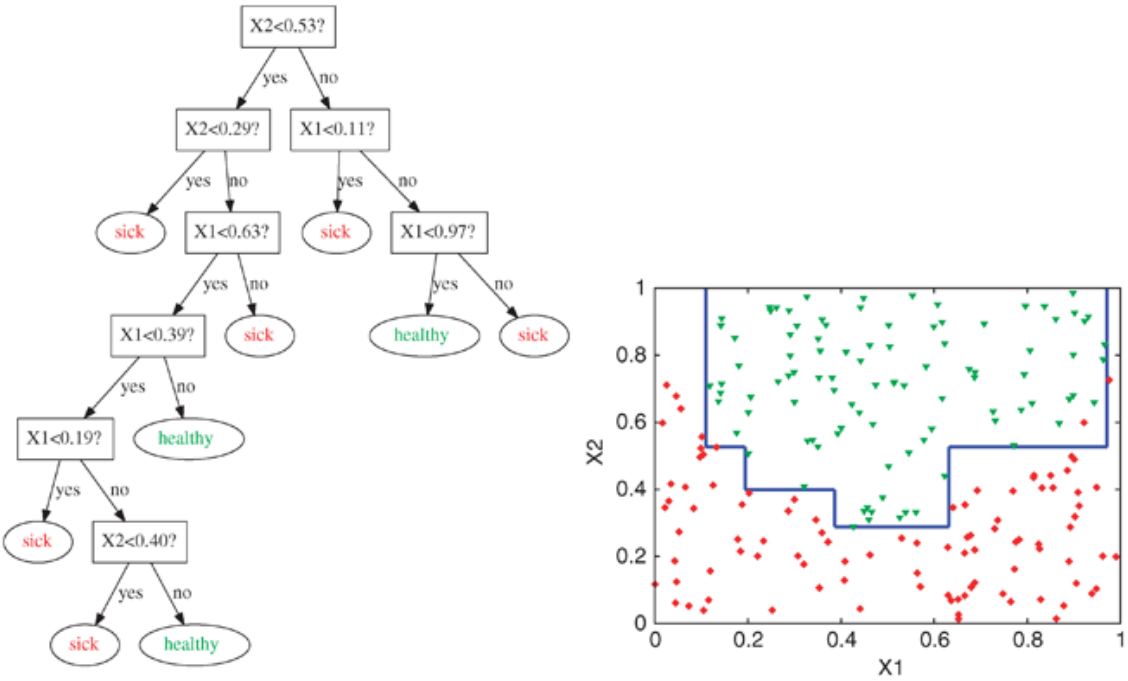

Fig4 : Exemple d'arbre de décision pour apprentissage. Chaque nœud est une décision selon un descripteur. A droite, représentation des valeurs discriminés entre les descripteurs X1 et X2.

(*Supervised learning with decision tree-based methods in computational and systems biology. Pierre Geurts,* Alexandre Irrthum and Louis Wehenkel. Received 21st April 2009, Accepted 8th September 2009 First published as an Advance Article on the web 5th October 2009 DOI: 10.1039/b907946g*)

Mais cela ne garantit pas la bonne classification d'alignement étrangers à l'apprentissage. Il faudra tester la décision de notre algorithme avec des alignements positifs étrangers à l'apprentissage, et calculer le taux de vrais positifs, de faux positifs, de vrais négatifs et de faux négatifs. En fonction, il est envisageable de continuer l'apprentissage.

Discussion / Conclusion

Dans cette partie, nous allons discuter de ce qui a été fait lors de ce projet, puisque le cahier des charges et les stratégies mises en place personnellement n'ont pas toutes été appliquées.

Le programme peut, à partir d'une base de donnée Orion en local, générer les alignements positifs et négatifs en utilisant mafft sélectionnées par leur nom dans le fichier « posAlignUp25-35.txt » et par les lignes que désire l'utilisateur (paramètres) même si la base de donnée est lacunaire, extraire les propriétés AAINDEX et transformer les alignements en vecteur de propriétés AAINDEX. Les alignements sont séparés dans l'arborescence de fichier. Il prépare donc le terrain pour l'apprentissage par arbre de décision, qui aurait été fait avec le module scikit-learn.

Une fois le module scikit-learn maîtrisé, l'apprentissage aurait été lancé, avec comme sortie un fichier décrivant la prédiction du type d'alignement, lisible par le programme pour chaque type d'alignement par exemple. Ensuite, un autre script aurait été rédigé pour pouvoir tester le programme après apprentissage. Sélection des alignements à tester, option pour continuer l'apprentissage du programme et affichage des résultats de prédiction sous forme de graphiques sont la suite des objectifs du projet.

Une gestion d'erreurs et une manière intelligente de gérer les lacunes de la base de données sont les amélioration les plus évidentes relatives aux parties déjà programmées.

Pour conclure, même si le projet n'est pas terminé, la détermination des objectifs principaux et détaillés, et la mise en place des premières parties peuvent servir de bases solides à l'élaboration du projet complet. Une vingtaine de profils AAINDEX négatifs pouvant être générés en une heure, cette méthode permet de remplir la base de données d'alignement propres au programme avec une certaine rapidité, ce qui nous conforme dans le choix de l'utilisation de cette méthode. Pour ma part, le projet m'a apporté une pratique plus poussée du python pour la gestion d'une multitude de fichiers et des connaissances théoriques en machine-learning.