

Rapport de fouille de données

Contexte

Les opérons interviennent dans la régulation de l'expression des gènes chez les bactéries. Ils comprennent des gènes structuraux et des éléments de contrôle dans l'ADN, reconnus par des produits de gènes régulateurs. La connaissance des gènes appartenant à un même opéron est donc une étape importante dans la compréhension du fonctionnement des organismes. La disponibilité de génomes complets et l'accumulation de données et de connaissances sur un certain nombre d'organismes permettent la mise en œuvre de méthodes pour la prédiction de l'appartenance des gènes à un même opéron.

Notre équipe composée de Volle Arnaud, Chazalviel Maxime et Heurteau Alexandre, s'est concentrée sur l'étude d'une souche d'*Escherichia coli* K12 (MG1655). Les nombreuses recherches et publications à son sujet font de cette bactérie une référence quant à la disponibilité des informations génomique et génétique ce qui a facilité l'élaboration et surtout la pertinence de la méthode de prédiction réalisée.

Choix et analyses

Attributs sélectionnés

Afin de prédire les opérons présents dans cet organisme, l'équipe a commencé par sélectionner les attributs génétiques estimés comme étant les plus pertinents. La littérature indique que des données telles que le brin, le sens de transcription ou encore la distance inter-génique sont de bons indicateurs concernant la position et la constitution des opérons (H. Salgado, G. Moreno-Hagelsieb, T. Smith and J. Collado-Vides. Operons in *Escherichia coli*: *Genomic analyses and predictions*. PNAS 2000.).

Une publication plus récente « High accuracy Operon prediction method bases on STRING database scores, B. Taboada et al » indique que l'utilisation de données issues de la banque de données Search Tool for the Retrieval of Interacting Genes/Proteins¹ (STRING) permet d'améliorer significativement la pertinence des résultats. En effet, cette banque de données regroupe des informations sur l'interaction des protéines connues et prédites en s'appuyant sur des sources dérivées telles que la co-expression, le contexte génomique et les données de

Gene Ontology entre autres. Le score obtenu est la fusion de toutes ces données. Il est compris entre 0 et 1, 0 signifiant une absence totale de lien entre les protéines.

De ce fait, nous avons décidé de nous appuyer sur la distance inter-génique (au sein d'un opéron ainsi qu'entre les unités de transcription considérées comme non-opéronique), sur le brin ainsi que sur les données issues de la banque de données STRING. Nous avons aussi rajouté un facteur hypothétiquement discriminant à savoir la somme de la taille de deux gènes contigus au sein d'un même opéron.

Méthodes

Trois méthodes de fouille de données ont été employées afin de les comparer : Un classificateur bayésien naïf, un arbre de décision ainsi qu'un réseau de neurones.

Données sources utilisées. Transformations effectuées.

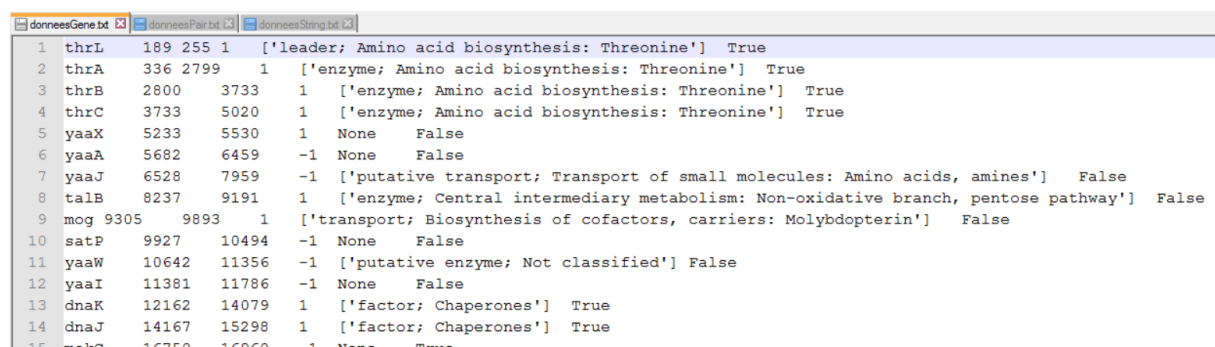
Description des données utilisées

Afin d'utiliser les différents classificateurs, il a été nécessaire de sélectionner plusieurs fichiers comportant les informations qui nous seront utiles.

Premièrement, un fichier de la banque de données GenBank nous fournit le génome d'*Escherichia coli* K12 (MG1655) avec plusieurs informations comme le nom de chaque gène, leur position sur le génome et leurs fonctions. Et ensuite, un fichier provenant de RegulonDB, qui nous apporte des informations sur l'organisation et la régulation des gènes d'*Escherichia coli*, avec les opérons présents sur le génome.

Après avoir choisi les fichiers à traiter, il a fallu réaliser plusieurs scripts en langage Python pour parser les données.

Un premier script nous a permis de créer un fichier contenant la liste des gènes présents dans les fichiers GenBank et RegulonDB. Les informations récupérées pour chaque gène sont leur nom, leur position sur le génome, leur brin, leur fonction et le fait qu'ils appartiennent ou non à un opéron.



1	thrL	189 255	1	['leader; Amino acid biosynthesis: Threonine']	True
2	thrA	336 2799	1	['enzyme; Amino acid biosynthesis: Threonine']	True
3	thrB	2800 3733	1	['enzyme; Amino acid biosynthesis: Threonine']	True
4	thrC	3733 5020	1	['enzyme; Amino acid biosynthesis: Threonine']	True
5	yaaX	5233 5530	1	None	False
6	yaaA	5682 6459	-1	None	False
7	yaaJ	6528 7959	-1	['putative transport; Transport of small molecules: Amino acids, amines']	False
8	talB	8237 9191	1	['enzyme; Central intermediary metabolism: Non-oxidative branch, pentose pathway']	False
9	mog	9305 9893	1	['transport; Biosynthesis of cofactors, carriers: Molybdopterin']	False
10	satP	9927 10494	-1	None	False
11	yaaW	10642 11356	-1	['putative enzyme; Not classified']	False
12	yaaI	11381 11786	-1	None	False
13	dnaK	12162 14079	1	['factor; Chaperones']	True
14	dnaJ	14167 15298	1	['factor; Chaperones']	True
15	makC	16750 16960	-1	None	True

Fichier généré par le premier script.

Le deuxième script nous permet de récupérer le nom de tous les gènes présents dans le fichier précédemment créé. Ce fichier a été utilisé pour lancer StringDB qui fournit pour chaque paire de gènes un score. Celui-ci est obtenu par la prédiction de l'interaction entre les protéines.

1	ilvH	leuO	4735305	4735303	b0078	b0076	0.425	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.425
2	cysS	tsf	4735710	4735393	b0526	b0170	0.307	0.000	0.000	0.000	0.000	0.000	0.000	0.491	0.624
3	proC	frr	4735579	4735395	b0386	b0172	0.173	0.000	0.000	0.000	0.000	0.000	0.000	0.368	0.443
4	recR	pdxA	4735662	4735281	b0472	b0052	0.102	0.000	0.000	0.000	0.000	0.000	0.000	0.741	0.752
5	secA	ribF	4735324	4735254	b0098	b0025	0.129	0.000	0.000	0.000	0.000	0.000	0.000	0.454	0.493
6	fabZ	glnD	4735403	4735390	b0180	b0167	0.273	0.000	0.000	0.000	0.000	0.000	0.000	0.470	0.589
7	hemH	lspA	4735665	4735256	b0475	b0027	0.500	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.500
8	mltD	carA	4735428	4735261	b0211	b0032	0.266	0.000	0.000	0.000	0.000	0.000	0.000	0.260	0.421
9	ecpD	yadM	4735364	4735362	b0140	b0138	0.677	0.000	0.744	0.000	0.000	0.000	0.000	0.588	0.961
10	ftsZ	ddlB	4735321	4735318	b0095	b0092	0.927	0.000	0.000	0.000	0.000	0.000	0.000	0.939	0.995

Fichier créé par StringDB.

Le troisième script nous a permis de créer un fichier qui puisse être utilisé par les différents classificateurs. Nous avons décidé de traiter les gènes par paires afin de savoir si deux gènes appartenaient ou non au même opéron. Ce script se sert donc du fichier créé par notre premier script pour faire des paires de gènes et utilise aussi le fichier créé par StringDB. Il crée un fichier, avec sur chaque ligne des attributs qui permettent probablement la prédiction des opérons. Ces attributs sont le nom des deux gènes concaténés, la distance entre ces deux gènes, leur deux tailles additionnées, leur brin et le fait qu'ils appartiennent ou non au même opéron. Il est à noter que les gènes d'une paire sont forcément sur le même brin.

1	Nom	Distance	TailleDeuxGene	Brin	Positif?	Score de string	Opéron
2	thrL thrA	81	2529	True	0.639	True	
3	thrA thrB	1	3396	True	0.999	True	
4	thrB thrC	0	2220	True	0.999	True	
5	thrC yaaX	213	1584	True	0.523	False	
6	yaaA yaaJ	69	2208	False	0.813	False	
7	talB mog	114	1542	True	0.563	False	
8	satP yaaW	148	1281	False	0.643400260580442	False	False
9	yaaW yaaT	25	1119	False	0.964	False	

Fichier généré par le dernier script.

Réalisation

Afin d'extraire les informations les plus pertinentes, trois classificateurs ont été utilisés : Un classificateur bayésien, un arbre de décision ainsi qu'un réseau de neurones.

Les tests ont été effectués au travers du logiciel Konstanz Information Miner (KNIME). KNIME permet, au travers de son interface graphique ergonomique, de performer une multitude de tests sur des jeux de données variés et d'en extraire les connaissances. Plusieurs

classificateurs sont mis à disposition dont le classificateur Bayésien, l'arbre de décision et le réseau de neurones.

L'apprentissage par la méthode bayésienne se base sur le calcul de la probabilité des hypothèses. Afin de déterminer la classe d'un objet, il faut déterminer la probabilité qu'a cet objet d'appartenir à toutes les classes possibles. La classe de l'objet étant finalement la classe ayant obtenue la probabilité la plus forte. Ceci se modélise par :

$P(C/X)$: Probabilité qu'un objet $X \langle x_1, \dots, x_n \rangle$ soit de classe C .

Il est à noter que les probabilités calculées pour chaque attribut sont considérées comme indépendantes.

La méthode d'apprentissage par arbre de décision est basée sur un algorithme glouton récursif. Les données sont représentées sous la forme d'un arbre dont les nœuds représentent un test sur les attributs et les branches la valeur de l'attribut.

Les exemples à classer sont partitionnés récursivement par sélection d'un attribut sélectionnés par une heuristique ou des statistiques. L'algorithme termine lorsqu'il n'y a plus d'exemples à classer ou lorsque qu'il n'existe plus d'attributs à partitionner (la classe attribuée correspond donc à la classe la plus représentée dans l'arbre).

Finalement, la méthode du réseau de neurones se base sur l'obtention d'un ensemble de poids permettant de classer correctement la quasi-totalité des objets du jeu d'apprentissage. Pour cela, il faut d'abord initialiser les poids avec des nombres aléatoires avant de passer les objets au réseau (un par un). Pour chaque neurone, l'entrée est calculée comme la combinaison linéaire de toutes les entrées (vecteur entrée X) puis la sortie est calculée en utilisant le vecteur poids et la fonction d'activation qui prend en entrée la somme pondérée du vecteur entrée X et du vecteur poids. Enfin, le calcul de l'erreur permet de mettre à jour le biais et donc les poids.

Le jeu de données contient 2885 paires de gènes (opéroniques et non-opéroniques).

Résultats obtenus

Voici les résultats obtenus en fonction des classificateurs :

Utilisation pour tous les classificateurs de la méthode de partitionnement leave one out qui permet un apprentissage sur $(n-1)$ observations ou individus puis on valide le modèle sur la n -ième observation et l'on répète cette opération n fois.

Classificateur bayésien

En prenant en compte la distance intergénique, le brin et la taille des gènes, le classificateur nous renvoie un taux d'erreur de 33,75 %.

Confusion Matrix - 0:14 - Scorer		
File	Hilite	
Operon \ ...	True	False
True	1785	58
False	916	127
Correct classified: 1 912		Wrong classified: 974
Accuracy: 66,251 %		Error: 33,749 %
Cohen's kappa (κ) 0,11		

L'ajout du string score permet de le faire chuter à 23,25 %

Confusion Matrix - 0:14 - Scorer		
File	Hilite	
Operon \ ...	True	False
True	1418	425
False	246	797
Correct classified: 2 215		Wrong classified: 671
Accuracy: 76,75 %		Error: 23,25 %
Cohen's kappa (κ) 0,514		

Sensibilité : $1418/1624 = 0,873$

Spécificité : $797/1222 = 0,652$

Précision : $1418/1843 = 0,769$

Exactitude : sensibilité+ spécificité= 1,525

Arbre de decision

Paramètres : indice de gini, méthode d'élagage : MLD (maximum likelihood detection)
taux d'erreur de 16,6 %

Confusion Matrix - 0:12 - Scorer		
File	Hilite	
Operon \ ...	False	True
False	891	152
True	328	1515
Correct classified: 2 406		Wrong classified: 480
Accuracy: 83,368 %		Error: 16,632 %
Cohen's kappa (κ) 0,652		

Sensibilité : $1515/1667 = 0,909$

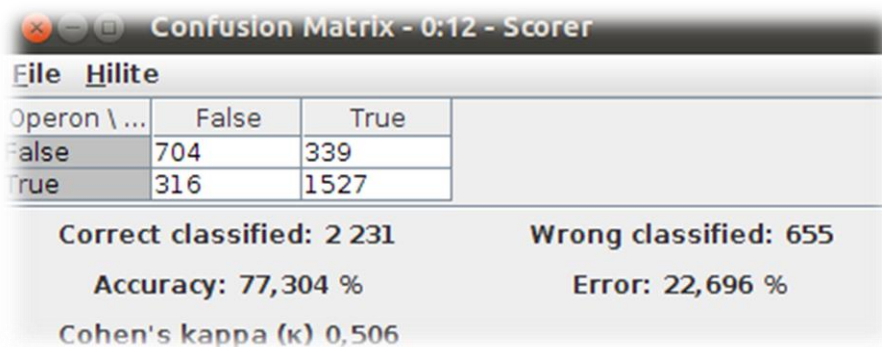
Spécificité : $891/1219 = 0,731$

Précision : $1515/1843 = 0,822$

Exactitude : sensibilité + spécificité = 1,64

Réseau de neurones PNN (Probabilistic Neural Network)

Nécessite une normalisation des données, taux d'erreur 17,25 %



Operon \ ...	False	True
False	704	339
True	316	1527

Correct classified: 2 231 **Wrong classified: 655**
Accuracy: 77,304 % **Error: 22,696 %**
Cohen's kappa (κ) 0,506

Sensibilité : $1580/1815 = 0,871$

Spécificité : $808/1071 = 0,754$

Précision : $1580/1843 = 0,857$

Exactitude : sensibilité + spécificité = 1,625

Bilan et perspectives

Comme le montrent les résultats, le classificateur le plus performant s'avère être l'arbre de décision avec une précision de prédiction de 83.4% (contre 76.75% pour le classificateur bayésien et 82.75% pour le réseau de neurones). En comparaison des nouvelles méthodes de prédictions d'opérons avoisinant les 98% de précision, ces résultats sont forts peu probants. Cependant, cette étude nous a permis de souligner la performance des données issues de la banque de données STRING. Elle a également permis de minimiser l'importance de certains facteurs tels que le brin ou le sens de lecture.

Le sujet de ce projet est des plus intéressants puisqu'il nous a permis de rechercher des informations sur des publications et banques de données variées, et nous avons pu mettre en place tous les outils nécessaires à la prédiction de manière autonome.

De plus, nous avons réussi à pallier diverses difficultés liées à l'utilisation d'un nouveau langage tel que Python par exemple ou bien encore la limitation de nos connaissances sur un sujet biologique particulier.

La prédiction d'opéron mérite d'être développée de manière plus approfondie. En effet, il existe un grand nombre de données susceptibles d'apporter des précisions quant à la prédiction des opérons dans un organisme. Parmi celle-ci, il serait possible de prendre en compte les différents promoteurs et terminateurs de chaque opéron et de chaque unité de transcription. De même, la fonction des gènes est un précieux indicateur puisque les gènes d'un même opéron ont beaucoup plus de chances d'avoir la même fonction métabolique que des gènes sélectionnés de manière aléatoire. Nous aurions pu les utiliser puisque nous avons récupéré ces données via un script mais le temps nous a manqué pour approfondir et mener à bien cette étape.

Enfin, il pourrait être intéressant d'approfondir les informations provenant de la banque de données STRING (Données COG, MOG...). La littérature montre en effet que la précision de la prédiction peut être augmentée de manière étonnante par le biais de la normalisation de tels scores.

Répartition des tâches

Arnaud Volle :

Au début du projet, nous avons tous les trois participé à la récupération des données. En parallèle, je me suis intéressé à la partie bibliographique avec Alexandre ce qui nous a permis de sélectionner les attributs qui nous ont semblés les plus pertinents.

Nous avons ensuite rejoint Maxime alors qu'il continuait la réalisation des Scripts. Finalement, j'ai mis en place les différents jeux de paramètres pour la comparaison des classificateurs via le logiciel KNIME.

Maxime Chazalviel :

Après avoir participé au début de la récupération des données, je me suis penché sur la partie qui consistait à réaliser les scripts pour formater les données. Arnaud et Alexandre ont ensuite réalisé certaines fonctions de parsing en parallèle. Nous nous sommes mis d'accord sur les classificateurs et Arnaud a réalisé la prédiction via KNIME pendant que nous commençons ce rapport.

Alexandre Heurteau :

Comme l'ont dit Arnaud et Maxime, nous avons commencé par rechercher et récupérer les données d'intérêts. Nous avons ensuite lu quelques publications et autres documents relatifs à la prédiction d'opérons. Arnaud et moi-même avons aussi réalisé une partie des scripts de parsing pour faire avancer Maxime. La répartition des tâches s'est faite naturellement mais cela ne nous a pas empêchés de comprendre et de participer à toutes les étapes du projet.

Bibliographie

¹ [Franceschini et al. 2013](#)