Prédiction du site de clivage

- Changqing LIU
-Biao SHI

May 19, 2022

1 Introduction

Dans notre projet, nous considérons le problème de la prédiction de la position du site de clivage dans les protéines en fonction d'un ensemble d'apprentissage téléchargé vers le lien https://www.lri.fr/~nowak/teaching/cleavage/GRAM+SIG_13.red. Ici un protéine est identifié comme sa séquence primaire, c'est-à-dire, le séquence de ses acides aminés. Chaque acide aminé est codé par une lettre.

On admet que le site de clivage peut être caractérisé par des acides aminés dans son voisin défini comme p acides aminés avant et q après. Par la suite, nous allons classifier le site de clivage dans un morceau de longueur de $p+q$ à l'aide d'apprentissage supervisé. En particulier, nous allons améliorer la précision avec quelques noyeau spécifiques de SVM.

2 Pré-traitement des données

Pour réduire l'ambiguïté, nous définissons les données utilisées comme suit.

- **Séquence d'acides aminés** : la séquence d'acides aminés obtenue directement de la base de données. Son premier acide aminé doit être M, sa longueur est incertaine, et il existe un et un seul site de clivage par séquence d'acides aminés.

  Par exemple : la séquence (1) `MARFSIVFAAAGVLLLVAMAPVSEASTTTIIITTIIEENPYGRGRTESGCYQQMEE` a un site de clivage à la position 25 (entre `A` et `S`).

- **Fragment d'acide aminé** : fragment découpé d'une séquence d'acides aminés. Sa longueur est de $p+q$, avec p et q étant les paramètres que nous déterminons. Si le site à la position p de ce fragment est exactement le site de clivage, nous considérons qu'il s'agit d'un fragment positif, sinon il est négatif.

  Par exemple, lorsque $p=13$ et $q=2$, `LLLVAMAPVSEASTT` coupé par la séquence (1) est un fragment positif car le site de clivage (entre `A` et `S`) est à la position $p=13$. Et `VLLLVAMAPVSEAST` coupé par la séquence (1) est un fragment négatif car la position $p=13$ (entre `E` et `A`) n'est pas un site de clivage.

Notre tâche consiste à déterminer si le fragment d'acide aminé d'entrée est un fragment positif, ce qui est en fait une tâche classique de classification binaire. Dans notre projet, nous avons principalement utilisé le jeu de données contenant 140 séquences de procaryotes Gram-positifs. Nous utilisons d'abord 80% des séquences d'acides aminés pour l'entraînement de notre modèle et les séquences d'acides aminés restantes pour le test de notre modèle. Si nous coupons avec $p=13$ et $q=2$, il y a 112 fragments positifs et 5195 fragments négatifs dans l'ensemble d'apprentissage et 28 fragments positifs, 1327 fragments négatifs dans l'ensemble de test. Il est intéressant de noter que les fragments négatifs sont beaucoup plus grands que les fragments positifs.

3 Approches différentes

3.1 Modèle statistique

Nous avons d'abord essayé d'appliquer un modèle statistique utilisant une matrice de position. Dans ce modèle, les fréquences des acides aminés sont utilisées pour estimer leur distribution de probabilité,

puis un score est défini pour déterminer si le fragment est positif ou non. Pour éviter l'effet de 0 sur le logarithme pris dans les étapes ultérieures, nous avons utilisé des pseudo-comptes.

La fréquence observée de l'acide aminé a à la position relative i est défini comme suivant :

$$f(a,i) = \frac{c(a,i) + \alpha}{N + \alpha},$$

où N est le nombre total de séquences dans l'ensemble d'apprentissage, $c(a,i)$ est le nombre d'occurrences de l'acide aminé $a \in A$, à la position $i \in \{-p, ..., q-1\}$, par rapport au site de clivage et α est une constant de pseudo-compte.

La fréquence de fond observée de l'acide aminé a est défini comme suivant :

$$g(a) = \frac{n(a) + \alpha}{T + \alpha},$$

où $n(a)$ est le nombre d'occurrences de l'acide aminé a dans toutes les séquences, T est le nombre total des acides aminés dans l'ensemble d'apprentissage (à l'exception du premier acide aminé 'M' de la séquence).

Nous définissons un score comme suivant :

$$\sum_{i=-p}^{q-1} s(a_{p+i}, i) = \sum_{i=-p}^{q-1} [\log(f(a,i)) - \log(g(a))]$$

Si le score d'un fragment d'acides aminés dépasse un certain seuil, nous le jugeons positif.

Dans notre projet, nous avons calculé les scores moyen, maximum et minimum des fragments d'acides aminés positifs dans l'ensemble d'apprentissage et les avons testés comme un seuil dans l'ensemble de test. Nous avons constaté que le score moyen fonctionnait mieux, donc dans l'étape suivante nous l'avons défini comme seuil.

3.2 Machine à vecteurs de support (SVM) du noyau RBF de Gaussien

Après le onehot encoding de chaque fragment d'acide aminé, nous pouvons obtenir un vecteur de $\mathbb{R}^{26(p+q)}$. Avec ce vecteur, nous pouvons utiliser les noyaux SVM populaires : noyau linéaire, polynômial et RBF de Gaussien. Dans ce projet, nous utilisons la bibliothèque libsvm, qui nous oblige à stocker les données au format .svm. Ce codage nous permet d'ignorer les zéros dans les vecteurs dans l'étape de stockage, ce qui permet de gagner de l'espace et d'augmenter la vitesse des opérations. Particulièrement, nous avons utilisé le noyau RBF de Gaussien et obtenu de bons résultats.

3.3 SVM du noyau en fonction du score de similarité

Nous pouvons définir la simlarité entre les acids aminés en utilisant la matrice de substitution, c'est-à-dire, l'entrée $M(x,y)$ induit le score de similarité de la paire (x,y). Par la suite, nous définissons la similarité entre deux mots de longueur n $a = a_0...a_{n-1}$ et $b = b_0...b_{n-1}$ comme :

$$S(a,b) = \sum_{i=0}^{n-1} M(a_i, b_i)$$

Nous pouvons considérer la fonction S comme un pseudo-produit scalaire, d'où nous pouvons définir un noyeau pseudo-linéaire.

Notons que ce pseudo-produit scalaire ne suffit pas la condition de Mercer, mais on peut quand même définir un noyeau pseudo-RBF en fonction de la similarité :

$$K(a,b) = \exp(-\gamma |x-y|^2)$$

3.4 SVM du noyau probabiliste

Nous avons également essayé un noyau probabiliste basé sur un modèle statistique, qui est défini comme suit. Pour deux fragments d'acides aminés a and b,

$$\log K(a,b) = \sum_{i=-p}^{q-1} \phi_i(a_{p+i}, b_{p+i})$$

$$\phi_i(x,y) = \begin{cases} s(x,i) + s(y,i) & \text{si } x \neq y \\ s(x,i) + \log(1 + e^{s(x,i)}) & \text{si } x = y \end{cases}$$

où $s(x,i)$ est identique à la définition mentionnée ci-dessus. Afin d'éviter la duplication des calculs, la matrice des noyaux est calculée dans le programme principal et stockée dans le format spécifié par libsvm, puis passe comme entrée au programme svm-train de libsvm.

La première difficulté que nous avons rencontrée est le grand nombre de permutations possibles d'acides aminés, avec 26^{15} permutations possibles pour un fragment d'acides aminés de longueur 15. Il rend la probabilité d'une permutation d'acide aminé particulière très faible. Pour le jeu de données utilisé, nous avons calculé que $\log K(a,b)$ était d'environ -200, ce qui signifie que la taille de $K(a,b)$ est de l'ordre de 10^{-87}. Ce nombre est si petit que si l'on prend les premiers chiffres significatifs, on obtient 0. Non seulement cela, mais ce petit nombre n'est pas propice aux calculs ultérieurs et pourrait facilement déborder les limitations des types. Nous avons donc décidé d'utiliser $logK$ comme fonction noyau. Il est vrai que cette fonction ne satisfait pas les conditions de Mercer car elle n'est pas définie positive. Mais comme nous la calculons directement la matrice du noyau, l'étape suivante consiste simplement à trouver le minimum de la fonction de perte. Alors que $\log K$ et K ont les mêmes propriétés croissantes et décroissantes, nous supposons que cela n'a aucun effet sur la recherche du minimum de la fonction de perte. D'après nos essais, l'algorithme c-svm de libsvm a pu calculer un modèle avec des résultats relativement bons, ce qui corrobore notre conjecture.

Cependant, cela présente l'inconvénient de prendre plus de temps pour calculer la matrice du noyau une fois et le fichier de sortie prend plus de place. Même pour notre jeu de données de 5307 fragments d'acides aminés, le calcul prend plus de 10 minutes et le fichier de sortie est de taille de 500 MB. De plus, le temps de calcul et l'espace occupé augmentent quadratiquement avec le nombre de données, et il se peut que nous ne puissions pas utiliser cette méthode pour des jeux de données plus importants. Pour l'instant, cette méthode est adaptée à l'ensemble de données que nous utilisons.

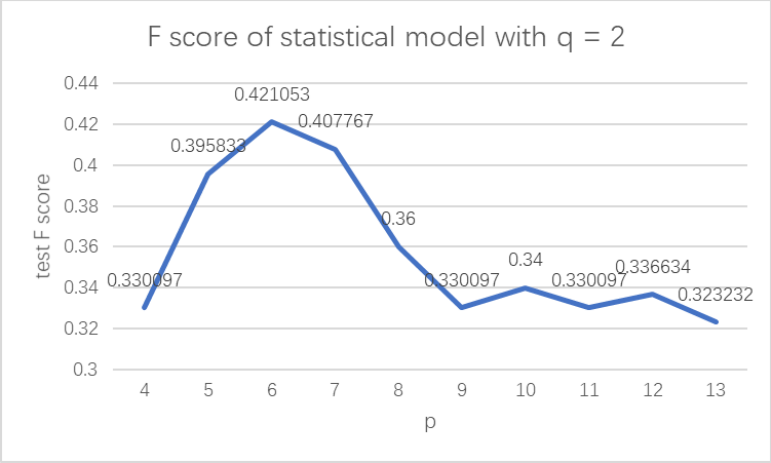
4 Résultats

Nous avons d'abord dû décider du critère pour déterminer si notre modèle était bon ou non. Nous notons que nos fragments d'acides aminés négatifs sont beaucoup plus grands que nos fragments positifs, ce qui rend inutile le calcul de l'exactitude. Nous sommes plus concernés par la précision et le rappel, qui montrent le pourcentage de vrais positifs. Donc nous utilisons le fscore, qui mesure la précision et le rappel, comme notre critère de jugement.

4.1 Sélection du p et du q

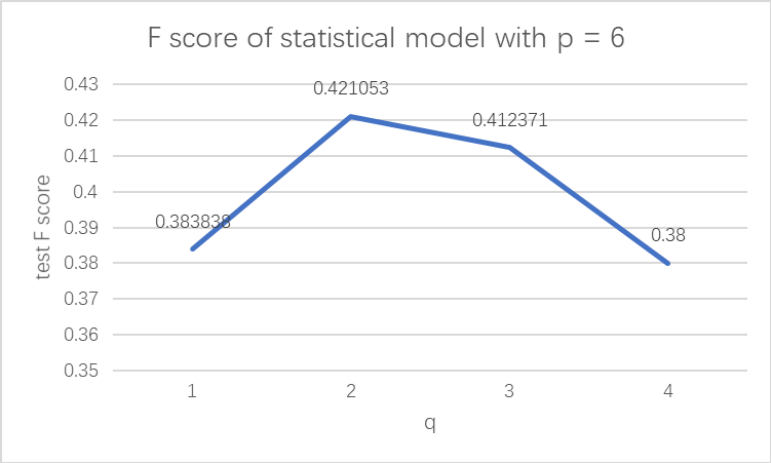
p et q définissent ensemble la taille du fragment d'acide aminé et la position du site de clivage. Chaque fois que p et q sont modifiés, le fichier de format svm stockant l'ensemble de données doit être régénéré, ce qui rend la sélection de p et q à l'aide du modèle dans libsvm complexe et longue. Donc nous avons utilisé le modèle statistique pour trouver p et q. Il est vrai que le p et q optimal pour le modèle statistique ne signifie pas qu'il est le même pour les autres modèles, mais après notre comparaison avec $p = 8, q = 2$ et $p = 6, q = 2$, tous les modèles que nous avons utilisés ont obtenu de meilleures performances avec $p = 6, q = 2$.

Tout d'abord, nous fixons $q = 2$ et changeons la valeur de p.

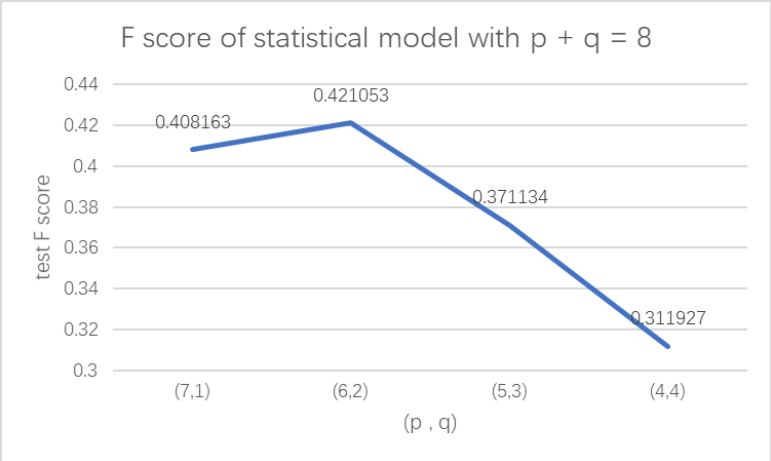


Nous trouvons un optimum local pour $p = 6$, mais nous constatons également que la fonction n'est pas concave et que l'optimum local n'est pas nécessairement l'optimum global.

Nous fixons ensuite $p = 6$ et changeons la valeur de q.



Après cela, nous fixons $p + q = 8$ et changeons les valeurs de p et q.

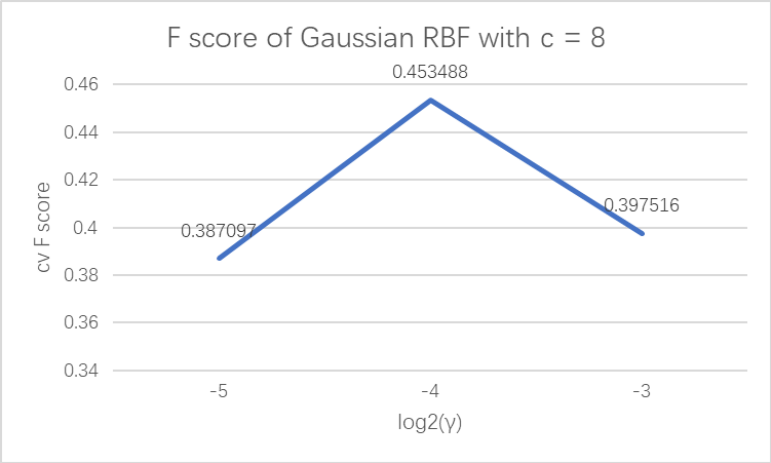


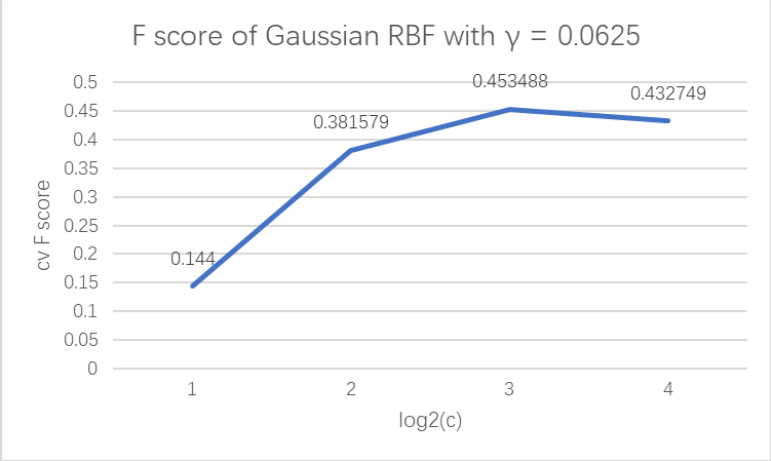
Dans tous ces cas, nous trouvons que $(p = 6, q = 2)$ est un optimum local, mais en même temps, rien n'assure que ces fonctions sont concaves. Bien que nous ne puissions pas être sûrs que $(p = 6, q = 2)$ soit un optimum global, il est au moins optimal pour toutes les combinaisons que nous avons testées.

4.2 Sélections des paramètres C et γ optimales pour le noyau RBF de Gaussien

Le paramètre C indique à le SVM dans quelle mesure vous voulez éviter de mal classer dans l'ensemble d'entraînement. Pour de grandes valeurs de C, l'optimisation choisira un hyperplan à marge plus petite si cet hyperplan permet de mieux classer correctement toutes les données de l'ensemble d'entraînement. Le γ est un paramètre dans la fonction du noyau : $K(x, y) = \exp(-\gamma|x - y|^2)$

En pratique, notre programme teste chaque paire (c, γ) dans la liste de c et la liste de γ. Les graphiques suivants montrent le changement de Fscore pour changer γ après avoir fixé c et le changement de Fscore pour c après avoir fixé γ.



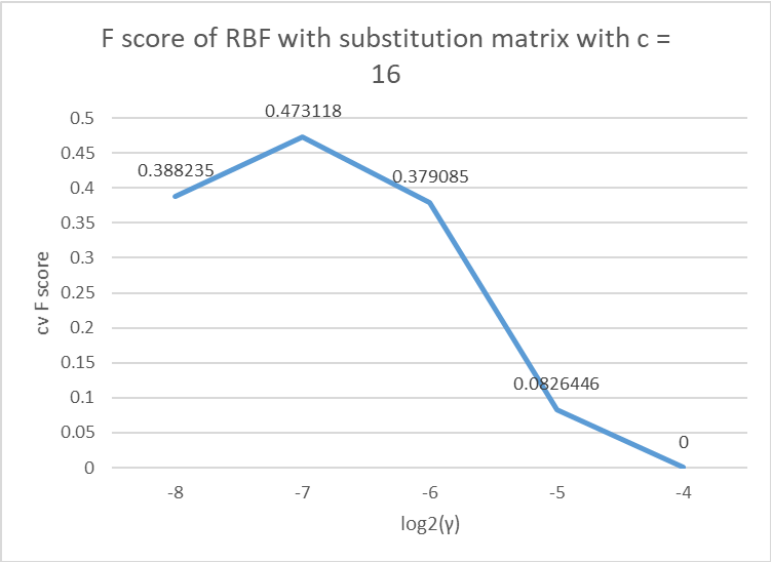


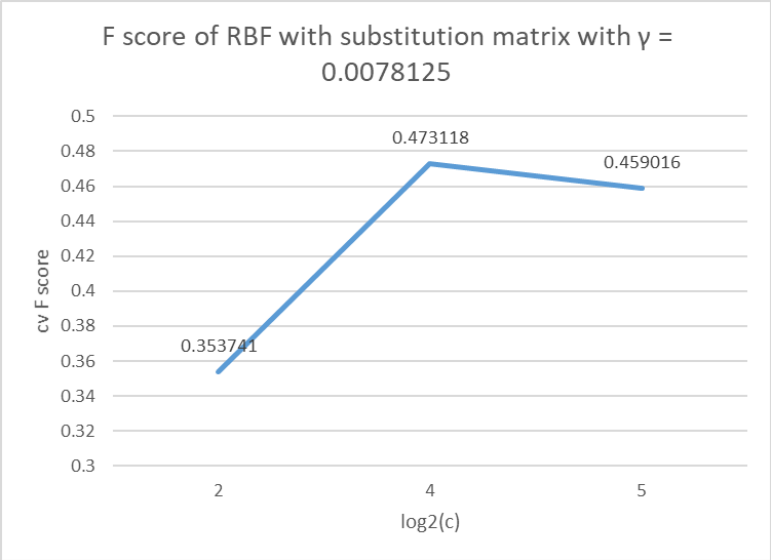
Nous remarquons que le Fscore ici est différent de celui dans la sélection de p et q. Le Fscore ci-dessus est celui de l'ensemble de données de test alors qu'ici il s'agit du Fscore après validation croisée de l'ensemble de données d'entraînement. Les Fscore suivants sont également obtenu de cette manière.

Nous trouvons les paramètres optimaux $c = 8$ et $\gamma = 0.0625$. En utilisant ce modèle sur l'ensemble de test, nous obtenons les résultats suivants :

```
Precision = 100% (26/26)
Recall = 92.8571% (26/28)
F-score = 0.962963
BAC = 0.964286
Accuracy = 99.8711% (1549/1551)
AP = 0.998768
```

4.3 Sélections des paramètres C et γ optimales pour le noyeau Pseudo-RBF



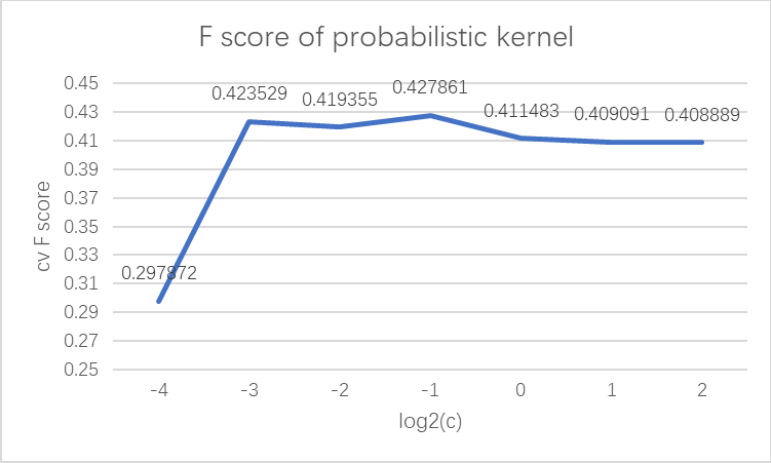


Nous trouvons les paramètres optimaux $c = 16$ et $\gamma = 0.0078125$. En utilisant ce modèle sur l'ensemble de test, nous obtenons les résultats suivants :

```
Precision = 100% (26/26)
Recall = 92.8571% (26/28)
F-score = 0.962963
BAC = 0.964286
Accuracy = 99.8711% (1549/1551)
AP = 0.998768
```

4.4 Sélection du paramètre C pour le noyau probabiliste

En changeant la valeur de C, on obtient le résultat suivant :

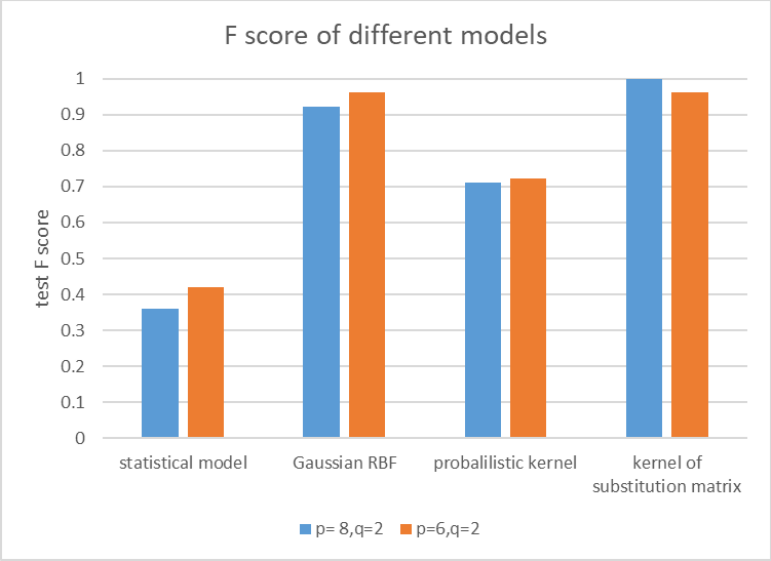


Nous trouvons le paramètre optimal $c = 8$. En utilisant ce modèle sur l'ensemble de données de test, nous obtenons les résultats suivants :

```
Precision = 89.4737% (17/19)
Recall = 60.7143% (17/28)
F-score = 0.723404
BAC = 0.802915
Accuracy = 99.1618% (1538/1551)
AP = 0.774927
```

4.5 Comparaison des modèles

Nous avons comparé les résultats des 4 différents modèles mentionnés avant.



Comme mentionné précédemment, pour tous les modèles, le choix de $p = 6, q = 2$ est meilleur que $p = 8, q = 2$. De plus, les modèles de SVM sont bien meilleurs que le modèle statistique simple.

Le SVM du noyau probabiliste a un temps d'exécution beaucoup plus important et il consomme beaucoup plus de mémoire que celui du noyau le noyau RBF de Gaussien, mais il ne fonctionne pas aussi bien qu'on pourrait s'y attendre. Cela peut s'expliquer par le fait que, comme nous l'avons mentionné précédemment, nous avons utilisé $logK$ au lieu de K. Nous avons également lu l'article proposant cette méthode et avons constaté que la définition de la fréquence de fond était différente, la fréquence étant calculées après que la séquence a été découpée en fragments, Et l'auteur n'a pas

soustrait la fréquence de fond comme nous l'avons fait lors du calcul du noyau. Cela peut également avoir contribué aux résultats pires que prévus.

5 Conclusion

Dans ce projet, nous avons bien vu la puissance de l'algorithme SVM. Il s'adapte bien à ce problème de classification par rapport aux résultats obtenus par la matrice PSSM. A l'avenir, nous pouvons entraîner le modèle avec plus de donnée et en choisissant certain noyeau plus pertinent pour améliorer son F-score.