Université Paris-Saclay

Utilisation des graphes pour le broker Fink

January 24, 2024

Rapport de stage

Pierre Cavalier

Supervisé par:
Julien Peloton, IJCLab

Remerciements

Je tiens à exprimer ma gratitude envers toutes les personnes qui m'ont soutenu et accompagné tout au long de mon stage au sein de l'IJCLab.

Je tiens tout d'abord à remercier mon tuteur de stage, Julien Peloton, pour m'avoir offert cette opportunité enrichissante. Merci à lui d'avoir su se rendre disponible et ce, de manière très régulière, de m'avoir aidé sur les nombreux problèmes que j'ai pu, de ce fait, surmonter. Ses remarques ont toujours été pertinentes et m'ont permis d'avancer, de progresser aussi bien dans mon travail que personnellement, merci encore à lui.

Je n'oublie pas de remercier Roman Le Montagner ainsi que Danah Haker d'avoir partagé le même bureau que moi et d'avoir échangé sur de nombreux sujets aussi variés soient-ils. Vous avez contribué à rendre de stage d'autant meilleur et je vous en suis reconnaissant.

Ce stage a été une expérience formatrice et épanouissante, et je suis convaincu qu'elle aura un impact significatif sur mon développement professionnel futur. Merci à toutes les personnes qui ont contribué à rendre cette expérience possible.

Contents

1 Introduction

Dans le cadre de mon stage de M1 j'ai eu l'occasion d'effectuer un stage de 4 mois à IJCLab, un laboratoire de la vallée de l'Université Paris-Saclay, en travaillant sur le projet Fink dans le département informatique du laboratoire.

Fink est un broker [4], c'est-à-dire un système informatique qui facilite les échanges de données entre les télescopes et les utilisateurs qui étudient les phénomènes transitoires en astrophysique. Même si chaque télescope à une manière propre de fonctionner, la démarche de chacun pour communiquer l'information reste la même dans les grandes lignes. En effet, à chaque fois qu'un objet apparaît ou disparaît dans le ciel, une alerte est émise et récupérable par la communauté scientifique. Le broker Fink récupère ces données, les agrège et les enregistre pour une utilisation ultérieure. De surcroît, de nombreux modules ont été mis place par Fink dans le but d'ajouter de la valeur en proposant par exemple des modèles utilisation l'intelligence artificielle pour classifier les alertes.

L'objectif de ce stage est de proposer une nouvelle méthode de classification en ne se basant plus sur les attributs d'une alerte elle-même mais sur les relations entre les différentes alertes.

En analysant les relations entre les alertes, on peut obtenir une compréhension plus approfondie de leur fonctionnement et de leur interaction. Cela permet d'identifier les schémas, les dépendances et les causalités qui peuvent être masqués lorsque l'on considère chaque entité de manière isolée. En étudiant les relations entre les entités, on peut alors développer des modèles et des systèmes prédictifs plus performants. Les relations peuvent fournir des indices précieux pour anticiper les comportements futurs, les tendances émergentes et les résultats potentiels. En examinant les relations entre les entités, on peut repérer plus facilement les anomalies ou les schémas inhabituels. Les écarts par rapport aux relations normales peuvent servir d'indicateurs de problèmes potentiels, de comportements anormaux. Les relations entre les entités sont essentielles pour la modélisation et la simulation de systèmes complexes. En intégrant les relations dans les modèles, on peut obtenir des simulations plus précises et réalistes, ce qui permet de tester différentes hypothèses, d'anticiper les conséquences et de prendre des décisions éclairées pour programmer rapidement de nouvelles observations d'un objet dans une phase transitoire intéressante.

Durant ce stage j'ai pu mettre en œuvre différentes méthodes vues lors de mon cursus notamment sur l'analyse de données et l'apprentissage supervisé. A partir de différentes stratégies, nous avons pu créer différents graphes à partir des alertes qui ont donnés des résultats plus ou moins concluant. Le travail effectué durant ce stage était principalement exploratoire et consistait à chercher la meilleure manière d'exploiter les objets nouveaux dans ce domaine que sont les graphes. Une grande partie du stage a donc consisté à étudier de nouvelles méthodes, parfois fructuante ou non, et à les mettre en place dans le cadre du broker Fink.

2 Jeu de données

2.1 Les alertes ZTF

Le jeu de données est composé d'alertes de télescope principalement du télescope ZTF. Une alerte est émise quand la variation lumineuse d'un pixel de plus ou moins cinq fois le bruit ambiant de l'image est détectée. Une alerte est la représentation partielle d'un phénomène astrophysique à un instant t, mais il n'est pas impossible que ce phénomène évolue au cours du temps. Il peut donc peut émettre plusieurs alertes qui partageront une partie de leur information. Dès lors que la figure de l'observation (la première sur la figure 1) diffère de la figure de référence (la deuxième sur la figure 1), qui est une photo prise à un instant donné dans le temps (généralement au lancement du télescope) une alerte est émise par le télescope. La troisième figure sur la figure 1 correspond à la soustraction de la photo de l'observation avec la figure de référence. Les photos dans le jeu de données sont tronquées et centrées sur l'alerte en question.

Ainsi une alerte est un ensemble d'information capturée par le télescope à un instant t, dès lors que le point au centre de la troisième figure de la figure 1, a une luminosité supérieure à cinq fois le bruit de l'image. Une alerte est donc associée à un point dans l'espace et contient l'information passée de ce point.

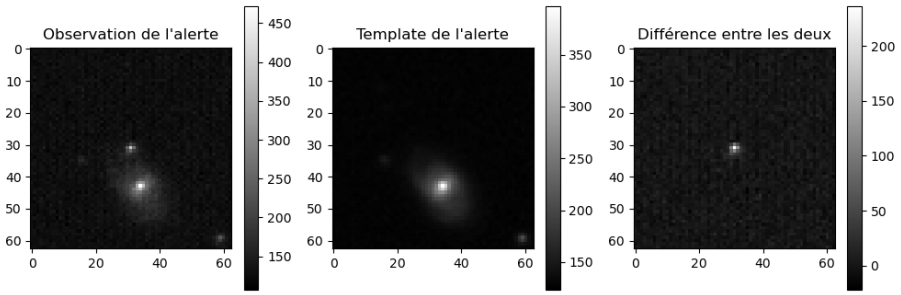


Figure 1: Une émission d'alerte

Les alertes envoyées par ZTF contiennent un ensemble de caractéristiques qui se décompose en 3 composantes :

- ztf.alert: contient les détails sur l'alerte en tant que telle tels que les photos, les détails techniques, comme des détails sur la calibration du télescope, l'atmosphère ambiante, des numéros d'identification pour l'alerte, l'objet qui la émise etc..
- ztf.alert.candidate: contient les détails techniques de l'alerte tels que la date, la fréquence de détection, l'intensité de la luminosité etc..
- ztf.alert.prv_candidate: contient les détails techniques des alertes ayant eu lieu au même endroit durant les 30 derniers jours.

Les différent détails techniques sont très variés et peuvent aller de l'intensité lumineuse de l'alerte, qui s'avère être une donnée importante, au temps d'exposition dont l'utilité est plus ou moins relative selon notre utilisation.

Au total, cela nous donnes deux cents caractéristiques uniques, que l'on appellera *features* par la suite. Une alerte est donc un ensemble de features portant sur l'observation ainsi que les observations passées, qui ont eu lieu au même endroit. Si un compte chaque pixel de chaque photo (64 par 64), cela nous amène à un peu plus d'une dizaine de millier de features. Celà nous conduira inévitable à parler de réduction de

dimensionalité pour ne garder que les features pertinentes.

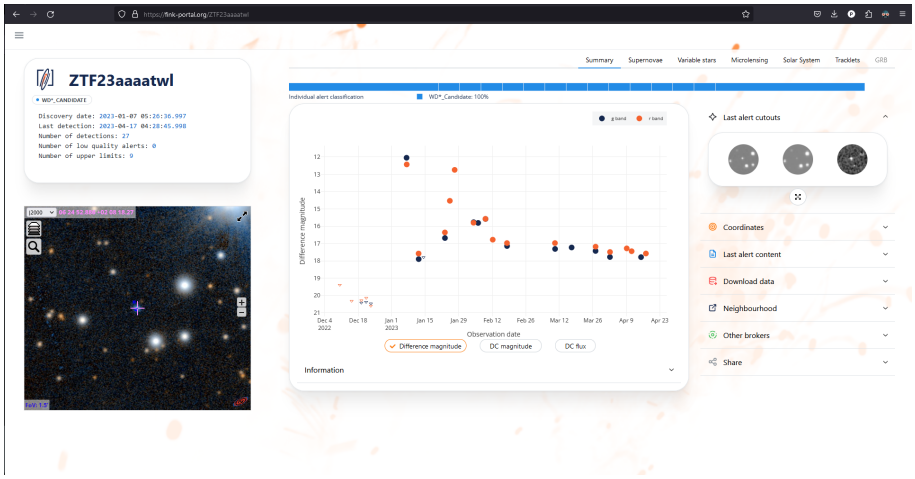


Figure 2: La page d'un objet donné sur le site Fink

A partir de ces données, on peut obtenir une courbe de lumière, c'est-à-dire l'évolution de la luminosité en fonction du temps (voir figure 2), pour chaque objet, ce qui permet d'observer les tendances. Le problème étant que selon l'ancienneté de l'objet, sa courbe de lumière contiendra plus ou moins de points.

Pour pallier ce problème, nous allons aborder le travail fait par Fink sur les alertes, c'est à dire une vue partielle du phénomène transitoire, dans le but de mettre une nom sur ce phénomène pour le classer. On notera aussi que chaque point de la courbe de lumière a fait l'objet d'une alerte, ainsi une courbe de lumière est composée de plusieurs points issus à chaque fois d'alertes différentes.

C'est donc la raison pour laquelle nous allons classifier les alertes en tant que telles qui, une fois émises sont fixes dans le temps, contrairement aux objets dont la classification peut évoluer dans le temps.

2.2 Valeur ajoutées par Fink

Dans le but de standardiser les données, le broker Fink calcule des valeurs statistiques données dans la section A.1 et A.2 de l'article [1] et qui sont au nombre de 42. De plus, ces calculs sont effectués deux fois, sur la courbe de lumière bleue et rouge, qui correspondent aux deux bandes de fréquences observées par ZTF (500 et 800 nm respectivement). Ces valeurs permettent de décrire la courbe de lumière sans trop de perte d'information et permet d'avoir la même structure de données pour chaque alerte, indépendamment du nombre d'observations passées pour chacune d'entre elles.

De plus, à chaque alerte, on rajoute les différentes identifications d'alertes dans divers catalogues d'objets célestes ainsi que les différents scores donnés par les classifieurs Fink quant au type de l'alerte observée. Les différents modules sont disponibles ici et permettent d'extraire une information scientifique à partir des données factuelles de l'alerte envoyée par le télescope ZTF.

Pour résumer brièvement, les données exploitées au cours de ce stage sont les alertes de ZTF, comprenant les caractéristiques et détails techniques de l'image de l'alerte ainsi que les valeurs ajoutées par Fink. Chaque nuit comportant une centaine de milliers d'alertes, le volume total d'alertes s'élève à 130 millions, Fink fonctionnant depuis plus de 4 ans.

Notons que parmi ces alertes, environ 20% sont des points uniques. Ainsi leur courbe de lumière est inexploitable car la grande majorité des données statistiques calculées à partir de cette courbe nécessite à minima deux points. Cela peut-être dû à du bruit ou à un objet relativement proche et se déplaçant dans l'espace laissant de multiples alertes à différents endroits comme des astéroïdes de notre système solaire.

2.3 Classes des alertes

En se basant sur les catalogues d'astres déjà connus et sur les classifieurs de Fink, chaque alerte aura un attribut `finkclass` qui correspondra au type de phénomène transitoire qu'est l'alerte. Si aucune classe n'est proéminente alors son attribut `finkclass` sera `Unknown`.

Le nombre de classes total est de 370. Lors de ce stage nous avons pris un échantillon de celles-ci que nous avons regroupées en *métaclasses* plus grandes. Ainsi l'échantillon d'étude contient respectivement, des éléments de deux métaclasses et des éléments de quatre classes individuelles:

- Les étoiles variables (Variable Star) sont des astres qui connaissent des fluctuations de luminosité au fil du temps. Leurs variations peuvent être périodiques et assez lisses. Ces changements sont causés par divers facteurs tels que les pulsations internes, les éruptions stellaires ou les éclipses binaires. Ces dernières sont souvent connues et cataloguées.
- Les noyaux actifs de galaxies (AGN - Active Galactic Nuclei) sont des régions très lumineuses situées au centre des galaxies. Ils abritent des trous noirs supermassifs qui accrètent de la matière à un taux extrêmement élevé.
- Les étoiles cataclysmiques (CataclyV*) sont des systèmes binaires d'étoiles ; principalement une étoile classique qui transfert sa masse à une naine blanche, avec des sursauts périodiques dans la courbe de lumière.
- Les supernovas (SN) sont des événements cosmiques qui marquent la fin explosive de la vie d'une étoile massive. Lorsqu'une étoile épuise son combustible nucléaire, elle s'effondre sous sa propre gravité. Ce processus déclenche une réaction en chaîne qui produit une explosion d'une grande intensité en contraste avec les alertes précédentes dont la variation de luminosité est plus légère
- Les kilonovas (KN) sont des événements astronomiques violents et éphémères qui se produisent lors de la fusion de deux étoiles à neutrons. Lorsque deux de ces étoiles extrêmement denses entrent en collision, elles libèrent une quantité incroyable d'énergie, créant une explosion extrêmement lumineuse à l'instar des supernovas.
- Les alertes ambiguës (Ambiguous) sont des classes qui surviennent lorsque deux classifieurs de Fink, ou plus, attribuent à une alerte des classes différentes, la plupart des alertes possédant une telle classe sont des supernovas que Fink confond avec des kilonovas ou des évènements de microlentilles gravitationnelles.

Notons que bien que nous travaillons avec des métaclasses, les éléments à l'intérieur de celles-ci peuvent avoir des disparités entre eux bien que moindres en comparaison avec les éléments d'autres classes. Pour la suite, nous travaillerons avec 200 éléments de chacune de ces classes, soit un sous ensemble de 1200 alertes. La répartition des classes est homogène ici mais ce n'est pas le cas dans le volume entier de données.

3 Réduction de dimensionalité

3.1 Étude qualitative

Au vu du grand nombre de variables possédées par chaque alerte et du fait que ces dernières n'étaient pas toutes numériques, nous avons, dans un premier temps, cherché à éliminer celles qui n'apportaient, à priori, aucune information permettant de conclure quant à la proximité de deux alertes.

Dans un premier temps, nous allons nous concentrer sur l'alerte initiale de ZTF, avant qu'elle ne soit retouchée par Fink. Parmi les variables envoyées, la plupart concerne la qualité de l'image et donne des informations techniques. Elles sont déjà utilisées par Fink pour décider du fait de la véracité ou non d'une alerte et ne permettent pas de déterminer la nature d'une alerte. Par exemple, les photos des alertes (voir la figure 1) ne donnent pas réellement d'information ce qui est dû au fait qu'avoir l'intensité lumineuse dans une seule bande de fréquence ne permet pas de conclure sur quoi que ce soit. De même pour les trois photo de l'alerte (l'observation, le template, et la différence des deux), seule la luminosité au centre nous intéresse car les objets observés ne s'étalent pas sur toute l'image.

De même, il n'y a pas de raison que certaines périodes de temps soient plus propices ou non à un certain type d'évènement en particulier, et nous n'avons donc pas conservé la date d'émission. Finalement, les observations étant effectuées dans la voie lactée, la probabilité de trouver des étoiles dans le plan de cette dernière est supérieure à celle d'en trouver hors de ce plan et vice versa pour les objets galactiques. Pour éviter d'introduire ce biais, nous ne considérerons pas la position de l'astre. Le même genre de raisonnement peut s'appliquer sur un grand nombre de features et il n'est pas spécialement intéressant de détailler ce propos ici.

Finalement, nous avons gardé les features suivantes:

- jdstarthist : la date d'émission de la première alerte à ce point
- magpsf : la magnitude du point
- sigmapsf : l'incertitude sur cette magnitude
- fid : le filtre (rouge ou vert) utilisé
- magnr : la magnitude du point le plus proche si il y en a un assez proche dans le catalogue PSF
- sigmagnr : l'incertitude sur cette magnitude
- isdiffpos : Indicateur de si l'objet est gagné ou perdu en magnitude (par rapport au template)
- neargaia : Distance de la source la plus proche du catalogue Gaia DR1
- sgscore1 : Probabilité que l'objet soit une étoile/galaxie du catalogue PS1
- classtar : Classification des étoiles et des galaxies par SExtractor [2]

De plus, nous avons gardé les valeurs statistiques citées dans la partie 2.2 sur les deux courbes de lumières disponibles, le rouge et le vert. A cela s'ajoute les différents scores des classifieurs Fink de supernovas ainsi que de kilonovas ce qui nous donne un total de 66 features.

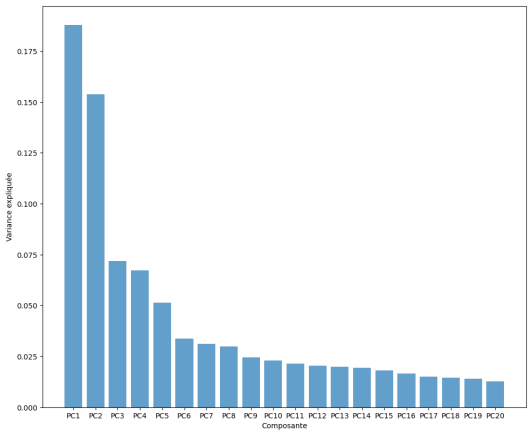
3.2 Analyse en composante principale

En ayant réduit le nombres de variables explicatives, on obtient un jeu de variables quasiment entièrement numériques, la variable qualitative `isdiffpos` ne contient que deux modalités donc se convertit en variable numérique binaire.
Pour réduire le nombre de variables qui reste grand (100 variables) et éviter d'avoir trop de redondance dans les informations, on se propose de faire une analyse en composante principale (PCA). On transforme chaque variable explicative X de la manière suivante, les unité des variables n'étant pas toutes les mêmes.

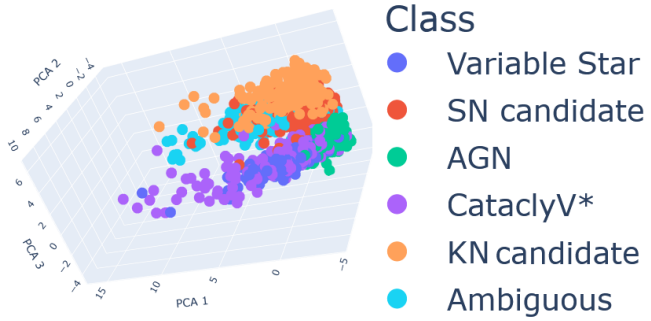
$$X^{(\mathrm{PCA})} = \frac{X - \bar{X}}{\sigma(X)} \tag{1}$$

Si certaines alertes présentent des valeurs manquantes, on les définit à 0 faute de mieux. L'impact de cette décision n'a pas été étudié lors du stage. Cependant, le nombre de valeurs manquante correspond à 20% de l'ensemble de valeurs ce qui est assez conséquent. 80% des valeurs manquantes se concentrent sur 20% des features ce qui revient à dire que les alertes considérées ne contiennent pas uniquement des valeurs manquantes.

En utilisant le module PCA de scikit learn, on obtient les différentes composantes principales de notre jeu de données ainsi que leur variable expliquée. On remarque sur la figure 3a que la vingtième composante n'explique qu'un seul pourcent de la variance et que les vingts premières composantes expliquent à elles seules 90% de la variance du jeu de données.



(a) Variance expliquée par composante principale



(b) Alertes en fonction des trois premières composantes

Figure 3

Sur la figure 3b, on a représenté un sous ensemble de 200 points de notre jeu de données dans l'espace des 3 première composantes principales et certaines classes se séparent les unes des autres mais certaines restent très mêlées entre elles. On remarque cependant que les trois premières composantes expliquent seulement 41 % ce qui peut potentiellement expliquer des frontières assez floues entre chaque région.

4 Méthode de construction de graphe

Definition 1. *Un graphe G est défini comme un couple ordonné de nœud (Vertex) contenant un ensemble d'informations et d'arêtes (Edge) contenant elles aussi des caractéristiques. Finalement, on peut retranscrire un graphe sous cette forme :* $G = (V, E)$

Dans notre cas, les nœuds de notre graphes sont les différentes alertes et les caractéristiques de chaque nœud sont les composantes principales obtenues en transformant les features des alertes comme expliqué dans la partie 3.2.

La problématique fondamentale du stage a été de trouver les conditions pour relier deux alertes ou non et quelles informations y mettre. Pour chaque paire de nœud, on associera un score entre 0 et 1 qui permettra de déterminer la pertinence de la connexion entre deux nœuds. Si ce score est supérieur à 0.5, nous lierons ces nœuds par une arête dont la caractéristique sera ce score. Nous définirons plus tard les conditions d'attribution de ce score.

L'importance est de, au delà de l'aspect visuel du graphe, réussir à définir une manière d'évaluer les performances du graphe à l'aide de métriques. Nous avons donc défini deux métriques pour évaluer et comparer les graphes. Pour cela, nous allons dans un premier temps définir les bonnes arêtes comme étant les arêtes entre deux alertes de même classe. De même, les mauvaises arêtes seront définies comme les arêtes reliant des alertes de classes différentes. Ces définition restent subjectives, dans le sens où une bonne alerte ne sera pas forcément une arête voulue par un utilisateur, de même pour les mauvaises arêtes.

Definition 2. *On définit la* ***précision*** *d'un graphe G comme le rapport du nombre de bonnes arêtes avec le nombre d'arêtes total.*

Definition 3. *On définit la* ***densité de similarité*** *d'un graphe G comme le rapport du nombre de bonnes arêtes avec le nombre possible de bonnes arêtes dans le graphe.*

Remarque 1. *Ce nombre vaut* $\sum_{Classes} \frac{n_{classe}(n_{classe}-1)}{2}$ *avec* n_{classe} *le nombre d'élément dans la classe en question.*

La métrique qui est la plus importante est la précision du graphe. En effet, dans notre cas, nous avons décidé de privilégier la qualité des arêtes par rapport à la quantité. Un graphe présentant toutes les bonnes arêtes mais qui possède 50% de mauvaises sera jugé moins performant qu'un graphe possédant la moitié de bonnes arêtes possibles mais seulement 10 % de mauvaise arêtes.

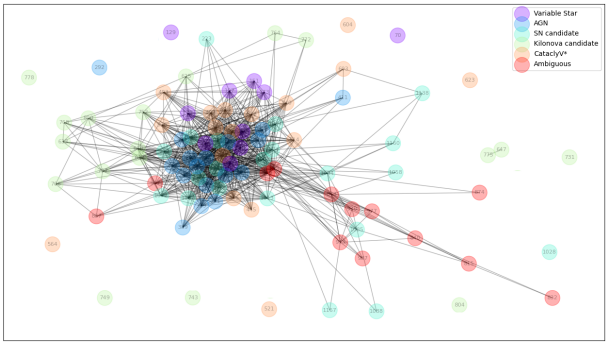
La seconde métrique a plus un but indicatif finalement et servira à comparer des graphes ayant plus ou moins la même précision. Elle permet aussi d'écarter des graphes possédant une seule arête qui serait une bonne arête dans la totalité des arêtes. Un tel graphe posséderait une précision de 100% mais ne serait que très peu utile en pratique et n'apporterait aucune valeur.

Le but n'est pas d'obtenir un graphe avec une précision de 100% et une densité de similarité de 100 % car les alertes au sein d'une même classe peuvent présenter des disparités et ne pas être aussi similaires qu'escompté. De même, le fait que des alertes de classes différentes peuvent présenter des similitudes et des liens entre elles pourrait ouvrir de nouvelles questions.

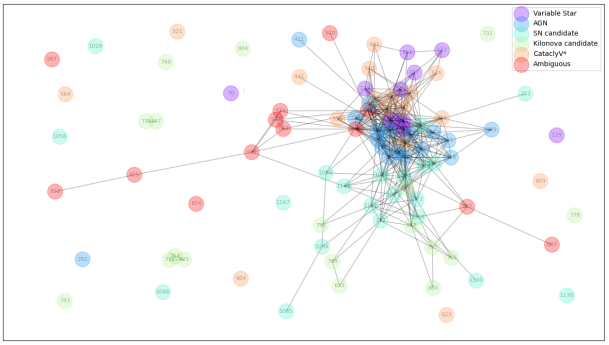
4.1 Distance euclidienne

La première idée que nous avons eue a été de relier les alertes entre elles (avec un score de 1) en fonction de leur distance dans $\mathbb{R}^{20}$, l'espace des composantes principales. En effet, le but est de relier deux alertes

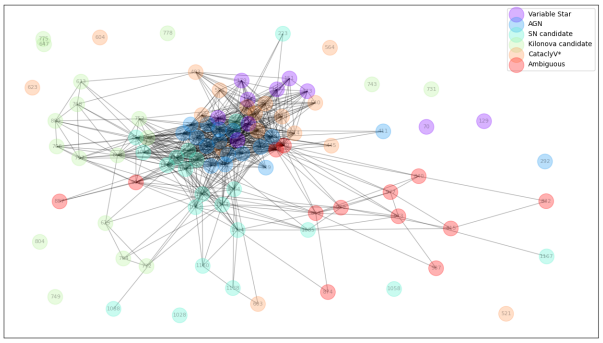
si et seulement si leur distance euclienne est inférieure à un rayon r. Autrement dit, chaque alerte sera reliée avec toutes les alertes contenues dans la boule centrée en cette alerte et de rayon r. La question que cette définition ouvre reste la définition de cette distance limite. Il a été cité plus haut que notre jeu de données comprend 1200 élements répartis de manière homogène parmi 6 classes, cependant pour une représentation visuelle plus interprétable, les graphes sur la figure 4 sont tracés avec seulement 100 points tirés aléatoirement dans ce jeu de données.



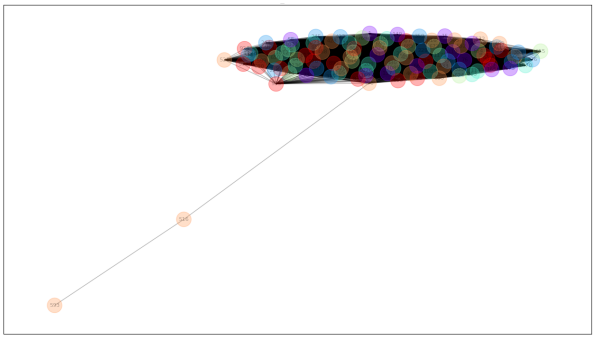
(a) $n\sqrt{n}$



(b) $n \ln n$



(c) Nombre théorique d'arête (NTA)



(d) Une seule composante connexe

Figure 4: Différents graphes construits sur la base de la distance euclidienne (SCC)

La première remarque sur ces graphes est que la distance, celle que l'on voit entre deux alertes sur le graphe, n'a pas de signification particulière. En effet, le module networkx arrange les alertes selon un algorithme de Fruchterman-Reingold force-directed qui optimise simplement l'espace graphique disponible. Pour les figures 4a et 4b, nous avons défini un nombre d'arêtes à atteindre à partir de la taille de l'échantillon utilisé, ici $n = 100$.
Pour chacun des graphes, on remarque que plus d'une vingtaine de nœuds sont complètement isolés du reste du graphe. La composante connexe au milieu semble présenter quelques agrégats, sur les extrémités notamment, de supernovas et kilonovas tandis qu'il semble y avoir une masse au centre assez peu interprétable et où beaucoup d'alertes semblent connectées.

La figure 4c correspond à une figure contenant le nombre théorique d'arêtes qu'il y aura sur un graphe où toutes les arêtes sont des bonnes arêtes et où toutes les bonnes arêtes sont présentes. Ce nombre est donnée dans la remarque 1 et vaut ici 791. La construction en elle-même n'est pas extrapolable à un jeu de données dont on ne connaît pas les classes des éléments, néanmoins, il nous a semblé intéressant de voir les résultats sur un jeu de données dont les classes étaient connues.

Sur la dernière figure 4d, nous avons essayé de trouver le r minimal de façon à ce que le graphe obtenu soit connexe, i.e chaque point est accessible depuis un autre par une chaîne d'arêtes. Cette méthode est très sensible aux valeurs aberrantes et crée une énorme masse en haut du graphe, relativement peu interprétable.

Type de construction	$n\sqrt{n}$	$n \log n$	NTA	SCC
Précision	34%	40%	36%	16%
Densité de similarité	38%	23%	34%	79%

On calcule les métriques données dans la section 4 et on se rend compte que la précision reste relativement faible. Sur les quatre types de constructions, on obtient un maximum de 40% pour une densité de similarité minimale de 23%. Comme on pouvait s'y attendre, la densité de similarité augmente au fur et à mesure que l'on croit r, là où la précision en pâtit grandement. Pour atteindre une précision convenable (plus de 50%), il faudrait baisser encore plus r ce qui nous mènerait à une densité de similarité très basse, cela nous a motivé à changer d'approche.

4.2 Réseau de neurones

Pour relier deux nœuds, nous avons essayé de passer par un réseau de neurones en nous basant sur l'approche de l'article [5], un article de physique des particules, qui a mis en place des réseaux de neurones pour relier deux détections de particules entre elles. Notre objectif est, pour une paire d'alertes données, d'étudier leur similarité et d'évaluer à quel point créer un lien pour cette paire à un sens. Pour mettre en place un tel réseau, nous avons utilisé la bibliothèque logicielle Pytorch.

Pour cela nous allons utiliser le réseau de neurones représenté sur la figure 5. En couche d'entrée on utilise les 20 composantes des 2 alertes soit un total de 40 neurones en couches d'entrée.

Les couches cachées sont des *Fully Connected Layer*, c'est-à-dire que les neurones de la couche $n-1$ et n sont tous reliés entre eux.

Pour les fonctions d'activation de ces couches nous avons utilisé la fonction ReLU qui, pour une entrée x, renvoie $\max(0, x)$.

Enfin nous avons connecté la dernière couche cachée à un neurone de sortie, avec la fonction d'activation sigmoid pour obtenir un nombre entre 0 et 1 qui aura comme signification la similarité entre ces deux alertes. 0 signifiant que les alertes ne présentent pas de point commun à l'inverse de 1.

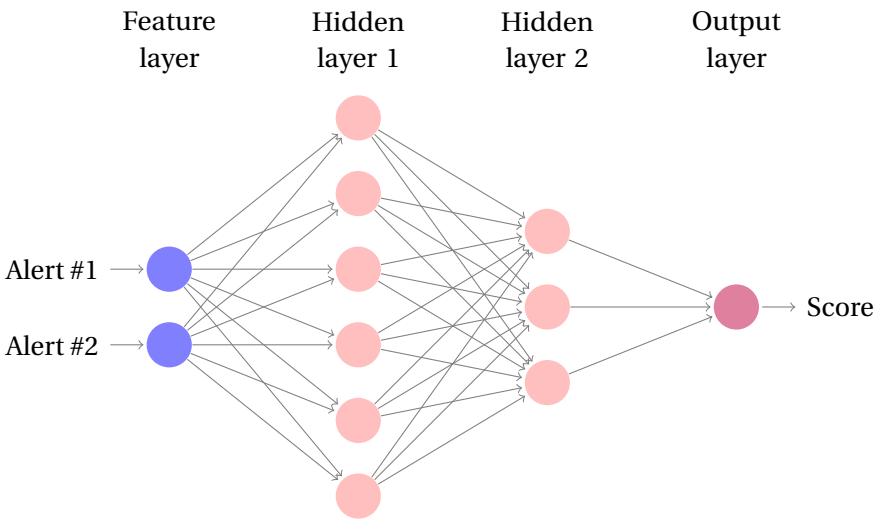


Figure 5: Réseau de neurones utilisé

Pour entraîner ce modèle nous avons, dans un premier temps, enlevé les alertes de la classe `Ambiguous` de notre jeu de données. En effet, la classification de ceux-ci étant floue par définition, nous ne voulions ni entraîner ni évaluer le modèle à partir de cette classe. Pour entraîner le modèle, nous avons pris un jeu d'entraînement de 200 alertes et un jeu de test de la même taille.

Rappelons que nous entraînons des paires de ce jeu de données soit $\frac{200\times(200-1)}{2} = 19900$ paires pour l'entraînement.

Pour ce qui est de l'évaluation, à chaque paire on associe une étiquette, qui vaut 1 si les alertes sont de la même classe et 0 sinon.

Nous avons utilisé la Binary Cross Entropy comme fonction de perte qui pour un label y et une prévision $\tilde{y}$ renvoie $-y(\log\tilde{y} - (1-y)(1-\log 1-\tilde{y}))$.

En entraînant le modèle sur 200 epochs avec un batchsize de 16, c'est-à-dire que l'on entraîne le modèle sur tout les paires et ce 16 par 16, et ceux 200 fois. En utilisant Adam comme optimiseur et un taux d'apprentissage de 0.001 on arrive à une précision de l'ordre de 99.9 % sur le jeu de données de test. En traçant seulement les arêtes dont le score est supérieur 0.5 on obtient la figure 6

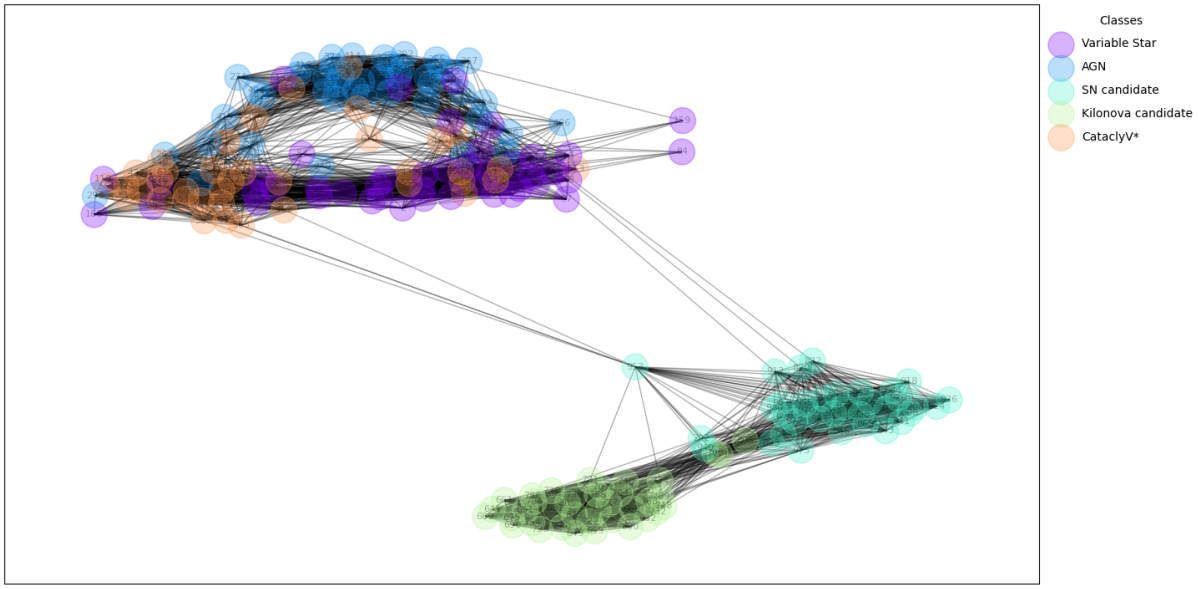


Figure 6: Neural network graph

La première chose que l'on remarque sur cette figure, est que, contrairement à celles produites dans la partie 4.1, le graphe est connexe et qu'il n'y pas cette forme d'amas central.
On remarque aussi très clairement la présence de deux clusters, l'un comportant les variables rapides (Supernovas et Kilonovas) et l'autre comportant les variables lentes (Variable Star, Cataclysmic Variable et AGN). Les variables lentes sont les astres dont les variation sont relativement lentes au cours du temps contrairement aux variables rapides dont la luminosité croit et décroît très rapidement, notamment car celle-ci correspond à une explosion. Il n'y a que quelques liens entre ces deux clusters ce qui est satisfaisant aussi. Au sein du cluster des variables rapides, on observe une bonne séparation entre les kilonovas (KN) et les supernovas (SN) bien qu'il y ait quelques liens entre ces deux catégories. On n'obtient pas une séparation aussi nette pour le cluster des variables lentes mais on remarque qu'il y a une forme de cycle entre ces variables qui est très intéressante.

Précision	Densité de similarité
64%	61%

Pour ce qui est des performances, on obtient des résultats beaucoup plus probants que dans la partie 4.1 avec une très bonne précision de 64% et 61% de densité de similarité.

Notons que 78 % des alertes sont reliées en majorité à des alertes de la même classe. Autrement dit, si on cherchait à prédire la classe d'un objet dans le graphe, on aurait environ quatre chances sur cinq d'obtenir la bonne classe.

Finalement, l'approche via un réseau de neurones semble être une bonne approche. Nous aurions pu la creuser en complexifiant le réseau ou en augmentant la taille du jeu d'entraînement. Comme dit précédemment, le but initial n'est pas d'obtenir un réseau "parfait" mais de relier les alertes les plus similaires entre elles, ainsi une densité de similarité de l'ordre de 60% est tout à fait satisfaisante.

5 Anomalies

Fink possède un module de détection d'anomalie qui, en se basant sur la courbe de lumière d'une alerte, va octroyer un score à cette dernière [1]. Ce score permettra de définir un caractère anormal. Nous allons donc rajouter les dix alertes avec le plus haut score d'anomalie pour voir comment ces dernières vont se placer dans le graphe. Nous allons, en outre, rajouter vingt alertes de classe *Ambiguous* prises dans le jeu de données initiales tout en rappelant que la plupart sont des supernovas qui ont mal été identifiées par un autre classifieur.

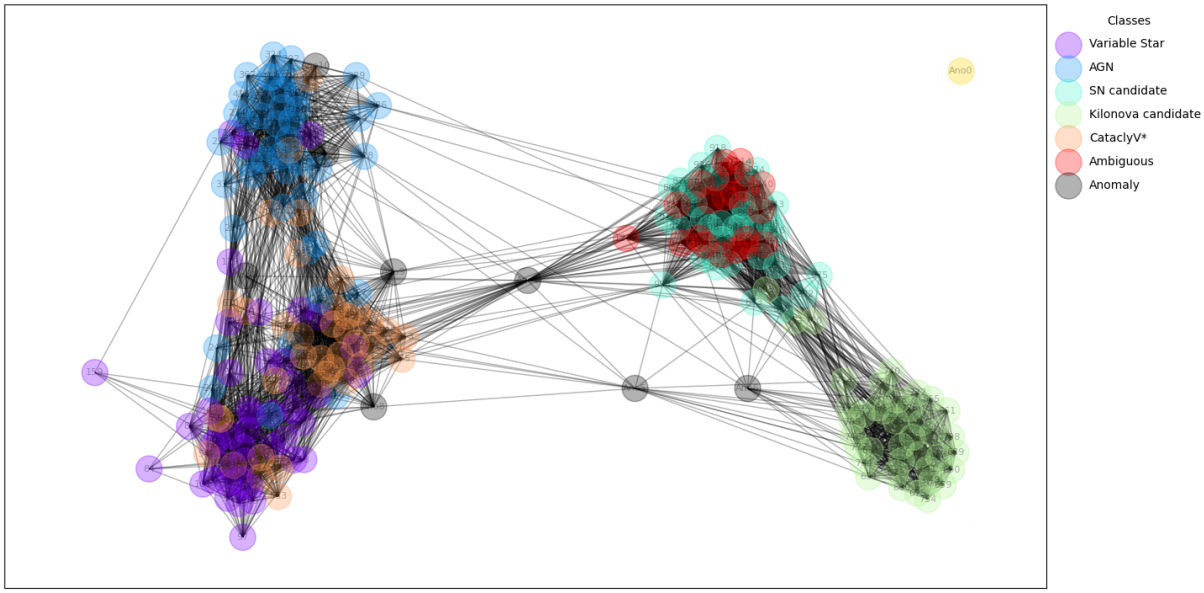


Figure 7: Adding anomalies and ambiguous alerts

Dans un premier temps, on remarque que les anomalies ont chacune des comportements qui leur sont propres. En effet, certaines se greffent au cluster des variables lentes tandis que d'autre se rattachent au cluster des variables rapides. Certaines anomalies tissent des liens entre ces deux clusters et présentent un comportement des plus intéressants. Cette approche permet de passer d'une valeur scalaire caractérisant une anomalie aux liens qu'elle crée à une position parmi un ensemble d'anomalie et ajoute ainsi une information pertinente supplémentaire.

Malheureusement, il est complexe de vérifier la réelle nature de ces anomalies simplement à partir des données Fink et il faudrait une étude plus approfondie pour pouvoir conclure.

Nous avons rajouté en outre une anomalie du nom de Ano0 en haut à droite de la figure 7. Elle correspond à une alerte de classe AM CVn WZ Sge qui est un type d'astre rare, seulement 4 anomalies de ce type ont été détectées à ce jour [3]. Cette anomalie est donc une "vraie" anomalie et est isolée du reste des autres alertes ce qui est un point intéressant.

En effet, si les comportements d'une alerte sont vraiment anormaux elles n'aura aucun lien avec les autres alertes ce qui permettra de les isoler avec une grande facilité.

Les alertes *Ambiguous* se rassemblent toutes autour du cluster des supernovas et leurs voisins sont, en moyenne, à 90% des supernovas ce qui confirme de manière plus formelle l'étude de ces ambiguous et permet de se rassurer quant à la qualité du graphe et à sa capacité à isoler les éléments uniques et à regrouper les éléments semblables.

6 Recherche sur les GNN

Les Graphs Neural Networks abrégé en GNN sont un type particulier de réseau de neurones qui a pour but de traiter des données sous forme de graphe en entrée. Il existe trois types de GNN:

- Graph-level task: L'objectif est de prédire la propriété d'un graphe entier. Par exemple, pour une molécule représentée sous forme de graphe, nous pourrions vouloir prédire l'odeur de la molécule ou si elle se liera à un récepteur impliqué dans une maladie.
- Node-level task: Ils consistent à prédire l'identité ou le rôle de chaque nœud d'un graphe. Pour une image par exemple, cela reviendrait à trouver le rôle de chaque pixel.
- Edge-level task: Le but est d'étudier le sens des arêtes et ce qu'elles apportent. En continuant avec une image, cela permettrait de décrire l'image via les interactions des éléments dans cette dernière.

C'est le dernier type qui va nous intéresser. En effet, le but est d'analyser les arêtes, leurs fonctionalités ainsi que leur pertinence.

En se basant sur la présentation [6] sur l'article [7], nous avons voulu mettre en place un GNN sur le graphe obtenu sur la figure 6 dans le but d'analyser la pertinence de chaque arête. Le but ne serait plus de regarder seulement une paire de neurones mais pour une paire de nœuds reliés par une arête, regarder un membre de la paire avec les voisins de l'autre nœud.

Nous nous sommes notamment inspiré de la slide n°5 de la présentation [6] et du module de transmission de messages pour pouvoir agréger l'information contenue dans les neurones pour pouvoir l'exploiter et actualiser les arêtes.

L'article [7], nous a permis de créer les fonctions f et g que nous expliciterons par la suite et garantir la convergence des algorithmes.

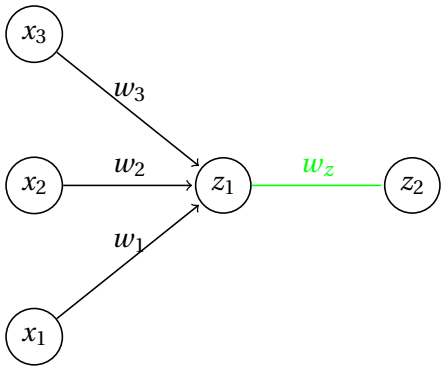


Figure 8: Étude de l'arête verte

Par exemple, sur la figure 8, on peut chercher à étudier la pertinence de l'arête verte, pour cela nous allons agréger l'information des voisins de l'arête avec une fonction d'aggrégation f.

Cette fonction prendrait en entrée deux ensembles :
les nœuds voisins E (= (x_1, x_2, x_3) ici) ainsi que les poids W (= (w_1, w_2, w_3) ici) les reliant.

Une définition possible de f serait:

$$f(E = (x_1, ..., x_n), W = (w_1, ..., w_n)) = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i} \tag{2}$$

Il est important de rappeler que les x_i sont des vecteurs de $\mathbb{R}^{20}$ et que, de ce fait, f effectue la moyenne arithmétique composante par composante des x_i.

Une autre fonction d'agrégation pourrait être choisie. Par exemple, une moyenne harmonique ou même une fonction comportant des poids ajustables.

Par la suite le but serait le suivant : à partir d'une autre fonction g prenant en entrée le score de l'arrête, la sortie de la fonction d'agrégation ainsi que les composantes de l'autre nœud pour actualiser le score de l'arête et ainsi de consolider les bonnes arêtes et supprimer les autres.

Le but aurait été dans un premier tant d'entraîner un réseau sur la fonction g pour converser les bonnes arêtes et ainsi converser la même densité de similarité, tout en augmentant la précision du graphe. L'objectif étant qu'en appliquant cette méthode plusieurs fois on arrive a un graphe stable avec une précision de l'ordre de 90%.

Par manque de temps, nous n'avons pas pu mettre cette méthode en place mais, au vu du temps consacré ci-dessus, il semblait important de mentionner le travail effectué.

7 Fonction de perte

En plus des GNN, nous avons voulu essayer différentes approches que nous n'avons pas eu le temps de concrétiser mais que nous aborderons dans cette section.

Dans un premier temps, nous avons voulu penser à changer la fonction de perte car notre réseau de neurones prédisait principalement des valeurs de similarité de 0 ou 1. Cela ne facilitait pas l'interprétabilité du résultat et ne permettait pas une mesure de similarité mais cela agissait comme un classifieur binaire alors que nous espérions plus une répartition plus homogène.

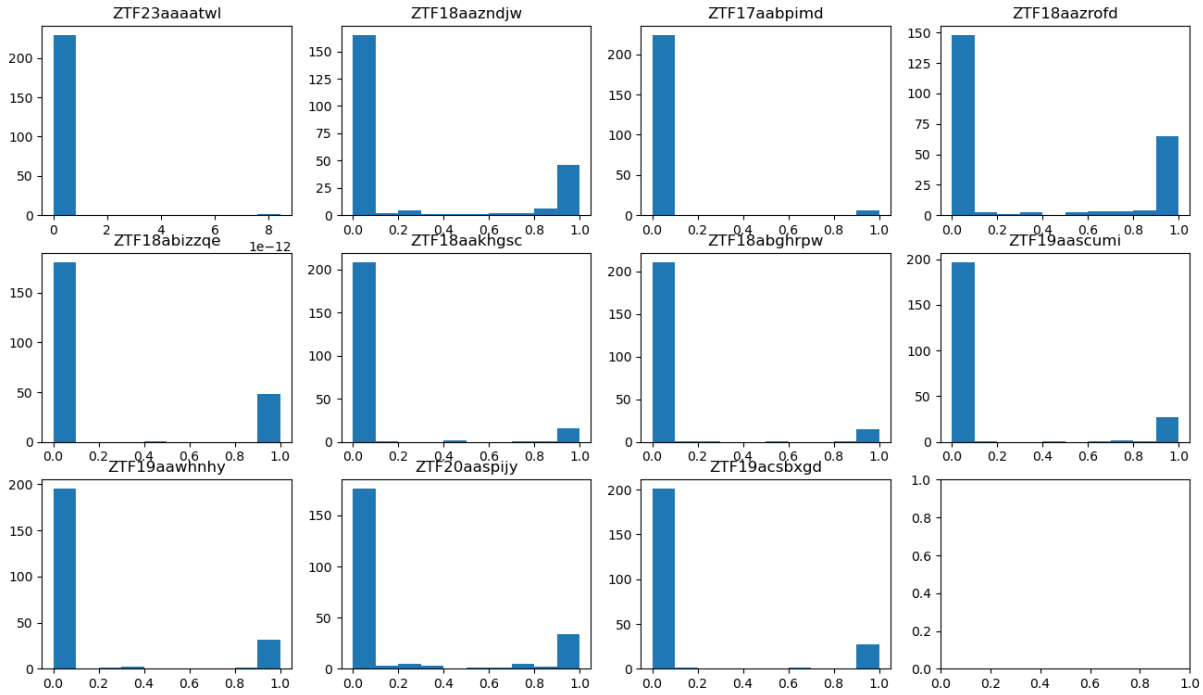


Figure 9: Histogramme des similarité des anomalies

Par exemple, sur la figure 9, sur laquelle on a représenté, pour chaque anomalie, l'histogramme des prédictions faites par le réseau de neurones pour chaque autre alerte dans le réseau de neurones. Par ailleurs, le titre correspond à l'id de l'objet mais en allant de gauche à droite puis de haut en bas on retrouve les anomalies numérotées respectivement dans l'ordre de 0 à 10.

La fonction de perte que nous avons utilisée est la Binary Cross-Entropy qui, pour un label y et une prévision $\tilde{y}$, s'exprime de la manière suivante:

$$BCE(y,\tilde{y}) = -(y\log\tilde{y} + (1-y)(1-\log 1-\tilde{y})) \tag{3}$$

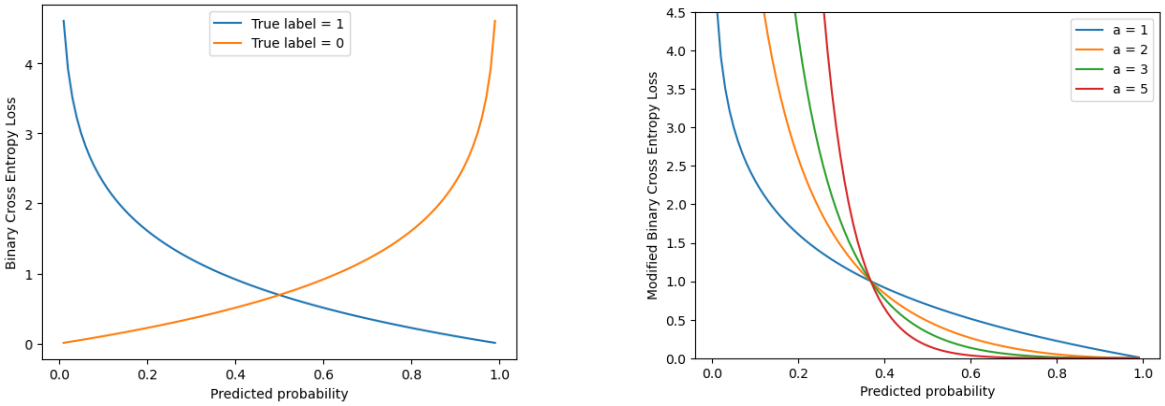
Nous avons précédemment passé sous silence l'indétermination de cette fonction de perte quand la prédiction est diamétralement opposée au label, mais dans l'implémentation, la BCE s'exprime avec un seuil de cette manière:

$$B\tilde{C}E(y,\tilde{y}) = \max(100, -(y\log\tilde{y} + (1-y)(1-\log(1-\tilde{y})))) \tag{4}$$

Et on peut graphiquement la représenter comme sur la figure 10a.
Notre objectif a été de faire une fonction moins punitive quand la prédiction se rapproche d'un ou deux dixièmes du label, nous avons essayé la formule suivante avec a comme paramètre:

$$BCE_{\text{modified}}(y,\tilde{y}) = (-(y\log\tilde{y} + (1-y)(1-\log(1-\tilde{y}))))^{a} \tag{5}$$

En gardant le même seuil numérique à 100 pour ne pas faire exploser la fonction de perte aux alentours de 0, en faisant varier a comme sur la figure 10b (et en fixant y à 1, $y = 0$ s'obtenant par symétrie), on peut faire tendre la fonction de perte plus rapidement vers 0. Cela permet ainsi de ne pas pénaliser les résultats tel que 0.9 par exemple. En outre, on remarque que le point d'inflexion de cette courbe se situe à e^{-1} qui vaut environ 0.36. En appliquant un facteur multiplicatif α à la BCE, nous pourrions modifier ce point et l'ajuster en fonction des résultats obtenus.



(a) Binary Cross Entropy

(b) Binary Cross Entropy Modified

Figure 10

Nous n'avons pas eu le temps de mettre en place et d'évaluer cette méthode mais cette approche nous semblait pertinente pour augmenter l'interprétabilité des résultats et la mettre en place dans un avenir proche.

8 Conclusion

Durant ce stage, nous avons procédé à un travail très exploratoire sur un domaine qui n'est que peu étudié pour le moment dans l'astrophysique en proposant une représentation de données sous la forme de graphe au lieu d'une représentation classique sous la forme d'un tableau. Nous avons successivement essayé de nombreuses méthodes qui n'ont pas abouties et n'avaient pas nécessairement leur place dans ce rapport.

Finalement, au cours de ces quatre mois, nous avons développé une preuve de concept basée sur un réseau de neurones qui nous permet, à partir d'un échantillon d'alertes, de créer un graphe qui relie les alertes similaires et permet de détecter des caractères anormaux parmi ces alertes.

A partir du jeu de données nous avons proposé une transformation pour réduire le nombre de composantes en passant de plus de deux cents features à vingt composantes issues d'une PCA, ce qui permet, si de nouvelles features sont ajoutées, de pouvoir réitérer la même méthode en ré-entraînant le modèle déjà existant.

Après avoir décrit deux constructions différentes, une basée sur la distance euclidienne et l'autre sur un réseau de neurones, nous avons pu écarter la première et obtenir un résultat concluant avec la seconde qui permet de relier les alertes entre elles de façon pertinente et permet d'ajouter de nouvelles fonctionnalités pour Fink ainsi que ces utilisateurs, telles qu'un système de recommandation et la détection d'anomalies.

References

[1] "Anomaly detection in the Zwicky Transient Facility DR3". In: (2020). K. L. Malanchev, M. V. Pruzhinskaya, V. S. Korolev, P. D. Aleo, M. V. Kornilov, E. E. O. Ishida, V. V. Krushinsky, F. Mondon, S. Sreejith, A. A. Volnova, A. A. Belinski, A. V. Dodin, A. M. Tatarnikov, S. G. Zheltoukhov. URL: https://arxiv.org/abs/2012.01419.

[2] E. Bertin and S. Arnouts. "SExtractor: Software for source extraction." In: 117 (June 1996), pp. 393–404. DOI: 10.1051/aas:1996164.

[3] Keisuke Isogai et al. "NSV 1440: first WZ Sge-type object in AM CVn stars and candidates". In: 71.2, 48 (Apr. 2019), p. 48. DOI: 10.1093/pasj/psz018. arXiv: 1901.11425 [astro-ph.SR].

[4] Anais Möller et al. "FINK, a new generation of broker for the LSST community". In: 501.3 (Mar. 2021), pp. 3272–3288. DOI: 10.1093/mnras/staa3602. arXiv: 2009.10185 [astro-ph.IM].

[5] *Novel deep learning methods for track reconstruction*. Steven Farrell, Paolo Calafiura, Mayur Mudigonda1, Prabhat, Dustin Anderson, Jean- Roch Vlimant, Stephan Zheng, Josh Bendavid, Maria Spiropulu, Giuseppe Cerati, Lind- sey Gray, Jim Kowalkowski, Panagiotis Spentzouris, and Aristeidis Tsaris. 2018. URL: https://ui.adsabs.harvard.edu/abs/2018arXiv181006111F/abstract.

[6] *Performance of GNN-based tracking for ATLAS ITk*. Xiangyang Ju. 2023. URL: https://indico.jlab.org/event/459/contributions/11414/.

[7] *The Graph Neural Network Model*. Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. URL: https://ro.uow.edu.au/cgi/viewcontent.cgi?article=10501&context=infopapers.