

Détection de l'Astroturfing sur les réseaux sociaux

Supervisor : Henri Verdier

Antoine Barré



1 Abstract

Astroturfing detection in online social networks is a long-lasting challenge which involves the design of detection techniques capable of identifying social networks phenomenon who might be manipulated. The primary research in the domain attempted to classify accounts based on their characteristics and induce whether or not they might be colluding. Recently, more attention has been given to capture global behaviour of multiple accounts and identify if they are attempting to manipulate twitter's diffusion in some ways. In this report, we focus on astroturfing detection in Twitter using the techniques of outlier detection. We apply various techniques to data consisting in groups of retweeters and succeed in finding concrete examples of political manipulation.

La détection de l'astroturfing dans les réseaux sociaux en ligne est un défi de longue date qui implique la conception de techniques de détection capables d'identifier les phénomènes de réseaux sociaux qui pourraient être manipulés. Les premières recherches dans le domaine ont tenté de classer les comptes en fonction de leurs caractéristiques et d'en déduire si c'étaient ou non des comptes robots. Récemment, une plus grande attention a été accordée au comportement global de plusieurs comptes et à l'identification de leur éventuelle collusion. Dans ce rapport, nous nous concentrerons sur la détection de l'astroturfing sur Twitter et les techniques d'identification d'anormalité. Nous appliquons des techniques d'Outlier Detection sur des données consitant en des *groupes de retweeters*. Nous arrivons à identifier des groupes anormaux véhiculant des messages politiques.

Keywords : Astroturfing detection ; bot detection ; Twitter ; Outlier Detections

2 Remerciements

Je tiens d'abord à remercier Henri Verdier, Ambassadeur du Numérique au Ministère des Affaires Étrangères, pour m'avoir permis de réaliser ce stage.

Je remercie aussi toute l'équipe de l'ambassade du numérique - Marine Guillaume, Jérémy Hureaux, Valérie Marzo, Mahe Dersoir, Elsa Trujillo, Paul Schmite, Martin Ratinaud et Clément Biron - pour m'avoir accueilli chaleureusement et m'avoir aidé dès que cela était nécessaire !

Contents

1 Abstract	1
2 Remerciements	2
3 Introduction	5
3.1 Motivation de l'étude	5
3.2 Plan	6
4 Revue de littérature	6
4.1 Botometer	7
4.1.1 Remarques	8
4.2 CatchSync	8
4.2.1 Remarques	9
4.3 LSTM architecture	9
4.3.1 Remarques	10
4.4 DeBot	10
4.4.1 Remarques	10
4.5 RTBust	11
4.5.1 Remarque	11
4.6 Social Fingerprinting	12
4.6.1 Remarque	13
4.7 Remarques générales sur la revue de littérature	13
4.7.1 Problème de <i>ground truth</i>	14
4.7.2 Variété des types de compte et d'attaques	14
4.7.3 Utilisation du contenu des tweets	14
4.7.4 Datasets	14
5 Présentation des données	15
5.1 Collecte des données	15
5.2 Statistiques observées sur les groupes de retweeters	16
5.3 PCA et clustering	20
6 Résultats	21
6.1 Distance de Mahalanobis	21
6.2 Premiers résultats	21
6.2.1 Jeux	21
6.2.2 Promotion Commerciale	23
6.2.3 Activités exotiques	24
6.2.4 Activités politiques	24
6.3 Méthodes d'identification d'outliers	26
6.4 Evaluation	30
6.4.1 Mass volume curves and anomaly ranking - Stephan Clémenton and Albert Thomas [11]	30
6.4.2 On Anomaly Ranking and Excess-Mass Curves – Nicolas Goix, Anne Sabourin, Stéphane Clémenton [36]	31
6.4.3 Résultats de l'évaluation	32
6.5 Avantages	32

7 Conclusion	33
7.1 Limites et améliorations	33
7.1.1 Limites	33
7.1.2 Clusteriser	33
7.1.3 Revenir aux comptes	33
7.1.4 Autres features	33
7.1.5 Focalisation sur la politique	34
7.1.6 Ensembling	34

3 Introduction

Les médias sociaux sont des plateformes qui permettent aux gens de partager, de communiquer et de collaborer. Bien que les gens apprécient l'ouverture et la commodité des médias sociaux, de nombreux comportements malveillants, tels que l'intimidation, la planification d'attaques terroristes et la diffusion d'informations frauduleuses, peuvent se produire.

Avec l'intensification des usages des réseaux sociaux ce type de menaces est en train de prendre une place de plus en plus importante pour définir l'activité politique de nos sociétés. Une pratique particulièrement dangereuse est de diffuser des contenus de manière inorganique pour en amplifier la propagation. Cette pratique, appelée *astroturfing* [24] s'est de plus en plus répandue pour promouvoir du contenu de toutes sortes. Les campagnes d'*astroturfing* peuvent ou non s'appuyer sur des *bots* (c'est à dire des comptes twitters opérés par une machine) : elles présentent dans les deux cas un degré de coordination anormal entre les différents acteurs cherchant à promouvoir le contenu.

Il faut noter que la proportion des comptes twitters qui sont des bots est estimé entre 9 et 15 % [40]. Malgré une amélioration de la qualité de la détection de ces comptes par Twitter [38], il reste qu'un grand nombre de bots, sophistiqués ou non [17], ne sont pas repérés.

Dans le cadre des fonctions du Ministère de l'Europe et des Affaires Etrangères, ces questions sont particulièrement intéressantes dans deux cas d'usage :

- Les tentatives d'*astroturfing* d'un candidat ou d'un parti pour donner une influence disproportionnée à sa parole médiatique.
- Les tentatives d'ingérence de la part de pays étrangers dans la politique interne du pays. Il faut noter que certains pays en ont fait une pierre angulaire de leur stratégie de puissance [9], [31], [5].

Les opérations d'influences menés par tous les acteurs sont de plus en plus structurés et des réseaux dédiés commencent à apparaître dans de nombreux pays et institutions pour les conduire. Une des caractéristiques nouvelles des opérations d'influence modernes est qu'elles s'appuient fortement sur les réseaux sociaux pour déstabiliser et propager des valeurs non démocratiques ou de la désinformation.

Le fonctionnement des démocraties libérales fait qu'elles sont particulièrement sensibles à ce type d'attaques, du fait de la liberté garantie à tous d'exprimer ses opinions. Il est donc important de détecter ce type de campagne le plus tôt possible pour pouvoir leur opposer des réactions adéquates.

3.1 Motivation de l'étude

Nous nous focalisons sur Twitter, comme d'ailleurs la plupart de la littérature, car il s'agit du réseau dont les données sont les plus facilement accessibles. De plus le mode de fonctionnement de ce réseau avec le format retweets permet plus facilement qu'ailleurs d'observer les communautés[18]. En se focalisant sur twitter nous disposons de ce fait de bien plus d'informations et d'une information plus structurée que celle que nous aurions sur d'autres réseaux.

Notre objectif est donc d'essayer de caractériser les opérations de retweet massif sur twitter en analysant les profils des individus qui ont retweetés et de voir si on peut observer des différences dans le cas où ces processus ne seraient pas organiques.

Notre idée majeure est donc d'étudier les groupes de retweeters pris dans leur ensemble. Pour cela nous extrayons de l'ensemble des retweeters d'un tweet un certain nombre de caractéristiques détaillées plus bas. Nous obtenons ainsi une représentation d'un groupe de retweeters que nous compressons à l'aide d'un VAE. La représentation latente du VAE nous fournit une représentation plus riche de chaque groupe de retweeters. Nous utilisons ensuite cette représentation latente pour

clusteriser les groupes de retweeters et observer ceux qui sont caractéristique ment différents des autres.

L'utilisation d'une approche non supervisée provient de plusieurs caractéristiques propres à cette étude, que nous détaillerons par la suite : absence de *ground truth* pour la reconnaissance de campagnes d'astroturfing, nécessité d'avoir plus un indicateur qu'un critère de vérités plus dur, nécessité d'avoir une approche globale de la question en étudiant des groupes plutot que des comptes individuels.

3.2 Plan

Nous nous consacrons d'abord (section 4) à l'étude de la littérature sur le sujet. Puis nous analysons les données dont nous disposons et la façon dont nous les avons recueilli (section 5) et enfin nous présentons les résultats de notre analyse (section 6).

4 Revue de littérature

La recherche sur la question de la manipulation de l'influence par le biais de réseaux sociaux est déjà largement traitée dans la littérature. Sur la question des bots uniquement, une revue de littérature de 2020 [12] recense 236 détecteurs de compte bots avec une variété de méthodologies et d'outils utilisés importante.

Figure 1: Etude sur les différents type de détecteurs de bot existants [Source : [12]]

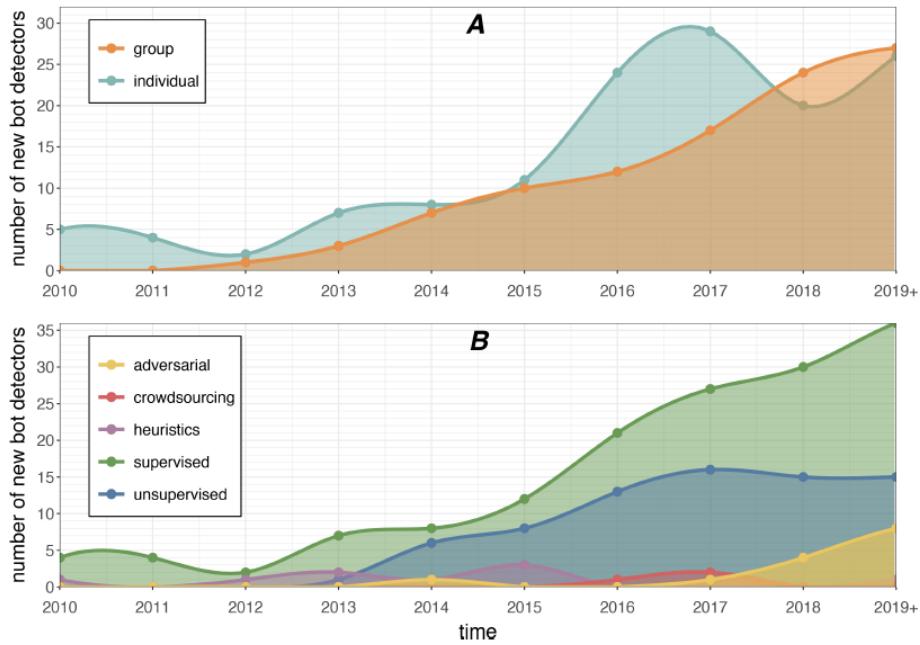


Fig. 5. Longitudinal categorization of 236 bot detectors published since 2010. Data points indicate the number of new detectors per type published in a given year. In panel A, detectors are classified as either focusing on the analysis of individual accounts, or on the analysis of groups of accounts. In panel B, the same detectors are classified based on their high-level approach to the task. Both panels clearly document the rise of a new approach to bot detection, characterized by group-analyses and many unsupervised detectors. Interestingly, the plateau reached by unsupervised approaches since 2017 occurred in conjunction with the recent rise of the adversarial ones.

Une autre revue de littérature [37] montre qu'un nombre croissant d'article est publié sur le sujet :

Figure 2: Nombre d'articles publiés sur la détection de bot sur Twitter [Source : [37]]

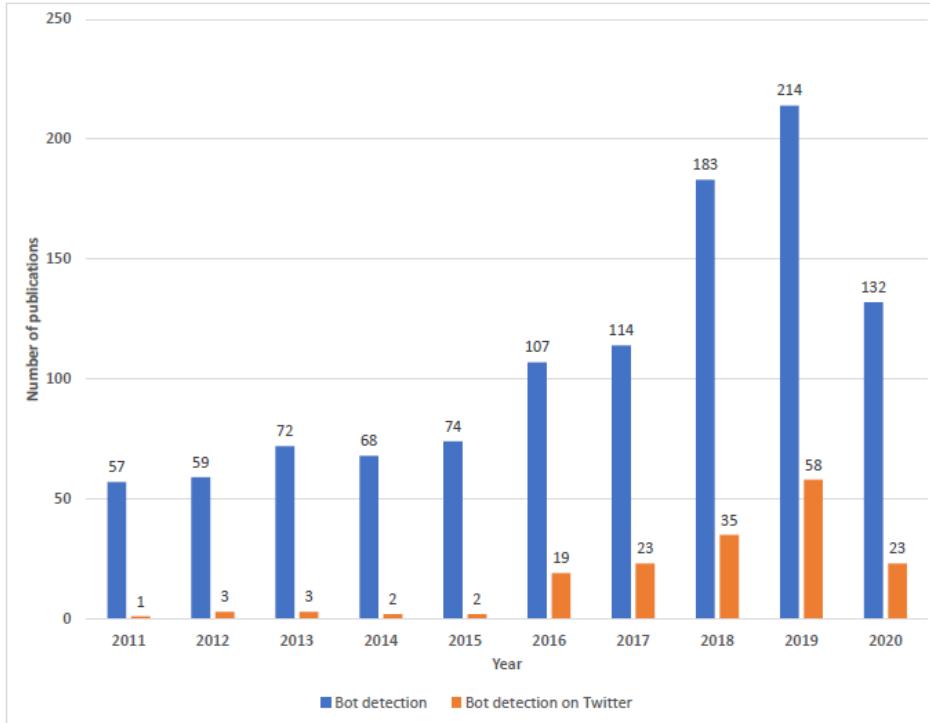


Figure 1. The number of publications on bot detection and bot detection on Twitter for the last ten years in Scopus. We observed a significant increase in interest in 2016 matching the controversy of the U.S. federal election. The data for 2020 were gathered until September.

Nous faisons ici une brève synthèse des publications et différentes méthodes utilisées dans le domaine en séparant. Nous nous concentrons sur toute la recherche qui cherche à repérer des comportements anormaux dans l'usage des réseaux sociaux et en particulier ceux qui s'occupent de l'usage de comptes robots ou bots.

4.1 Botometer

La méthode la plus connue et la plus utilisée provient du service en ligne botometer[14]. Leur approche bien que non totalement précis pour éviter le contournement est dans l'ensemble très claire : utiliser un classifier pour catégoriser le compte en bot ou non. L'algorithme utilisé est un Random Forest Classifier, entraîné sur une base de données de compte (30000 comptes répartis à moitié entre comptes robots et comptes humains). Plus de 1000 features sont récupérées sur chaque compte réparties entre plusieurs classes[41] :

1. Features de réseaux : l'algorithme construit plusieurs graphes en utilisant les retweets de l'utilisateur, les comptes qui le mentionne, les hashtags qui co-occurrent avec d'autres comptes et extrait des informations statistiques de ces graphes (distribution de degré, coefficient de clustering)

2. Features de l'user : l'algorithme récupère les données du compte en particulier la langue, la géolocalisation, la date de création du compte
3. Features des amis : l'algorithme extrait des informations statistiques (médiane, moyenne, entropie) sur la distribution du nombre de followers, de followees et de status du compte
4. Features temporelles : l'algorithme récupère de même des informations sur les distributions d'actions (tweets, retweets, quote tweets) du compte
5. Feature de content : l'algorithme récupère 200 anciens tweets du compte et cherche des indices de langage sur cet historique, en particulier des mots clés et des expressions clés. L'algorithme évalue également différents scores de sentiment analysis et les inclue dans les features

4.1.1 Remarques

La méthode de Botometer est critiquable à plusieurs égards et ses insuffisances ont été souvent remarquées[15]. Les deux problèmes les plus importants selon nous est le grand nombre de features comparés à la petitesse de l'échantillon d'entraînement : il est très probable qu'un grand nombre de features ainsi sélectionné (en particulier celles reposant sur le contenu des tweets du compte) soit en fait très peu robustes et ne permettent pas d'aller classifier des comptes d'un nouveau type.

Le deuxième problème est que la méthodologie telle qu'elle ait donnée n'a que peu de chances de trouver des classes de bots sur laquelle elle n'a pas été entraînée : or les programmes de bots sont modifiés de façon constante et au fond peu importe que l'algorithme classifie correctement des comptes dont tout le monde voit bien que ce sont des bots. Ce sont les bots qui sont invisibles aux usagers qu'ils nous importe de découvrir puisque c'est dans leur cas que les tentatives de manipulation ont le risque d'être plus importantes et convaincantes.

D'autres articles utilisent les mêmes méthodes avec des features différentes, par exemple [3] [16] [1] [42], parfois en composant cette analyse avec une PCA pour diminuer la dimensionnalité des données ou en étudiant les différences qui sont créées si on entraîne sur des sous-parties du dataset.

Une difficulté est d'évaluer l'efficacité de ces différents modèles étant donné la diversité des datasets. Bien souvent les modèles sont entraînés sur des datasets spécifiques et relativement petits et les sophistications proposées pour augmenter leur précisions ne sont pas forcément efficaces pour traiter toutes les catégories de bots ou d'attaques.

4.2 CatchSync

CatchSync [8] s'appuie sur les graphes pour essayer de repérer des comportements d'utilisateur synchrones. Ils appliquent leur méthodologie à différents réseaux sociaux : Twitter, Facebook, Weibo.

L'idée est de repérer les comptes suspicieux en les caractérisant à partir de ceux qu'ils suivent. Dans le cas de Twitter, on se donne le graphe orienté où les noeuds sont les comptes d'utilisateurs et les arêtes représentent la relation "suivre quelqu'un".

Ce qui arrive souvent lorsque les comptes de bots se créent est qu'un groupe de bots se suivent entre eux d'une façon repérable par rapport à l'ensemble de la population. En effet, un compte normal suivra des comptes de toute taille : ses amis proches qui auront peu de gens qui les suivent et des plus gros comptes qui auront beaucoup de followees mais suivront eux mêmes peu de gens. Dans le cas d'un groupe de bots, ils se retrouveront souvent à s'entresuivrent entre eux et la distributions des degrés sortant de leurs amis sera donc anormale.

Pour mettre en place mathématiquement ces idées, l'approche de CATSYnc est d'utiliser des caractéristiques du graphe à calculer notamment des idées de [27] qui traduisent la "centralité" d'un point. Ils attribuent un score au noeud du graphe en fonction de sa proximité (en terme de features, ici justement la centralité) avec ses voisins. Un point qui est trop proche de ses voisins est considéré comme suspect.

4.2.1 Remarques

La méthode est très prometteuse et elle paraît très difficile à éviter. Dans le cas de Twitter il faudrait que les bots parviennent à imiter la même distribution de followers qu'un compte naturel ce qui est rendu difficile pour des raisons d'efficacité et de mise en place du réseau.

Malheureusement, les données à récupérer sur les followers sont difficiles à obtenir par twitter du fait des limitations imposées par l'API officielle ; nous ne connaissons pas de méthodes permettant de contourner ces limites.

Une autre remarque est que la caractérisation mathématique utilisée est un peu arbitraire et qu'elle pourrait être renforcée en ajoutant d'autres caractéristiques aux noeuds. Pour le dire autrement : l'espace des features qu'on a utilisé pourrait être agrandi pour mieux caractériser cette notion de proximité entre les noeuds.

Deux méthodes intéressantes sont développées dans l'article pour vérifier que l'on parvient bien à repérer les anomalies.

La première consiste à injecter dans le dataset des points artificiels et de vérifier que l'algorithme continue de repérer ces points même si on essaye de les camoufler. (Dans le cas de l'article, il s'agissait de rajouter des groupes synchronisés et le camouflage consistait à les intégrer en partie au réseau organique)

La deuxième consiste à vérifier que des distributions manifestement anormales redeviennent analogues si on enlève les outliers. Le graphe ci-dessous tiré de[23] montre le genre de choses qu'on attend.

Figure 3: Graphe avant/après avoir enlever les comptes malicieux

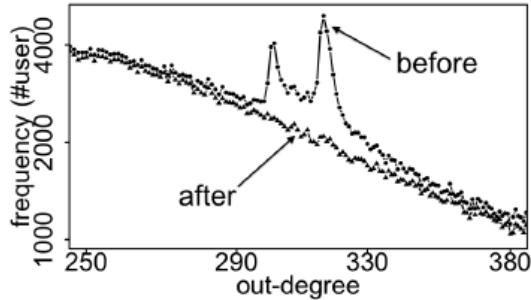


Fig. 6. The out-degree distribution (in log-log scale) becomes smoother after the removal of "lockstep" followers. The followers have similar out-degree values, i.e., similar numbers of followees from the same group.

4.3 LSTM architecture

Kuduganta & al. [29] construisent un détecteur de bots Twitter fondé sur une architecture LSTM. L'architecture mélange à la fois des métadonnées sur chaque tweets et un réseau LSTM classique.

Le détecteur est ensuite testée sur un des datasets de bots disponibles.

4.3.1 Remarques

Là encore il est très curieux que les métadonnées rajoutées (nombre de likes, nombre de retweets, nombre de reply) puissent ajouter quoi que ce soit à une détection des bots qui va au-delà de caractéristiques accidentelles.

La précision obtenue, qui est très bonne (97% AUC pour un seul tweet) semble en fait reposer sur de l'overfitting - en effet, il est impossible que cette méthode parvienne à repérer un bot si celui-ci reprend du contenu créé par un humain.

4.4 DeBot

Debot [10] est un système qui utilise uniquement la corrélation temporelle entre les actions des différents comptes étudiés. L'idée est d'utiliser la distance de Déformation temporelle dynamique (*Dynamic Time Warping*) - voir le graphique ci-dessous - entre les historiques d'activités de compte afin de caractériser leur synchronicité. Si deux comptes sont trop synchrones par rapport à l'ensemble, ils sont considérés comme des bots.

Figure 4: Illustration de la distance DTW entre deux séries temporelles [Source : [25]

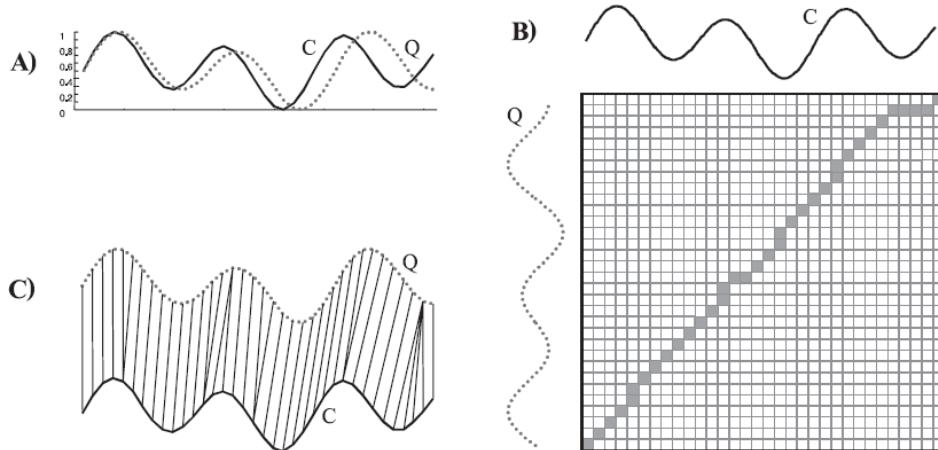


Fig. 3. A) Two sequences Q and C that are similar but out of phase. B) To align the sequences, we construct a warping matrix and search for the optimal warping path, shown with solid squares. C) The resulting alignment

Les calculs étant prohibitifs si on devait évaluer la distance entre tous les comptes étudiés, l'idée des auteurs est de hasher les historiques de comptes en calculant différentes valeurs de corrélation croisées avec un vecteur aléatoire. On peut ensuite comparer les comptes qui ont un grand nombre de valeurs de hashage en commun.

4.4.1 Remarques

L'idée est bonne et leur permet effectivement de repérer un grand nombre de bots. Le problème est que ceci sont plutôt frustres puisqu'il suffit de désynchroniser légèrement les comptes pour que la méthode échoue.

En implémentant une méthode à peu près similaire, il semblerait qu'aujourd'hui Twitter ait déjà mis en place un algorithme permettant de bannir les comptes trop synchronisés : nous ne repérons en effet que des comptes rapidement bannis. Par ailleurs ceux-ci n'avaient pas une activité très importante, ce qui semble montrer que les campagnes d'utilisation de comptes bots sont désormais plus sophistiquées.

4.5 RTBust

L'idée de RTBust [34] est d'observer les distributions de temps de retweets d'un utilisateur donné pour repérer les distributions anormales.

Concrètement les distributions de temps de retweet sont compressées à travers un VAE puis on réalise un clustering sur les représentations latentes associées aux utilisateurs. De cette manière, les groupes de bots synchronisés ayant normalement des historiques de retweet très similaire, les bots d'un même groupe de bots devraient être clusterisés relativement proches les uns des autres et les différents groupes synchrones peuvent donc être repérés et analyser à la main pour vérifier leurs états. Le schéma de la technique est ci-dessous :

Figure 5: Schéma du modèle utilisé par RTBust [Source : [34]]

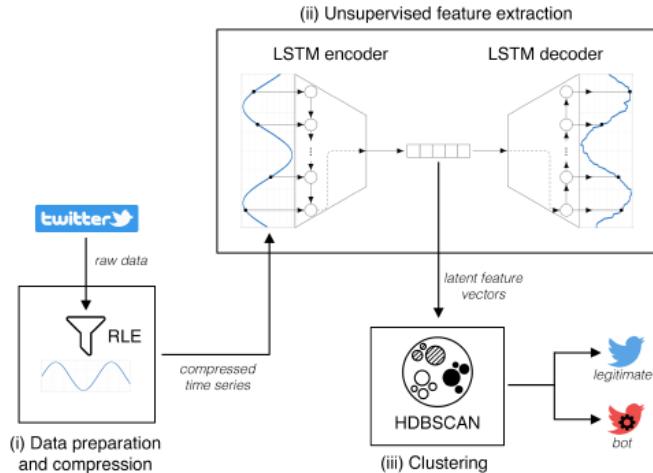


Figure 5: Main logical components of RTBUST.

4.5.1 Remarque

C'est l'approche qui nous a le plus inspiré pour nos propres expériences. La principale critique est que la méthode semble facilement contournable : il suffit que les bots soient légèrement asynchronisés pour que le

Néanmoins l'idée de chercher des anomalies dans les distributions des caractéristiques des usagers twitter est très bonne. Particulièrement nous avons été frappé par ce graphique de visualisation des temps de retweets pour un utilisateur donné :

Figure 6: Visualisation des temps de retweets pour un utilisateur donné [Source : [34]
WebSci 2019, June 30–July 3, 2019, Boston, MA, USA M. Mazza et al.

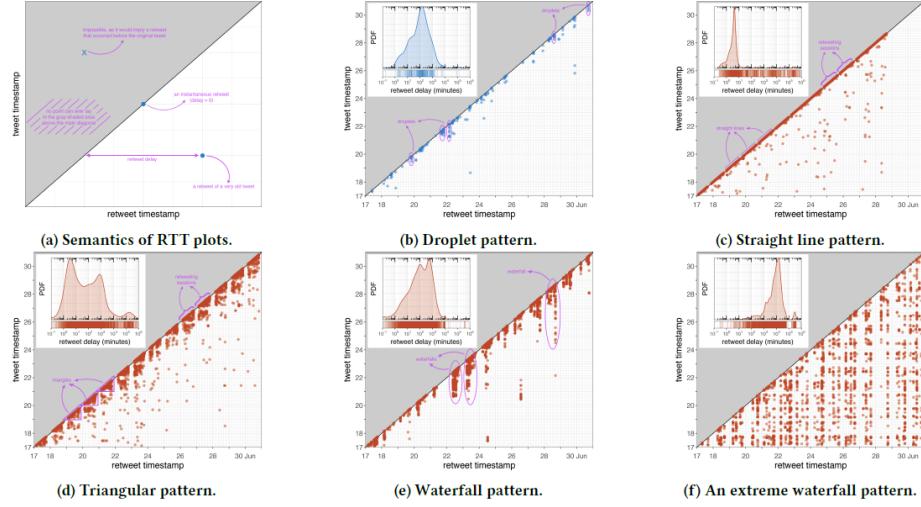


Figure 4: RTT plots depicting a normal tweeting behavior (blue colored) and many suspicious ones (red colored). The insets of RTT plots show the empirical probability density function (PDF) and a rug plot of retweet delays.

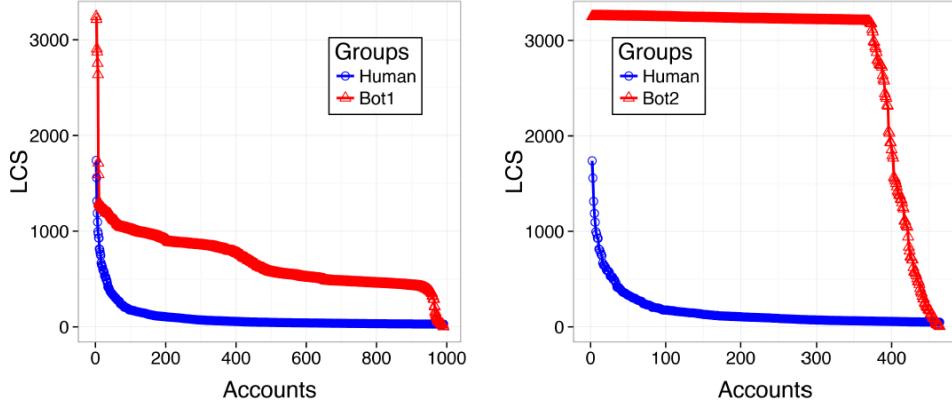
Les principales différences de notre travail avec celui-ci sont de nous concentrer sur les retweets pour un tweet donné plutôt que par utilisateur ; ceci nous permet également d'incorporer plus d'informations venant des retweeters - détaillées plus bas - et non plus seulement le temps entre le tweet et le retweet. L'avantage principal est qu'il nous paraît plus difficile de simuler une distribution normale sur les temps de retweets d'un tweet donné (et toutes les autres composantes !) que de désynchroniser l'action des bots.

4.6 Social Fingerprinting

L'approche de [13] est elle aussi basée sur l'historique d'activité du compte. L'idée ici est d'utiliser la séquence non temporelle des actions du compte pour essayer de repérer des groupes de bots dont l'historique des actions est similaire. Pour établir les séquences d'actions, l'article utilise comme alphabet à la fois des actions directement différenciés par Twitter (Tweet, Retweet, Quote Tweet) mais aussi des catégories *ad hoc* (Tweet contenant une url, un hashtag, etc...).

Ces séquences d'actions sont appelées *Digital DNA*. L'idée est de chercher dans un groupe de comptes la sous séquence commune la plus longue et de voir comment la longueur de cette sous-séquence évolue avec le nombre de comptes rajoutés. Chercher dans un ensemble de digital DNA la sous séquence commune la plus longue est un exemple de *k-common substring problem* qui est soluble en temps et en espace linéaire [4] par rapport au nombre de séquences étudiées. Le graphique ci-dessous fait la comparaison entre un groupe de bots et un groupe d'utilisateurs normaux.

Figure 7: Comparaison de l'évolution de la longueur de la LCS en fonction du nombre de comptes observés (Source : [13])



(a) Bot1 versus human accounts. (b) Bot2 versus human accounts.

Fig. 3. Comparison between LCS curves of spambots and genuine accounts for the \mathbb{B}_{type}^3 alphabet.

4.6.1 Remarque

La méthode est intéressante mais elle présente des caractéristiques qui la rendent difficile à exploiter en pratique.

D'abord dans l'article ne sont utilisés que quelques milliers de comptes et Twitter n'autorise la remontée d'historique que pour les 3000 dernières actions : si l'on cherche à utiliser cette technique pour repérer des comptes dans la nature, le grand nombre de comptes récupérés fera apparaître des faux positifs.

Par ailleurs un grand nombre de comptes utilisent les mêmes actions sans pour autant appartenir à un groupe coordonné : par exemple les comptes qui ne font que retweeter les tweets des autres.

Pour palier ces difficultés, il nous paraît utile de rajouter la composante temporelle de l'activité historique. Mais la technique de chercher la sous-séquence commune la plus longue ne pourrait plus être utilisée.

Ce qui nous semble intéressant est de générer un indicateur permettant de comparer les historiques de comptes, en utilisant par exemple une représentation latente de ces historiques qui incluraient les parties temporelles et les données des activités. Des VAE pour les séquences points (des séries temporelles mais comprenant des données d'actions à chaque point) ont déjà été développés[35] et l'approche pourrait être réutilisée dans notre cadre.

4.7 Remarques générales sur la revue de littérature

Plusieurs difficultés liées à notre problème apparaissent à la suite de cette revue de littérature. Nous les discutons ci-dessous.

4.7.1 Problème de *ground truth*

Un des problèmes les plus importants est que nous n'avons pas accès à la vérité finale. Cela est vrai non seulement si l'objectif est de discriminer les comptes robots des autres puisque nous n'avons pas accès à tous les comptes robots. C'est encore plus vrai pour ce qui est des attaques et des tentatives de manipulation de l'information[33]. Identifier une campagne de manipulation requiert un certain arbitrage de la part du chercheur sans que l'on voit bien quel sens quantitatif on pourrait donner pour indiquer qu'il s'agit d'une campagne abusive.

4.7.2 Variété des types de compte et d'attaques

Une autre difficulté qui se rajoute à la précédente est qu'une grande variété de cas existent : les comptes peuvent être des bots très peu sophistiqués ou au contraire imiter pour ce qui est de leur comportement assez fidèlement des humains ; ils peuvent être opérés parfois par des humains et parfois par des robots[21]. Pour ce qui est des tentatives de manipulation, il peut y en avoir de plusieurs sortes : soit le même message peut-être répétés et retweetés par un grand nombre de comptes, soit une campagne peut-être entièrement lancée par des humains à un moment précis et il peut être difficile de la distinguer alors d'une campagne de communication légitime - rien après tout n'interdit la concertation. Il faut donc définir un critère pour définir ce qu'on va chercher.

4.7.3 Utilisation du contenu des tweets

Beaucoup d'articles (par exemple : [29], [33], [17]) utilisent le contenu des tweets pour inférer si l'utilisateur est un bot ou non. Bien qu'il est assez clair que cela augmentera la performance du modèle quoi qu'il arrive en rajoutant de l'information, dans le cadre où nous travaillons (repérer des amplifications de message à travers le réseau) rajouter l'information du contenu des tweets seraient contre-productifs.

En effet nous ne voulons pas que le modèle infère qu'un certain type de messages est plus susceptible d'être manipulé qu'un autre à priori. Notre but est de trouver les tentatives d'amplification quelque soit le message véhiculé.

4.7.4 Datasets

Nous parlons ici brièvement des datasets qui ont été utilisés dans la recherche. La plupart du temps ceux-ci contiennent un ensemble de tweets et les caractéristiques du compte twitter associé et un label les caractérisant comme bot ou humain. Plusieurs difficultés se posent.

1. Une grande partie des datasets eux ne sont pas disponibles publiquement.
2. Il est difficile de labeliser correctement les comptes.
3. Les informations contenues dans les datasets ne permettent pas forcément d'expérimenter toutes les méthodes, en particulier celles qui reposent sur des graphes.
4. Un danger considérable est d'entraîner uniquement sur ces datasets qui ont tendance à ne comprendre que les bots les plus évidents et donc peuvent biaiser les résultats sur des bots qui de plus ne sont pas forcément les plus influents.
5. Par ailleurs un grand nombre de bots de ces datasets sont déjà bannis ou supprimés.

Toutes ces raisons nous ont poussé à choisir une approche non-supervisée.

5 Présentation des données

5.1 Collecte des données

Nous avons d'abord utilisé l'API publique de Twitter pour récupérer les données de tweets et de retweets. Néanmoins celle-ci souffre de plusieurs limitations [39] :

- Le nombre de tweets observables est de 300 par période de 15 minutes
- Le nombre de requête de retweets (c'est à dire qui demande les identifiants des retweeters et leurs caractéristiques) maximal est de 75 par période de 15 minutes
- Enfin, il n'y a qu'au maximum que les 100 derniers retweets qui sont accessibles.

Ces limitations font que les méthodes utilisées ici seraient difficiles à mettre en place en pratique.

Nous utilisons donc snscreape un crawler qui permet de lancer des recherches twitters en mode anonyme, sans se connecter avec un compte spécifique. Le problème est encore que les informations sur les retweets ne sont pas disponibles de cette manière.

Pour contourner ce problème, nous utilisons donc les fonctionnalités de recherche avancée. Précisément, nous faisons deux choses :

- D'abord nous lançons une requête de recherche Twitter sur la fenêtre de temps considérée (ici, une semaine du 23 au 29 août) qui cherche uniquement les tweets ayant un nombre minimal de retweets (ici, 100) et dans une certaine langue (ici, français, allemand et italien) - concrètement la requête de recherche est : "min_retweets:100 lang:fr since:2021-08-23 until:2021-08-29"¹.
- Pour chaque textes de tweets obtenu de cette manière, nous recherchons ensuite tous les retweets de ce contenu grâce à la commande : "filter:nativeretweets [contenu du tweet]"

Nous obtenons de cette façon un ensemble conséquent de données : pour chaque retweeter d'un tweet nous avons ainsi accès à :

- la date de création du compte du retweeter
- le nombre de followers du compte du retweeter
- le nombre de personnes que suit le compte du retweeter
- le nombre de statuts postés par le compte du retweeter
- le nombre de tweets likés par le compte du retweeter
- la date de retweet du tweet en question

La donnée d'un groupe de retweeters est constitué de toutes ces informations pour chaque retweeter du tweet en question.

¹A noter que la recherche Twitter n'est pas parfaite et que certains tweets considérés comme en Français sont en fait écrits dans d'autres langues.

5.2 Statistiques observées sur les groupes de retweeters

Dans l'exemple en question nous obtenons 13736 tweets en français qui comptent plus de 100 retweets (ce choix est arbitraire, mais garanti que les groupes de retweeters étudiés ne sont pas trop petits) pour la période du 23 au 29 août 2021. Pour la même période nous choisissons de limiter à 10000 tweets repris en allemand et en italien. Pour un certain nombre de tweets la recherche n'a pas fonctionné². Ces choix arbitraires nous permettent d'avoir une donnée facilement manipulable à notre échelle. Le tableau ci-dessus récapitule les données recueillies.

Langue	Nombre de Tweets	Nombre de Retweets	Nombre de Retweeters uniques
Français	5371	2678695	758716
Allemand	2912	920120	336787
Italien	3506	1044713	396214
Total	11789	4643528	1491717

Nous affichons la distribution d'un certain nombre de caractéristiques ci-dessous.

²Certains caractères ne sont pas reconnus par la recherche twitter et il faut les enlever. Comme nous voulions simplement prendre un grand nombre de tweets pour voir si la méthode fonctionnait, nous n'avons pas cherché à faire en sorte de retrouver tous les retweeters quelque soit le contenu du tweet initial même si c'est possible.

Figure 8: Distribution du nombre de followers sur tous les retweeters

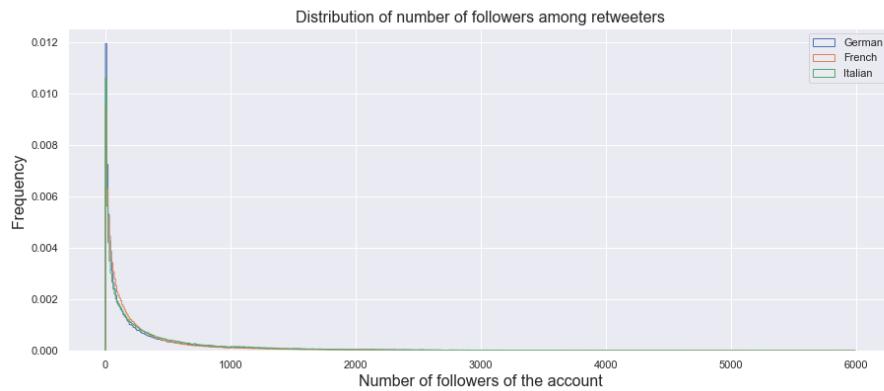


Figure 9: Distribution du nombre de friends (=followed) sur tous les retweeters

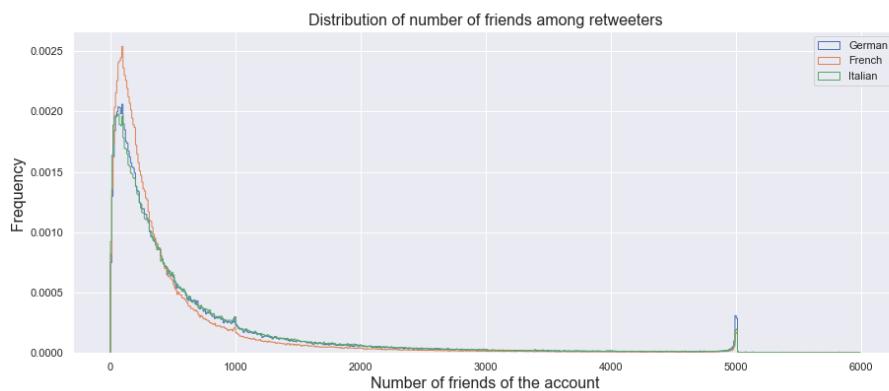


Figure	Moyenne	Ecart-Type	Min	25%	50%	75%	Max
1	876	1758	0	72	223	618	1325586
2	724	1510	0	15	324	734	36838

Figure 10: Distribution du nombre de like sur tous les retweeters

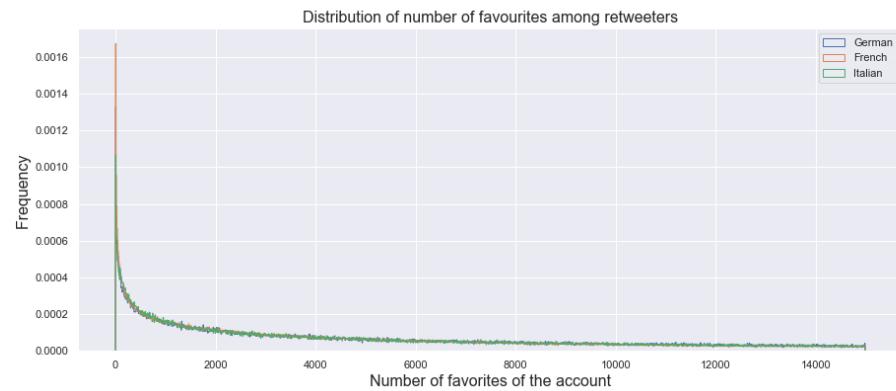


Figure 11: Distribution du nombre de status sur tous les retweeters

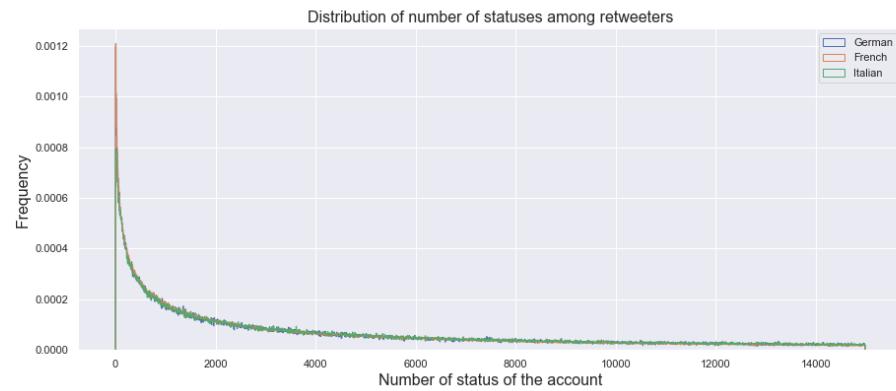


Figure	Moyenne	Ecart-Type	Min	25%	50%	75%	Max
3	32327	61339	0	3384	12187	36062	1680672
4	19129	46037	1	1097	4885	17800	2969689

Figure 12: Distribution de l'âge des comptes de tous les retweeters

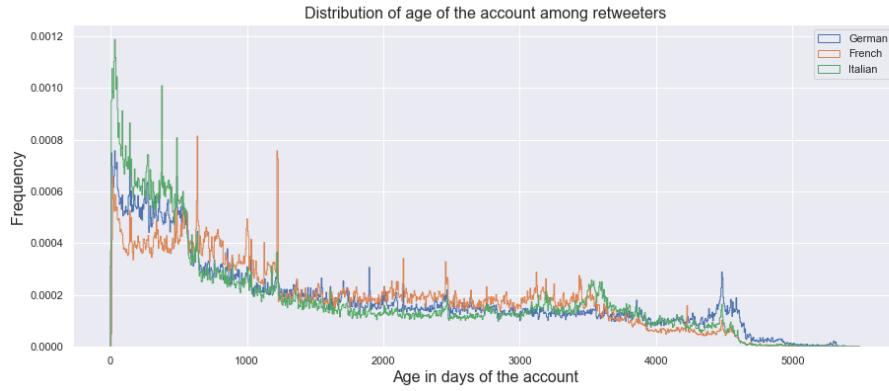


Figure 13: Distribution du temps pour retweeter sur tous les retweets

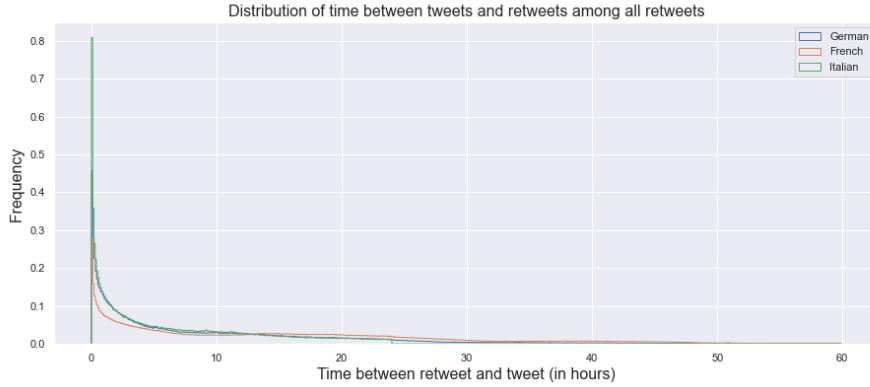


Figure	Moyenne	Ecart-Type	Min	25%	50%	75%	Max
5	32327	61339	0	3384	12187	36062	1680672
6	17.52 h	21.36	0	3.41h	12.66h	23.38h	156.40h

Rien qu'à l'inspection visuelle, deux distributions font apparaître des irrégularités manifestes : la distribution du nombre de *friends* où un pic suspect apparaît à 5000 et la distribution de l'âge des comptes, qui bien que plus irrégulières fait apparaître aussi deux pics à environ 500 et 1200.

Pour réduire déjà la dimensionnalité de ces données et comme les groupes de retweeters ne sont pas forcément de la même taille le plus simple est de discréteriser et normaliser nos données. Nous transformons donc les distributions obtenus pour chaque groupe de retweeters en les rassemblant dans un vecteur discret de taille fixe. Ce vecteur de binning est de taille et de grain différent pour chacune des caractéristiques. Le tableau ci-dessous récapitule cela :

Feature	Taille du vecteur	Granularité
Followers	100	20
Friends	300	20
Favourites	500	20
Status	600	20
Age	500	10
Retweet Time	2400	60 (1 minute)

Par exemple, si on regarde la distribution des followers d'un groupe de retweeters, on va approximer cette distribution par un vecteur de taille 100 dont la n-ième composante est le nombre de retweeters dont le nombre de followers est compris entre $n \times \text{granularité}$ - ici 20 - et $(n+1) \times \text{granularité}$. Ceux qui sont au dessus de Taille \times granularité sont comptés dans la dernière composante du vecteur.

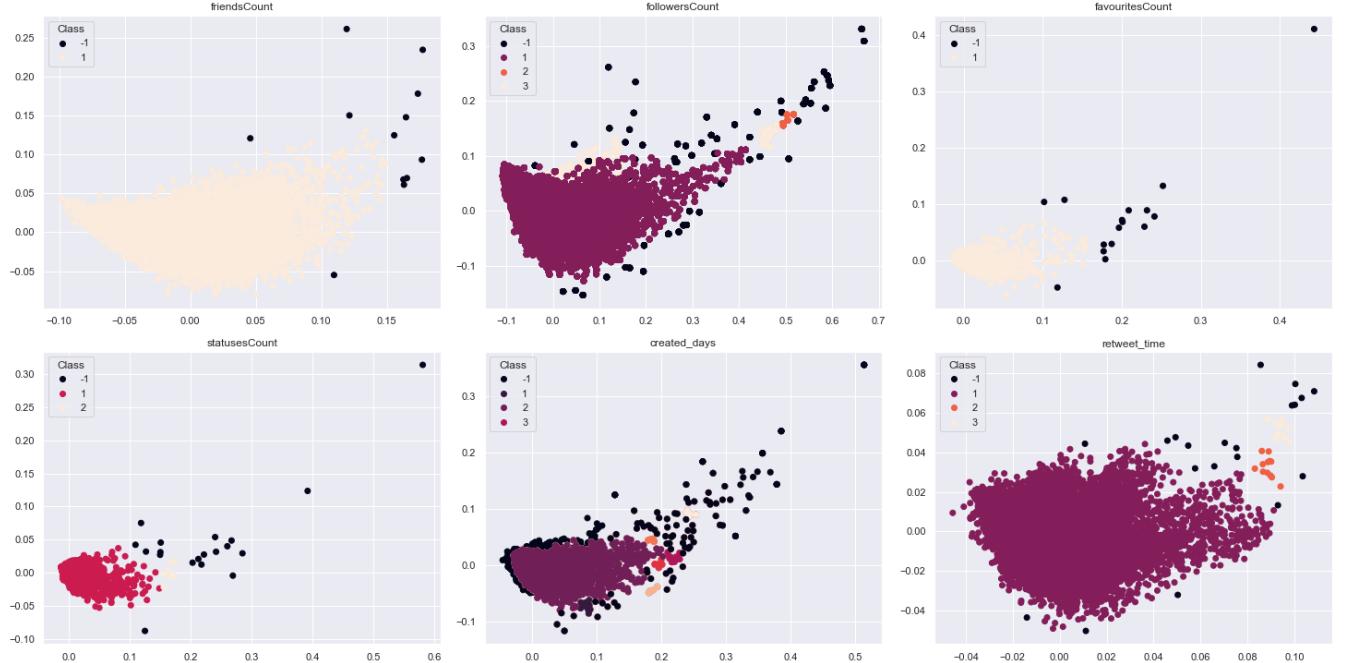
Enfin on divise le vecteur de binning obtenu par le nombre de retweeters du tweet, de sorte qu'on obtient un vecteur de fréquence.

5.3 PCA et clustering

Pour tester la pertinence de notre approche, nous essayons d'abord une méthode très simple. Nous choisissons une feature (par exemple la distribution du nombre de followers) et nous considérons les vecteurs de binning associés. On concatène ces vecteurs ce qui nous donne une matrice et on applique ensuite une PCA avec un nombre de composantes principales faibles (ici on prend 2 pour pouvoir visualiser). On applique ensuite un algorithme de clustering (ici on a choisi DBSCAN qui essaye de cluseriser les zones de forte densité ensemble).

Les graphes des résultats sont ci-dessous.

Figure 14: Résultat du processus de clusterisation



On voit que des outliers très manifestes apparaissent et que la plupart du temps on peut clairement identifier une classe centrale dans laquelle vont se loger la plupart des exemples. Il est donc possible d'observer des outliers dans les données

6 Résultats

6.1 Distance de Mahalanobis

Nous utilisons d'abord une méthode très simple pour repérer les outliers, en utilisant la forme des clusters obtenus ci-dessus.

Puisqu'il n'y a qu'une seule classe, l'idée est d'utiliser la distance de Mahalanobis comme indicateur du degré d'anormalité du point. La distance de Mahalanobis est une généralisation du z-score. Le z-score consiste à supposer pour une donnée unidimensionnelle que la distribution sous-jacente suit une loi normale. Le z-score consiste à calculer de combien de déviations standards le point est éloigné de la distribution sous-jacente. Il se calcule donc comme :

$$z = \frac{x - \mu}{\sigma}$$

où μ est la moyenne de la distribution et σ est l'écart-type.

La distance de Mahalanobis est une généralisation de cette idée pour des données multidimensionnelles. L'idée est donc encore de considérer que la distribution implicite suit une loi normale (multidimensionnelle) et de calculer le nombre de déviations standards du point étudié par rapport à cette distribution. On a donc :

$$D_M(x) = \sqrt{(x - \mu)^T \mathbf{S}^{-1} (x - \mu)}$$

où μ est le vecteur moyen de la distribution M et S est la matrice de covariance de cette distribution.

A noter que dans notre cas, nous appliquons la distance de Mahalanobis à des features issues d'une PCA donc qui sont en fait décorélées. De sorte que la matrice de covariance est diagonale et que la distance de Mahalanobis revient à la norme du vecteur des z-score selon les features de la PCA.

Nous calculons donc les distances de Mahalanobis pour chaque feature considérée et nous appliquons un seuil arbitraire (ici 12) pour repérer les points anormaux. Nous choisissons ensuite les tweets pour lesquels 2 des features sont considérés comme anormales ce qui nous fait trouver 336 tweets anormaux sur 11789. (Évidemment ce degré de sensibilité peut être facilement modifié et en particulier si différentes features nous paraissent plus importantes que d'autres).

6.2 Premiers résultats

Nous évaluons *de visu* les tweets trouvés avec la première méthode ci-dessus. Nous pouvons les répartir en plusieurs sous-groupes.

6.2.1 Jeux

Un grand nombre de tweets trouvés (plus de 50%) sont des tweets de "cagnottes à retweet" c'est à dire que le tweeter original choisit des comptes parmi les retweeters et leur donne un objet quelconque, le but étant de promouvoir commercialement des produits à faible coût.

Figure 15: Un tweet 'cagnotte'



Une inspection manuelle des retweeters montre d'ailleurs qu'un certain nombre de comptes sont en fait uniquement dédiés à retweeter ce type d'activité comme le montre les caractéristiques du compte et leurs historiques :

Figure 16: Un compte qui ne retweet que des comptes de jeux

Louh
241 Tweets

Louh
@Louh06864779

Joined October 2017

135 Following 4 Followers

Not followed by anyone you're following

Tweets Tweets & replies Media Likes

Louh Retweeted
RED by SFR (@REDbySFR) · Sep 28
[#Concours] A l'occasion de la Semaine du Développement Durable, on vous fait gagner 2 iPhone 8 reconditionnés ! Tentez de gagner l'un des deux !

Pour repartir avec : RT + Follow @REDbySFR

Fin du jeu et tirage au sort le 05/10 à 16h.

TEU CONCOURS

Le compte n'a pas de photo de profil, un grand nombre de chiffres dans son username, n'a quasiment pas de followers et son activité consiste uniquement à retweeter des cagnottes. La liste des comptes suivis est uniquement constitués de comptes qui font des concours de ce genre.

Si l'exemple n'est pas forcément très étonnant, il nous montre néanmoins que les distributions dont on pouvait s'attendre à ce qu'elles soient anormales sont bien repérées par notre méthode.

6.2.2 Promotion Commerciale

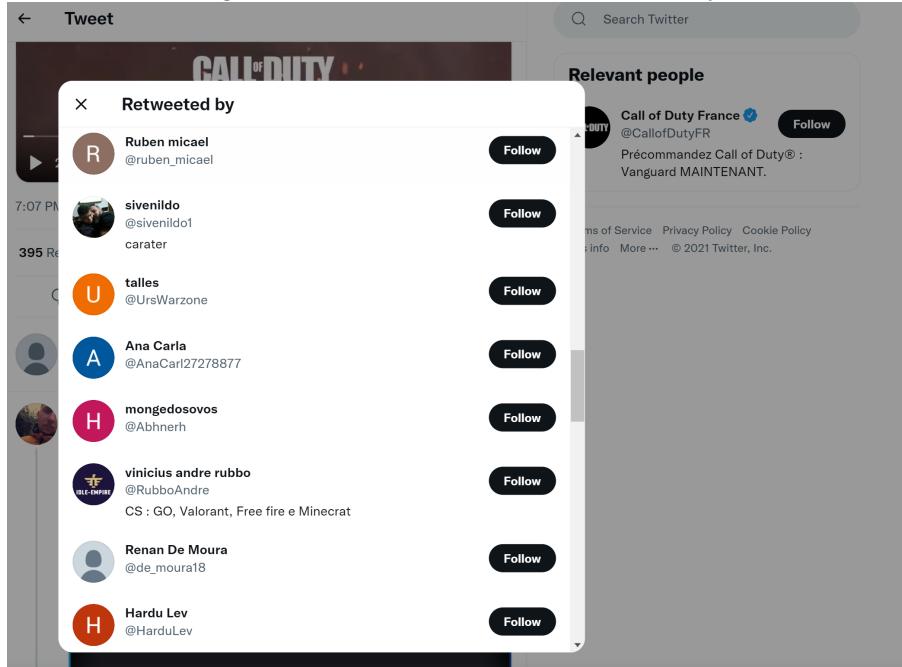
Un autre type d'exemples largement présent parmi les exemples anormaux (10%) : les tweets commerciaux qui ont manifestement été gonflés par des groupes de retweeters.

Figure 17: Tweet commercial gonflé



La liste des retweeters le montre dans ce cas de façon explicite avec des images de profils communes ; les historiques des profils confirment qu'il ne s'agit que de comptes qui promeuvent le compte principal dont est issu le tweet originel.

Figure 18: Retweeters du tweet Call Of Duty



6.2.3 Activités exotiques

Un autre type d'exemple est plus difficile à interpréter. Des tweets liés à des contenus érotiques, des NFT ou des communautés de fans apparaissent dans les tweets anormaux. Néanmoins une analyse *de visu* ne permet pas dans ce cas de repérer formellement des comptes bots.

Il semblerait de fait que ce soit plutôt des comptes très pro-actifs qui réagissent vite et de façon organisée aux contenus qui les intéressent. C'est en particulier le cas des différents comptes de fan qui à l'inspection paraissent tous être de vrais comptes mais ont des temps de retweets drastiquement plus faibles que la moyenne. On sait que ces communautés sont souvent très organisées et vont souvent auto-organiser de grandes campagnes de promotion.

Nous avons pu repérer de la sorte un ensemble de comptes indonésiens qui retweetaient sur de la k-pop de façon automatique en utilisant probablement l'API publique de twitter.

6.2.4 Activités politiques

Nous avons observé quelques activités politiques dont certaines sont manifestement coordonnées parmi les tweets étudiés. Ces tweets représentent une faible proportion des 336 tweets repérés (Moins de 20 tweets).

Nous avons en particulier observé une campagne assez massive en rapport avec la guerre du Tigré, en Ethiopie. Le tweet ci-dessous est un des exemples repérés de tweets anormaux lié à cette campagne .

Figure 19: Campagne politique autour du Tigré

The screenshot shows a tweet from the account @TigrayItaly. The tweet reads: "La nostra intervista con @ElisabettaBurba di @panorama_it in merito al ruolo di @Palazzo_Chigi, @ItalyMFA e @MinisteroDifesa nel #TigrayGenocide di @AbiyAhmedAli." Below the tweet, there is a reply from @Panorama.it: "Le relazioni pericolose del nostro governo con quello etiope. «A genocidio in corso, l'Italia si è spesso scontrata con la linea dell'Unione europea e degli Stati Uniti» denunciano gli attivisti. [panorama.it/news/dal-mondo...](#)". The tweet has 216 retweets, 12 quote tweets, and 112 likes. The timestamp is 3:44 PM - Sep 25, 2021 · Twitter Web App.

Le compte ci-dessous est un des exemples de compte qui ont retweeté cette campagne et dont l'activité est uniquement dédié à la soutenir. Cette campagne continue jusqu'aujourd'hui.

Figure 20: Un des comptes de la campagne du Tigré

The screenshot shows the Twitter profile of the user Zinash (@Zinash75277525). The profile picture is a dark blue silhouette of a person's head. The bio reads "my kids". The account was joined in January 2021 and has 34 Following and 39 Followers. The "Tweets" tab is selected, showing a single tweet from Kiros Araya (@Hara_Meret) dated Oct 12, 2021, which reads: "Breaking: @SenAlexPadilla does the right thing - declines to show up at pro #TigrayGenocide CA gathering that has been advertised as a Town Hall meeting. #TigrayGenocide #AllowAccessToTigray".

Figure 21: Activité du compte qui retweete plusieurs fois des contenus similaires pour essayer de les diffuser



Nous avons également repéré des tweets soutenus par des comptes qui sont explicitement des supporters politiques de partis, en particulier en Allemagne.

Ces résultats nous montrent que l'algorithme parvient à repérer au moins certains cas de campagnes d'astroturfing. Même si les exemples repérés n'étaient pas forcément très subtils, les mêmes méthodes devraient aussi fonctionner pour les campagnes politiques qui s'affichent moins.

Il faut néanmoins remarquer que ce type d'activité n'a pas l'air extrêmement courant, en réalité même marginal et que les impacts concrets sont encore plus incertains.

6.3 Méthodes d'identification d'outliers

Pour la suite nous utilisons le package *pyod*[43] en python qui implémente un certain nombre d'algorithmes. Nous décrivons ci-dessous les algorithmes que nous utilisons ainsi que leurs principes.

Pour pouvoir appliquer ces algorithmes à nos données, nous utilisons deux techniques de réduction de dimensionnalité. Nous concaténons les vecteurs selon les différentes features ce qui nous donne un vecteur unique, contenant toutes les features, pour un groupe de retweeter. Nous appliquons sur ce vecteur une PCA (avec 32 composantes) et un Autoencoder Dense dont la taille de la représentation latente est aussi 32. Les représentations apprises ne sont pas très distinctes l'une de l'autre, par la suite nous utilisons les algorithmes choisis sur la représentation de l'Autoencoder.

Article	Auteurs	Présentation de l'article
LOF: Identifying Density-Based Local Outliers[6]	Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, Jörg Sander	L'idée de la méthode LOF (Local Outlier Factor) est d'essayer de capturer une notion de distance locale. Au lieu d'utiliser une distance euclidienne classique, la méthode LOF utilise la notion de k-distance qui est la distance minimale pour atteindre aux moins k autres objets du dataset utilisé. On définit ensuite l'accessibilité d'un point comme étant le max entre cette distance et la distance euclidienne standard. Intuitivement ceci permet de pénaliser les points outliers puisque leur k-distance sera grande même pour des points très proches. On se sert ensuite de l'accessibilité du point pour calculer une <i>densité locale</i> du point : l'idée est de moyenner l'accessibilité sur un nombre N de points et de prendre ensuite l'inverse. On effectue cette opération pour tous les points puis on peut enfin calculer le LOF du point : c'est le ratio moyen de la densité locale des N points du voisinage par rapport à celui du point étudié. En d'autres termes les points de LOF grand sont ceux pour lesquels la densité du point est faible par rapport à la densité des points environnants. L'article procède ensuite à montrer certaines propriétés de cette distance qui garantisse en partie qu'elle parvient bien à trouver des outliers dans le dataset étudié.
Discovering cluster-based local outliers[22]	Zengyou He, Xiaofei Xu, Shengchun Deng	L'idée de l'article est de définir un facteur d'outlier à partir du résultat d'un clustering. Une fois qu'on s'est donné une clusterisation de notre dataset, on sépare les différents clusters entre ceux qui sont petits et ceux qui sont gros ; pour cela on les ordonne et on se donne deux paramètres α et β . Deux cas de figure peuvent se produire : soit le ratio de taille de deux clusters consécutifs est plus grande que β - dans ce cas, l'indice où cet événement se produit sert de séparateur entre les clusters gros et les clusters petits ; soit l'union des clusters consécutifs atteint au moins $\alpha\%$ du dataset ; dans ce cas l'indice où l'union des clusters atteint ce pourcentage sert de séparateur. L'idée est de garantir à la fois que les clusters larges sont bien plus gros que ceux qui sont petits et si ça n'est pas le cas, qu'il représente au moins une grande partie des données. Le CBLOF (<i>Cluster-based local outlier factor</i>) vaut alors la distance au cluster large si le point étudié est dans un cluster large et la distance au cluster le plus proche si le point étudié est dans un petit cluster. L'idée est là encore de faire en sorte que les données qui sont accidentellement proches les unes des autres ait un facteur d'outlier néanmoins grand. Le reste de l'article montre un algorithme qui permet de clusteriser les données en respectant ces propriétés et qui permet finalement de donner un score d'outlier à tous les points.

Article	Auteurs	Présentation de l'article
Histogram-based Outlier Score (HBOS): A fast Unsupervised Anomaly Detection Algorithm [20]	Markus Goldstein, Andreas Dengel	<p>L'idée de l'article est d'utiliser une répartition des données sous la forme d'histogrammes. Pour chaque feature, on construit un histogramme classiquement où les données sont répartis entre k (on choisit le plus souvent $k = \sqrt{N}$ où N est le nombre d'échantillons) bins en fonction de la valeur qu'elles ont pour cette feature. On aboutit ainsi à un histogramme qui représente la fonction de densité du dataset pour chaque feature. Le score donnée par l'algorithme est alors simplement :</p> $HBOS(p) = \sum_{i=0}^d \log\left(\frac{1}{\text{hist}_i(p)}\right)$ <p>où hist_i est la hauteur du bin dans laquelle le point p est localisé. L'idée est donc de caractériser à quel point se trouve en dehors des distributions de feature comparés au reste. Notons que l'algorithme n'est pertinent que si les données sont découplées. Il a aussi l'avantage d'être rapide (temps linéaire en fonction des données).</p>
Isolation Forest [32]	Fei Liu, Tony Kai Ming Ting, Zhi-Hua Zhou	<p>La plupart des approches existantes basées sur le modèle pour la détection des anomalies construisent un profil d'instances normales, puis identifient les instances qui ne sont pas conformes au profil normal comme des anomalies. Cet article propose une méthode basée sur un modèle fondamentalement différent qui isole explicitement les anomalies à la place des profils de points normaux. L'idée est de construire des arbres qui fonctionnent de la manière suivante : on choisit au hasard une des features du dataset et on sélectionne ensuite au hasard une valeur pivot de cette feature qui permet de diviser le dataset en 2. On recommence récursivement ce processus avec les 2 sous-datasets jusqu'à atteindre une hauteur d'arbre limite n ou que le dataset à ne comporter plus qu'un élément. On réitère le processus pour obtenir une forêt d'arbres de la sorte. Pour calculer le score d'anomalie d'un point on le fait passer à travers tous les arbres de la forêt et on note à quelle hauteur de l'arbre il s'arrête. On moyenne ensuite cette hauteur pour toute la forêt et on normalise le score en écrivant :</p> $s(x, n) = 2^{-\frac{\hat{h}(x)}{c_n}}$ <p>où \hat{h} est la moyenne de la hauteur et c_n est un coefficient de normalisation lié à la hauteur moyenne d'un chemin dans un arbre de recherche binaire. L'idée est que si un point est très anormal, il devrait rapidement pouvoir être isolé parmi les autres points et donc avoir un score d'anomalie plus faible. Un des avantages conséquent de la méthode est qu'elle est peu coûteuse en temps.</p>

Article	Auteurs	Présentation de l'article
COPOD: Copula-Based Outlier Detection [30]	Zheng Li, Yue Zhao, Nicola Botta, Cezar Ionescu, Xiyang Hu	<p>Pour y remédier, nous présentons un nouvel algorithme de détection des aberrations appelé COPOD en modélisant la distribution des données multivariées. L'idée est d'abord de calculer la fonction de répartition empiriques des données pour chaque feature :</p> $\hat{F}_j(x) = \frac{1}{n} \sum_{i=1}^n \mathbf{I}(X_{i,j} < x)$ <p>où n est le nombre d'échantillons et $X_{i,j}$ est la valeur de la feature j de l'échantillon X_i. Puis on se donne des probabilités d'observations de chaque feature pour un échantillon X_i donné :</p> $(U_{i,1}, \dots, U_{i,d}) = (\hat{F}_1(X_{i,1}), \dots, \hat{F}_d(X_{i,d}))$ <p>Ce qui nous permet finalement de donner un score à l'échantillon étudié sous la forme de :</p> $C((u_1, \dots, u_d)) = P(F_1(X_1) \leq u_1, \dots, F_d(X_d) \leq u_d)$ $= \frac{1}{n} \sum_{i=1}^n \mathbf{I}(U_{i,1} < u_1, \dots, U_{i,d} < u_d)$ <p>L'article continue en résolvant un certain nombre de problèmes théoriques de sorte que le score vraiment calculé n'est pas aussi simple que celui que nous avons mis ci-dessus. Mais l'idée est bien de caractériser une distribution implicite et de voir la distance du point à cette distribution.</p>
Efficient Algorithms for Mining Outliers from Large Data Sets	Ramaswamy, Sridhar and Rastogi, Rajeev and Shim, Kyuseok	<p>Enfin une approche très simple consiste à évaluer le degré d'anormalité du point en calculant sa distance k-NN avec les autres points. L'idée est similaire à [6] : on prend la distance minimale pour atteindre au moins k points du dataset. On peut ensuite sélectionner un ensemble de points autour du point pour calculer une distance k-NN moyenne ou une distance k-NN médiane. L'article montre que cette distance est à la fois rapide à calculer et pertinente pour les problèmes posés. Dans nos expériences cette distance fonctionne bien mais souvent moins que les autres.</p>

D'autres approches sont plus adaptées à des données géométriques : elles reposent par exemple sur des calculs d'angle en trois ou plus de dimensions [28] [2]. Ces approches n'ont pas présenté de bons résultats (ce qui était prévisible) et nous les avons donc exclues.

Encore une autre catégorie d'approches reposent sur des termes de reconstructions d'erreur du terme lorsqu'on fait passer les données à travers un autoencoder ou des architectures variantes (VAE, gamma VAE) [26] [7]. Il nous a semblé que la validation théorique de ces algorithmes étaient plus faibles et les résultats que nous avons obtenus avec étaient également moins bons.

6.4 Évaluation

Une difficulté de l'évaluation de nos résultats est que nous n'avons bien sûr pas accès à une *ground truth*. Nous ne pouvons donc pas utiliser les notions classiques d'évaluation des résultats (FC, AUC, etc...). Il existe de fait de la littérature et des résultats sur la qualité d'un clustering en l'absence de *ground truth* (silhouette score, Density-Based Clustering Validation) mais les résultats sur la qualité de la détection d'anomalies sont beaucoup plus rares. Même une revue de littérature comme [44] n'a pas connaissance des résultats permettant d'évaluer la détection d'anomalies *a*

A noter que nous aurions pu aussi essayer d'injecter dans nos datasets des outliers artificiels et vérifier que notre modèle permet bien de les repérer. Dans notre cas il est néanmoins difficile de générer des outliers artificiels pertinents, parce que nous cherchons justement à repérer n'importe quels types d'outlier..

Nous décidons donc d'utiliser des critères d'évaluation d'outlier qui n'utilisent pas d'états de *ground truth*. Il semblerait qu'il n'en existe que deux dans la littérature. Nous les présentons ci-dessous :

6.4.1 Mass volume curves and anomaly ranking - Stephan Cléménçon and Albert Thomas [11]

L'idée de ce papier est de réussir à classifier des fonctions de score (ici l'outlier score) pour un certain ensemble de données X . On modélise cette ensemble comme une distribution et on cherche à caractériser les lignes de niveaux pour une certaine fonction de score :

$$\Omega_{s,t} = \{x \in X | s(x) \geq t\}$$

où μ est la mesure de Lebesgue. On introduit également le volume de cette ligne de niveau :

$$\mu_s(t) = \mu(\Omega_{s,t})$$

ainsi que la masse :

$$\alpha_s(t) = \mathcal{P}(s(X) \geq t)$$

L'idée est de caractériser les outliers comme ceux qui appartiennent au complémentaire des lignes de niveaux de volume minimal, c'est à dire celles qui sont dans :

$$\Omega_\alpha^* = \mu(\Omega) \text{ avec } P(X \in \Omega) \geq \alpha$$

Les outliers se situent par définition dans le complémentaire de cet ensemble pour de grandes valeurs de α . Avec les outils introduits, nous pouvons alors définir la courbe de Masse-Volume :

$$MV_s(\alpha) = \lambda_s \circ \alpha_s^{-1}(\alpha)$$

Et cette courbe définit ensuite un ordre partiel entre deux fonctions de score :

$$\forall \alpha MV_s(\alpha) \leq MV_{s^*}(\alpha) \iff s \leq s^*$$

On montre ensuite que la courbe est minimale pour la fonction de score qui vaut la densité f de la distribution observée. On montre par ailleurs un résultat assez technique qui permet de borner la différence entre la courbe de masse volume optimale et celle étudiée :

$$MV_s(\alpha) - MV_f(\alpha) \leq \mu(\Omega_\alpha^* \Delta \Omega_{s,Q(s,\alpha)})$$

où $Q(s, \alpha)$ est le quantile de niveau $1 - \alpha$ de la distribution $s(X)$ où X est de densité f : $Q(s, \alpha) = F_s^{-1}(1 - \alpha)$ et Δ est la différence symétrique. Ce résultat nous montre que si la fonction s traduit

bien la distance à la distribution f considérée, alors la courbe MV associée sera proche de la courbe optimale.

Figure 22: Courbe de Mass Volume pour une gaussienne

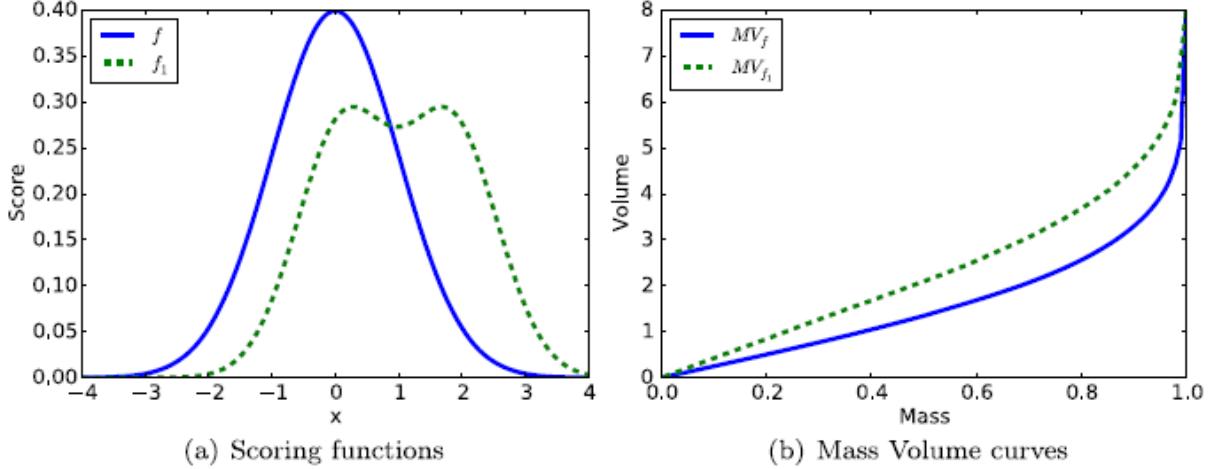


FIG 1. *Scoring functions and their associated Mass Volume curves when considering a truncated Gaussian distribution with density f .*

6.4.2 On Anomaly Ranking and Excess-Mass Curves – Nicolas Goix, Anne Sabourin, Stéphane Cléménçon [36]

L’objectif de cet article est de proposer une manière d’évaluer la pertinence d’un scoring $s : x \rightarrow s(x)$ via la construction d’un critère $C(s)$ quantifiable. On s’intéresse à une formulation lagrangienne de la méthode MV décrite précédemment [11], c’est à dire que on est face à un problème d’optimisation de la quantité suivante pour $t > 0$:

$$\{\mathcal{P}(X \in \Omega) - tLeb(\Omega)\}$$

où $Leb(\Omega)$ est la mesure de Lebesgue de Ω . On définit alors EM^* l’excess mass curve optimale par :

$$EM^* : t \in [0, +\infty[\longrightarrow \sup_{A \text{ borelien}} \mathcal{P}(X \in A) - tLeb(A)$$

On étudie aussi une transformation de s appellée excess mass curve de s définie par :

$$EM_s : t \in [0, +\infty[\longrightarrow \sup_{A \in \{\Omega_{s,l}, l > 0\}} \mathcal{P}(X \in A) - tLeb(A)$$

où $\{\Omega_{s,l}, l > 0\}$ est l’ensemble des lignes de niveau de s définies par $\Omega_{s,l} = \{x, s(x) \geq l\}$

Le critère $C(s)$ est construit d’une manière itérative à partir de l’écart entre ces fonctions

$$|EM^*(t) - EM_s(t)|$$

6.4.3 Résultats de l'évaluation

Enfin nous réalisons un tableau comprenant les algorithmes et leur score selon ces deux critères de la littérature. D'après [19], ces critères classifient les méthodes de détection la plupart du temps dans le même ordre que si l'on suit les critères traditionnels (courbe ROC ou courbe PR).

On voit donc que les méthodes d'Isolation Forest et de HBOS ont les meilleurs résultats. En regardant les outliers trouvés avec ces deux méthodes (121 et 165), 90 exemples se retrouvent dans les deux listes ce qui représente un bon taux d'intersection, sachant qu'en faisant jouer les paramètres on peut s'arranger pour que les méthodes renvoient le même nombre d'outliers. Sur les 197 outliers trouvés entre les deux méthodes 123 (62%) se retrouvent dans les 336 outliers trouvés plus haut. Ceci montre à la fois que les méthodes parviennent à s'accorder entre elles mais qu'elles permettent aussi de retrouver d'autres types d'outliers ignorés par les algorithmes.

Une inspection visuelle rapide montre que les outliers repérés avec les techniques ci-dessus semblent également valides, la plupart étant clairement des retweets modifiés.

Table 1: Tableau des méthodes et des métriques associés (EM et MV) gras la méthode la meilleure selon la métrique et en souligné la pire

Méthode	EM	MV
LOF	2.7e-05	3.4e02
CBLOF	<u>1.2e-05</u>	<u>4.1e02</u>
HBOS	6.7e-05	7.2e01
IForest	2.1e-04	8.3e01
COPOD	1.8e-05	2.8e02
k-NN (Avg, k=15)	5.4e-05	1.7e02

6.5 Avantages

1. Notre approche est entièrement non-supervisée : nous évitons ainsi tous les problèmes de ce domaine liés à l'absence de ground truth et à l'absence de données labellisées.
2. Contrairement à la grande majorité des autres méthodes de la littérature nous nous concentrons uniquement sur des contenus qui ont eu du succès (nous avons choisi arbitrairement 100 retweets au minimum). Ceci nous évite d'avoir à trier parmi des contenus peu intéressants ou qui n'ont pas eu d'impacts. Par ailleurs nous réduisons ainsi considérablement le volume de données à traiter.
3. Un des intérêts des méthodes utilisées est que le degré de sensibilité à l'anormalité est facilement modifiable. On peut facilement le modifier pour atteindre une part plus grande des outliers, tant en fait qu'on attend encore du contenu qui paraît suspect après inspection.

7 Conclusion

Nous avons proposé la première - à notre connaissance - méthode de découverte d'activité suspecte sur les réseaux sociaux qui s'appuient sur une classification de groupes de comptes.

Cette méthode a présenté de bons résultats, en particulier elle semble adéquate pour repérer toutes sortes de manipulation, au moins les plus élémentaires. Notre méthode semble montrer que les contenus politiques inflatés sont d'ailleurs relativement rares en proportion d'autres types de contenus.

Un des avantages de la méthode est que la collecte de données et le traitement sont en fait assez rapides de sorte qu'il est possible de traiter de gros volumes de données rapidement. Par ailleurs, l'absence de label utilisé fait que les résultats devraient être facilement améliorables en rajoutant simplement des données.

7.1 Limites et améliorations

7.1.1 Limites

Bien que la possibilité de contourner cette méthode est plus limité que pour d'autres techniques (en particulier celles liés à la synchronicité), il reste néanmoins possible de créer des réseaux de comptes bots ou humains dont les caractéristiques étudiées ici (distribution du nombre d'amis, de la date de création du compte, etc...) seraient similaires aux groupes de retweeters normaux.

Par ailleurs, nous avons vu dans l'exploration des résultats qu'il n'était pas forcément facile de distinguer des réseaux parfaitement organisés d'autres types d'organisation (groupes de fans, communautés d'artistes).

7.1.2 Clusteriser

Nous nous sommes volontairement limités à de petits ensembles de données pour pouvoir garder quelque chose de manipulable dans le cadre de ce travail. Mais en pratique les méthodes utilisées ici gagneraient énormément en utilisant des volumes de données plus importants.

En particulier, la typologie de résultat que nous avons relevés dans l'analyse des résultats pourrait se retrouver dans une analyse de cluster plus fine, qui nécessite néanmoins plus de comptes, qui ferait apparaître les caractéristiques par exemple des retweeters de tweet à cagnotte etc... Ceci permettrait de se concentrer sur les vrais outliers ou de repérer directement un cluster spécifique, par exemple, s'il existe, de tweets politiques inflatés.

Une action simple d'ailleurs pour réduire l'impact de ces comptes est tout simplement de les enlever des tweets analysés ce qui nous permettrait normalement de repérer plus finalement les anomalies dans le reste des tweets analysés.

7.1.3 Revenir aux comptes

A partir des tweets repérés comme étant anormaux, une idée intéressante est de stocker les retweeters de ces tweets anormaux et de vérifier s'ils se retrouvent dans d'autres tweets suspicieux. Cette technique permettrait potentiellement de repérer des réseaux de bots, notamment en prenant en compte les taux de co-retweets entre les différents retweeters.

7.1.4 Autres features

Parmi les features intéressantes à observer, on pourrait aussi rajouter l'heure où a été postée le retweet ce qui devrait pouvoir montrer des tentatives de diffusion externes si les pays ne sont pas sur les mêmes fuseaux horaires

7.1.5 Focalisation sur la politique

L'algorithme pourrait plus facilement s'adapter à la politique : il suffirait pour cela de ne considérer que les tweets contenant un certain type de hashtag/de langages. Le désavantage opérationnel de notre méthode c'est qu'il y a beaucoup de bruits venant d'autres catégories de tweets (en particuliers celles vue ci-dessus)

Par ailleurs, se cantonner uniquement à la sphère politique devrait permettre de calibrer encore un peu plus le type de tweets étudiés et donc de plus facilement repérer les anomalies par rapport aux autres tweets.

7.1.6 Ensembling

Combiner différents détecteurs d'outlier a normalement son intérêt[45] mais ici vu que les critères retenus pour évaluer l'efficacité ne sont pas absolument certains, il n'était pas facile de savoir quelle méthode d'ensembling favoriser, etc...

Après quelques tests il nous semble quand même important de bien sélectionner les détecteurs : certains ne renvoient pas des outliers forcément pertinents et leurs résultats biaisent les résultats d'ensemble. Nous avons indiqué ci-dessus quels détecteurs nous semblaient les meilleurs.

List of Figures

1	Etude sur les différents type de détecteurs de bot existants [Source : [12]]	6
2	Nombre d'articles publiés sur la détection de bot sur Twitter [Source : [37]]	7
3	Graphe avant/après avoir enlever les comptes malicieux	9
4	Illustration de la distance DTW entre deux séries temporelles [Source : [25]]	10
5	Schéma du modèle utilisé par RTBust [Source : [34]]	11
6	Visualisation des temps de retweets pour un utilisateur donné [Source : [34]]	12
7	Comparaison de l'évolution de la longueur de la LCS en fonction du nombre de comptes observés (Source : [13])	13
8	Distribution du nombre de followers sur tous les retweeters	17
9	Distribution du nombre de friends (=followed) sur tous les retweeters	17
10	Distribution du nombre de like sur tous les retweeters	18
11	Distribution du nombre de status sur tous les retweeters	18
12	Distribution de l'age des comptes de tous les retweeters	19
13	Distribution du temps pour retweeter sur tous les retweets	19
14	Résultat du processus de clusterisation	20
15	Un tweet 'cagnotte'	22
16	Un compte qui ne retweet que des comptes de jeux	22
17	Tweet commercial gonflé	23
18	Retweeters du tweet Call Of Duty	24
19	Campagne politique autour du Tigré	25
20	Un des comptes de la campagne du Tigré	25
21	Activité du compte qui retweete plusieurs fois des contenus similaires pour essayer de les diffuser	26
22	Courbe de Mass Volume pour une gaussienne	31

References

- [1] Kayode Sakariyah Adewole et al. "Twitter spam account detection based on clustering and classification methods". In: *The Journal of Supercomputing* 76 (2018), pp. 4802–4837.
- [2] Yahya Almardeny, Noureddine Boujnah, and Frances Grant. "A Novel Outlier Detection Method for Multivariate Data". In: *IEEE Transactions on Knowledge and Data Engineering* PP (Nov. 2020), pp. 1–1. DOI: 10.1109/TKDE.2020.3036524.
- [3] Mansour Alsaleh et al. "TSD: Detecting Sybil Accounts in Twitter". In: *2014 13th International Conference on Machine Learning and Applications*. 2014, pp. 463–469. DOI: 10.1109/ICMLA.2014.81.
- [4] Michael Arnold and Enno Ohlebusch. "Linear Time Algorithms for Generalizations of the Longest Common Substring Problem". In: *Algorithmica* 60 (Aug. 2011), pp. 806–818. DOI: 10.1007/s00453-009-9369-1.
- [5] Christopher A. Bail et al. "Assessing the Russian Internet Research Agency's impact on the political attitudes and behaviors of American Twitter users in late 2017". In: *Proceedings of the National Academy of Sciences* 117.1 (2020), pp. 243–250. ISSN: 0027-8424. DOI: 10.1073/pnas.1906420116. eprint: <https://www.pnas.org/content/117/1/243.full.pdf>. URL: <https://www.pnas.org/content/117/1/243>.
- [6] Markus M. Breunig et al. "LOF: Identifying Density-Based Local Outliers". In: *SIGMOD Rec.* 29.2 (May 2000), 93–104. ISSN: 0163-5808. DOI: 10.1145/335191.335388. URL: <https://doi.org/10.1145/335191.335388>.

- [7] Christopher P. Burgess et al. *Understanding disentangling in β -VAE*. 2018. arXiv: 1804.03599 [stat.ML].
- [8] Qiang Cao et al. “Uncovering Large Groups of Active Malicious Accounts in Online Social Networks”. In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* (2014).
- [9] Paul Charon and Jean-Baptiste Jeangène Vilmer. *Les opérations d'influence chinoise*. irsem, 2021.
- [10] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. “DeBot: Twitter Bot Detection via Warped Correlation”. In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. 2016, pp. 817–822. DOI: 10.1109/ICDM.2016.0096.
- [11] Stephan Cléménçon and Albert Thomas. “Mass volume curves and anomaly ranking”. In: *Electronic Journal of Statistics* 12.2 (2018), pp. 2806 –2872. DOI: 10.1214/18-EJS1474. URL: <https://doi.org/10.1214/18-EJS1474>.
- [12] Stefano Cresci. “A decade of social bot detection”. In: *Communications of the ACM* 63.10 (2020), 72–83. ISSN: 1557-7317. DOI: 10.1145/3409116. URL: <http://dx.doi.org/10.1145/3409116>.
- [13] Stefano Cresci et al. “Social fingerprinting: detection of spambot groups through DNA-inspired behavioral modeling”. In: *IEEE Transactions on Dependable and Secure Computing* (2017), 1–1. ISSN: 1545-5971. DOI: 10.1109/tdsc.2017.2681672. URL: <http://dx.doi.org/10.1109/TDSC.2017.2681672>.
- [14] Clayton Allen Davis et al. “BotOrNot”. In: *Proceedings of the 25th International Conference Companion on World Wide Web - WWW '16 Companion* (2016). DOI: 10.1145/2872518.2889302. URL: <http://dx.doi.org/10.1145/2872518.2889302>.
- [15] Juan Echeverri-López et al. “LOBO: Evaluation of Generalization Deficiencies in Twitter Bot Classifiers”. In: *Proceedings of the 34th Annual Computer Security Applications Conference*. ACSAC ’18. San Juan, PR, USA: Association for Computing Machinery, 2018, 137–146. ISBN: 9781450365697. DOI: 10.1145/3274694.3274738. URL: <https://doi.org/10.1145/3274694.3274738>.
- [16] Mohd Fazil and Muhammad Abulaish. “A Hybrid Approach for Detecting Automated Spammers in Twitter”. In: *IEEE Transactions on Information Forensics and Security* 13.11 (2018), pp. 2707–2719. DOI: 10.1109/TIFS.2018.2825958.
- [17] Emilio Ferrara et al. “The rise of social bots”. In: *Communications of the ACM* 59 (2016), pp. 96 –104.
- [18] Noé Gaumont, Maziyar Panahi, and David Chavalarias. “Reconstruction of the socio-semantic dynamics of political activist Twitter networks—Method and application to the 2017 French presidential election”. In: *PLOS ONE* 13.9 (Sept. 2018), pp. 1–38. DOI: 10.1371/journal.pone.0201879. URL: <https://doi.org/10.1371/journal.pone.0201879>.
- [19] Nicolas Goix. *How to Evaluate the Quality of Unsupervised Anomaly Detection Algorithms?* 2016. arXiv: 1607.01152 [stat.ML].
- [20] Markus Goldstein and Andreas R. Dengel. “Histogram-based Outlier Score (HBOS): A fast Unsupervised Anomaly Detection Algorithm”. In: 2012.
- [21] Robert Gorwa and Douglas Guilbeault. “Unpacking the Social Media Bot: A Typology to Guide Research and Policy”. In: *Policy & Internet* 12.2 (2020), pp. 225–248. DOI: <https://doi.org/10.1002/poi3.184>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/poi3.184>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/poi3.184>.

- [22] Zengyou He, Xiaofei Xu, and Shengchun Deng. “Discovering Cluster-Based Local Outliers”. In: *Pattern Recogn. Lett.* 24.9–10 (June 2003), 1641–1650. ISSN: 0167-8655. DOI: 10.1016/S0167-8655(03)00003-5. URL: [https://doi.org/10.1016/S0167-8655\(03\)00003-5](https://doi.org/10.1016/S0167-8655(03)00003-5).
- [23] Meng Jiang et al. “Inferring Strange Behavior from Connectivity Pattern in Social Networks”. In: vol. 8443. May 2014. DOI: 10.1007/978-3-319-06608-0_11.
- [24] Franziska B. Keller et al. “Political Astroturfing on Twitter: How to Coordinate a Disinformation Campaign”. In: *Political Communication* 37.2 (2020), pp. 256–280. DOI: 10.1080/10584609.2019.1661888. eprint: <https://doi.org/10.1080/10584609.2019.1661888>. URL: <https://doi.org/10.1080/10584609.2019.1661888>.
- [25] Eamonn Keogh and Chotirat Ratanamahatana. “Exact indexing of dynamic time warping”. In: *Knowledge and Information Systems* 7 (Jan. 2005), pp. 358–386. DOI: 10.1007/s10115-004-0154-9.
- [26] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2014. arXiv: 1312.6114 [stat.ML].
- [27] Jon M. Kleinberg. “Authoritative Sources in a Hyperlinked Environment”. In: *J. ACM* 46.5 (Sept. 1999), 604–632. ISSN: 0004-5411. DOI: 10.1145/324133.324140. URL: <https://doi.org/10.1145/324133.324140>.
- [28] Hans-Peter Kriegel, Matthias Schubert, and Arthur Zimek. “Angle-Based Outlier Detection in High-Dimensional Data”. In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’08. Las Vegas, Nevada, USA: Association for Computing Machinery, 2008, 444–452. ISBN: 9781605581934. DOI: 10.1145/1401890.1401946. URL: <https://doi.org/10.1145/1401890.1401946>.
- [29] Sneha Kudugunta and Emilio Ferrara. “Deep neural networks for bot detection”. In: *Information Sciences* 467 (2018), 312–322. ISSN: 0020-0255. DOI: 10.1016/j.ins.2018.08.019. URL: <http://dx.doi.org/10.1016/j.ins.2018.08.019>.
- [30] Zheng Li et al. *COPOD: Copula-Based Outlier Detection*. 2020. arXiv: 2009.09463 [stat.ML].
- [31] Qiao Liang and Wang Xiangsui. *Unrestricted Warfare: Two Air Force Senior Colonels on Scenarios for War and the Operational Art in an Era of Globalization*. Beijing: PLA Literature and Arts Publishing House, 1999.
- [32] Fei Tony Liu, Kai Ting, and Zhi-Hua Zhou. “Isolation Forest”. In: Jan. 2009, pp. 413 –422. DOI: 10.1109/ICDM.2008.17.
- [33] Franziska Martini et al. “Bot, or not? Comparing three methods for detecting social bots in five political discourses”. In: *Big Data & Society* 8.2 (2021), p. 20539517211033566. DOI: 10.1177/20539517211033566. eprint: <https://doi.org/10.1177/20539517211033566>. URL: <https://doi.org/10.1177/20539517211033566>.
- [34] Michele Mazza et al. “RTbust: Exploiting Temporal Patterns for Botnet Detection on Twitter”. In: *Proceedings of the 10th ACM Conference on Web Science*. WebSci ’19. Boston, Massachusetts, USA: Association for Computing Machinery, 2019, 183–192. ISBN: 9781450362023. DOI: 10.1145/3292522.3326015. URL: <https://doi.org/10.1145/3292522.3326015>.
- [35] Nazanin Mehrasa et al. *A Variational Auto-Encoder Model for Stochastic Point Processes*. 2019. arXiv: 1904.03273 [cs.CV].
- [36] Stéphane Cléménçon Nicolas Goix Anne Sabourin. In: () .
- [37] Luis Daniel Samper Escalante et al. “Bot Datasets on Twitter: Analysis and Challenges”. In: *Applied Sciences* 11 (Apr. 2021), p. 4105. DOI: 10.3390/app11094105.
- [38] V.S. Subrahmanian et al. “The DARPA Twitter Bot Challenge”. In: *Computer* 49.6 (2016), pp. 38–46. ISSN: 1558-0814. DOI: 10.1109/MC.2016.183.

- [39] Twitter Inc. *Rate limits*. <https://developer.twitter.com/en/docs/twitter-api/rate-limits>. Online; accessed 07 September 2021. 2021.
- [40] Onur Varol et al. “Online Human-Bot Interactions: Detection, Estimation, and Characterization”. In: *CoRR* abs/1703.03107 (2017). arXiv: 1703.03107. URL: <http://arxiv.org/abs/1703.03107>.
- [41] Onur Varol et al. “Online Human-Bot Interactions: Detection, Estimation, and Characterization”. In: *ArXiv* abs/1703.03107 (2017).
- [42] Kai-Cheng Yang et al. “Scalable and Generalizable Social Bot Detection through Data Selection”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.01 (2020), 1096–1103. ISSN: 2159-5399. DOI: 10.1609/aaai.v34i01.5460. URL: <http://dx.doi.org/10.1609/aaai.v34i01.5460>.
- [43] Yue Zhao, Zain Nasrullah, and Zheng Li. “PyOD: A Python Toolbox for Scalable Outlier Detection”. In: *Journal of Machine Learning Research* 20.96 (2019), pp. 1–7. URL: <http://jmlr.org/papers/v20/19-011.html>.
- [44] Arthur Zimek, Ricardo J.G.B. Campello, and Jörg Sander. “Ensembles for Unsupervised Outlier Detection: Challenges and Research Questions a Position Paper”. In: *SIGKDD Explor. Newsl.* 15.1 (Mar. 2014), 11–22. ISSN: 1931-0145. DOI: 10.1145/2594473.2594476. URL: <https://doi.org/10.1145/2594473.2594476>.
- [45] Arthur Zimek, Ricardo J.G.B. Campello, and Jörg Sander. “Ensembles for Unsupervised Outlier Detection: Challenges and Research Questions a Position Paper”. In: *SIGKDD Explor. Newsl.* 15.1 (Mar. 2014), 11–22. ISSN: 1931-0145. DOI: 10.1145/2594473.2594476. URL: <https://doi.org/10.1145/2594473.2594476>.