Reconnaissance d'auteurs d'oeuvres littéraires à partir du graphe d'interactions de leurs personnages

Simon Delarue
Télécom Paris - France

Novembre 2020

1 Introduction

Le problème de détection automatique d'auteur peut avoir plusieurs applications dans des domaines variés ; investigation judiciaire, détection de plagiat, détection de bots (notamment sur les réseaux sociaux) ou encore dans la littérature. Dans ce dernier cas, les travaux de recherche menés ces dernières années ont soulevé des questions intéressantes ; dans un article paru en 2015 [1], Boyd et al mettent en évidence le style d'écriture d'un second auteur dans une des pièces attribuée à Shakespeare. Plus récemment, en 2017 [2], Jacques Savoy révèle des similarités dans les styles d'écriture entre l'écrivain Domenico Starnone et la célèbre romancière connue sous le pseudonyme d'Elena Ferrante. Enfin, en 2020, Rachel Mc Carthy et al [3] montrent qu'il est très peu probable que Branwell Brontë, le frère d'Emily Brontë, soit en réalité l'auteur des *Hautes de Hurlevent*, comme une partie de la critique le supposait.

2 Approche générale

Les techniques classiques de stylométrie, utilisées dans les exemples précédents, se basent sur une approche majoritairement statistique de l'analyse d'un texte. Si celles-ci évoluent constamment et permettent de nouveaux résultats chaque année, une approche alternative est proposée par Mariona Coll Ardanuy et Caroline Sporleder [4] en 2014 ; En partant du principe qu'un 'roman est une société miniature', elles se demandent si un auteur, au-delà de son style d'écriture, ne possède pas des caractéristiques qui lui sont propres dans la création d'une intrigue romanesque. Dans le projet présenté ci-dessous, je me suis inspiré de cette approche pour identifier de manière automatique les œuvres d'un corpus français en fonction de leur auteur. L'hypothèse est qu'un même auteur aura tendance à créer des intrigues semblables, qu'on pourra représenter par les interactions entre les différents personnages. Pour cela, j'ai construit pour chaque roman, la matrice de co-occurrences des interactions entre les personnages, que j'ai représentée sous forme d'un graphe. Sur la base de métriques calculées au sein de chaque graphe, j'ai construit un vecteur unique de features par œuvre. J'ai ensuite appliqué un algorithme de clustering sur la matrice de données construite, afin de détecter automatiquement les auteurs des œuvres considérées.

3 Preprocessing

Sources

Afin de vérifier l'hypothèse proposée ci-dessus, j'ai choisi un ensemble de 47 textes de la littérature française, produits par 6 auteurs différents. Ces documents ont été téléchargés depuis la plateforme du projet Gutenberg [1] et sont détaillées dans le premier tableau d'annexe 3.

1. http ://gutenberg.org/ - Le projet Gutenberg est une librairie en ligne gratuite de eBooks, fondée par Michael Hart.

Plusieurs étapes sont nécessaires afin de traiter les documents sous forme de textes bruts. Ces étapes sont construites sous la forme d'un Pipeline d'actions, modulable facilement et applicable à chaque nouveau texte.

Nettoyage du texte

La première étape de ce Pipeline consiste à tokeniser le texte (dissocier chaque mots), et à le nettoyer, en faisant une première sélection pour ôter la ponctuation. Une étape importante pour réduire d'emblée la volumétrie du texte considéré est de retirer les stopwords [2].

Reconnaissance d'entités nommées

Cette étape consiste à rechercher des objets précis au sein d'un texte et à les catégoriser dans des classes. Pour notre étude, il s'agira de trouver les objets classés comme des personnes. J'utilise la librairie *Spacy* (python) qui met à disposition un algorithme [3] classant les objets en 4 catégories ; personne (PER), lieu (LOC), organisation (ORG) et divers (MISC). Cette étape est complexe et peut générer des incohérences dans les labels obtenus pour la classe PER, comme illustré dans la Figure 1. Pour remédier en partie à ces problèmes, j'analyse les classes obtenues sur des échantillons connus, afin de mettre en place des règles de gestion supplémentaires comme ; regarder la taille des mots classés et sélectionner le plus grand afin d'éliminer les titres des personnages (Mlle, Mr, Comte, etc), ou encore, je considère la classe majoritaire (ici $> 50\%$) comme la classe définitive de l'objet.

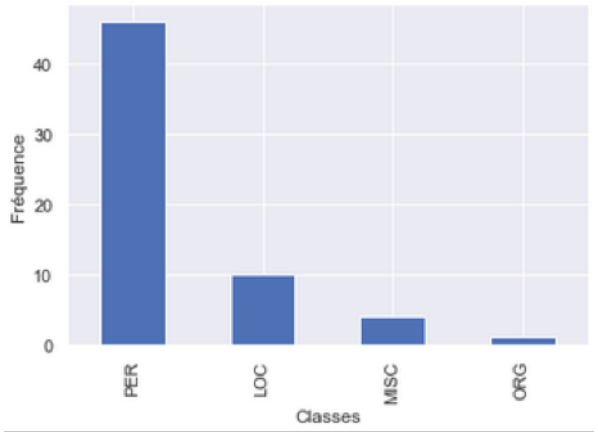

FIGURE 1 – Résultats du nommage d'entité – Personnage « Danglars » issu du Comte de Monte Cristo (Dumas)

Fenêtre de texte

Chaque œuvre est scindée en plusieurs fenêtres de texte, qui vont permettre de dégager les interactions des personnages sur un temps donné dans le récit. L'idée est de ne pas considérer le roman dans sa totalité, mais d'essayer de coller au schéma narratif et de découper l'histoire en sous-divisions. Ainsi, si deux personnages ne se rencontrent pas dans le récit, il n'existera aucune interaction entre eux dans le graphe final reliant les personnages. La définition de la taille de ces fenêtres de texte pose question. Une première approche peut être de scinder le texte original en fonction des paragraphes, mais le découpage de ceux-ci dans le projet Gutenberg n'étant pas normalisé, il est difficile d'automatiser cette étape (à court terme). L'alternative employée ici est la définition fixe d'une taille de fenêtre, correspondant chacune à 4% du texte nettoyé, soit 25 chapitres hypothétiques par œuvre.

2. Stopwords : « mots vides », qui n'apportent pas beaucoup d'information dans l'analyse d'un texte
3. Fr_core_news_sm : Réseau de neurones pré-entraîné

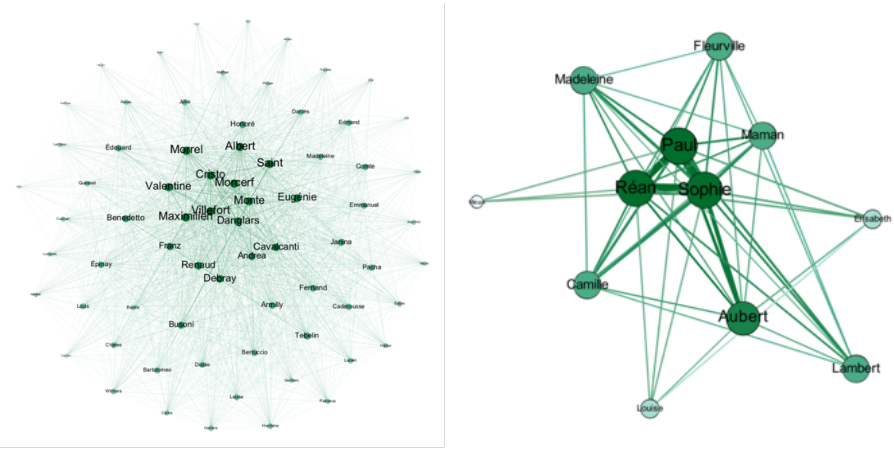

FIGURE 2 – Comparaison des graphes d'interactions des personnages dans deux romans (à gauche, "le comte de Monte Cristo" par Alexandre Dumas, à droite "les malheurs de Sophie" par la comtesse de Ségur

Matrice de cooccurrences et graphe d'interactions

Une des étapes principales du programme consiste à comptabiliser les interactions entre les personnages au sein d'une fenêtre de texte. Pour cela, je construis une matrice de cooccurrences des apparitions des personnages ; deux personnages cités de manière rapprochée dans le temps du récit, seront connectés au sein de cette matrice. Au travers de cette matrice, je retiens également le nombre d'interactions entre chaque personnage. Ce nombre servira dans la suite à pondérer les liens dans le graphe des interactions entre personnages, afin de dégager l'importance des échanges au fil du récit.

Pour chaque fenêtre de texte parcourue, on crée donc une matrice de coocurences, puis on agrège toutes ces matrices afin de constituer une unique vision des interactions du roman. De cette manière, si deux personnages ne se rencontrent jamais dans le récit, il n'existera aucun lien entre eux dans la matrice finale pour l'œuvre.

Étant donné le caractère bilatéral d'une discussion entre personnages, les données de la matrice de cooccurrences sont représentées sous la forme d'un **graphe**[4] **non orienté**. Afin de déceler les personnages centraux et les relations importantes dans le récit, les arrêtes du graphe sont **pondérées** par le nombre d'interactions entre personnages. La Figure 2, construite avec le logiciel libre *Gephi*[5] , montre le graphe d'interactions des personnages pour deux œuvres sélectionnées[6]. On peut noter que la structure de l'intrigue diffère largement, avec d'un côté une œuvre dont les personnages sont nombreux et quelques-uns très centraux ; le roman d'Alexandre Dumas raconte en effet les péripéties d'un groupe de personnages très liés pendant de nombreuses années. Durant tout le récit, chacun d'eux change d'identité, se déguise, voyage etc, ce qui augmente considérablement la quantité d'information récoltée dans le graphe. De l'autre côté, la Comtesse de Ségur évoque les aventures d'un cercle très restreint de personnages, centré autour de Sophie et son cousin Paul. Les personnages sont des enfants et évoluent dans un milieu familial et géographique clôt. Ce sont ces différences dans la construction du récit que nous allons tenter d'exploiter dans la suite de l'analyse.

4 Features engineering sur graphes

Le réseau d'interactions des personnages étant créé, il s'agit de définir des métriques pertinentes qui permettront de comparer ces graphes afin de dégager l'auteur des romans associé. Ces features *(22 au total)* - détaillées dans le Tableau 4 en annexe – se classent en quatre catégories :

4. Grâce à la libraire python NetworkX
5. https ://gephi.org/ : Complément à NetworkX, notamment pour la visualisation des graphes
6. Alexandre Dumas – « Le comte de Monte Cristo » et la comtesse de Ségur – « les malheurs de Sophie ». L'exemple montre volontairement des graphes très hétérogènes.

— Les métriques globales de graphes, ou l'on cherche à extraire une intuition générale de la forme du réseau d'interactions dans sa totalité ; densité, degré moyen, nombre de cliques, etc. Elles retranscrivent l'intensité de la fréquence des relations entre les personnages, leur proportion à se rencontrer dans le récit ou encore l'existence brève ou non de certains personnages.
— Des métriques de centralité, permettant de traduire l'importance d'un personnage par rapport à ses voisins ; la centralité [7] maximale (le personnage le plus « prestigieux » du roman, *resp.* la seconde centralité maximale et la troisième centralité maximale), la proximité [8] maximale (pour les trois premiers personnages) et une mesure du rôle d'intermédiaire [9] (des trois premiers personnages). On cherche ici à retranscrire la capacité d'un personnage à se positionner comme un intermédiaire dans le récit, ou encore sa capacité à affecter l'intrigue par sa présence ou non.
— Des métriques de clustering, permettant d'établir à quel point un personnage crée un schéma en 'étoile' dans le graphe ; coefficient de clustering maximal, coefficient de clustering du graphe sans le personnage lié au plus fort coefficient, etc. Un roman où les personnages ont tendance à maximiser cette métrique, sera un roman avec des personnages vers lesquels tendent à se rapprocher tous les autres.
— Des métriques de poids relatif pour des groupes de personnages ; calcul du poids relatif d'un nœud comme le rapport entre le poids du nœud considéré et la somme des poids des nœuds restants dans le graphe. On considère comme feature le poids relatif maximal, le second plus élevé, et le poids relatif du groupe des 10 premiers personnages. Ces métriques nous donnent une intuition quant à l'importance d'un groupe de personnage par rapport à l'ensemble des participants du récit.

Pour chaque œuvre considéré, un vecteur des métriques citées ci-dessus sera donc défini. L'ensemble de ces vecteurs constituera la base de données du modèle de clustering.

5 Clustering

Chaque roman étant désormais représenté sous forme d'un vecteur de paramètre, il s'agit de construire un algorithme permettant de différencier ces vecteurs. Pour cela, j'ai choisi d'utiliser un algorithme de clustering (apprentissage non supervisé), afin de laisser la machine créer des clusters d'auteurs. Afin d'avoir une idée plus concrète des différences entre romans (sous leur forme de features de graphes d'interactions), j'applique dans un premier temps une phase d'Analyse en Composantes Principales (ACP). Le but de cette étape est double, puisqu'il permet d'obtenir une représentation visuelle en 2 dimensions des données (les deux premières composantes principales, qui expliquent à elles seules plus de 60% de l'inertie – voir Figure 3), mais également peut permettre de réduire la dimension du jeu de données. En effet, dans notre cas, le nombre de covariables du modèle est de 22 pour un jeu de données de très petite dimension : 47 œuvres, ce qui peut générer des problèmes dans l'apprentissage.

Puis, dans un deuxième temps, j'applique l'algorithme de K-Means [10] en prenant pour nombre de clusters, le nombre d'auteurs initialement présent dans le jeu données, soit 6. Les représentations graphiques du jeu de données après ACP, ainsi que des résultats de l'algorithme KMeans, pour des données avec et sans ACP sont illustrées dans la Figure 4. On voit notamment que, si la comtesse de Ségur semble créer des intrigues facilement différentiables (figure de gauche), il peut être compliqué même pour un humain de discerner clairement la frontière entre d'autres auteurs (Stendhal et Jules Verne par exemple).

Une fois cet algorithme appliqué, il est également complexe d'analyser les différences de résultats (par exemple entre la méthode avec et sans ACP), puisqu'une étape de labellisation des clusters par un humain ayant la connaissance des données initiales est nécessaire. Afin de simplifier cette dernière contrainte, j'ai choisi d'utiliser dans une approche parallèle, le logiciel libre Weka [11] permettant une analyse simplifiée de ce type de problèmes. Pour cela, je teste

7. Eigen vector centrality
8. Closeness centrality
9. Betweeness centrality
10. Fonction *KMeans()* de la librairie Scikit-Learn
11. https ://waikato.github.io/weka-wiki/

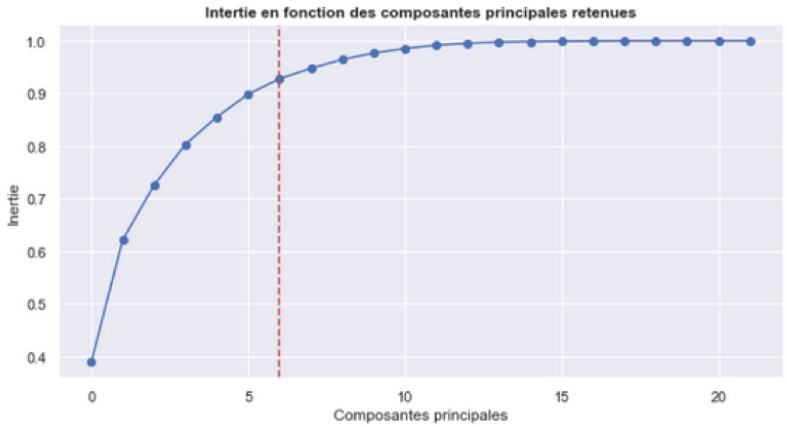

FIGURE 3 – Variance expliquée en fonction des composantes principales

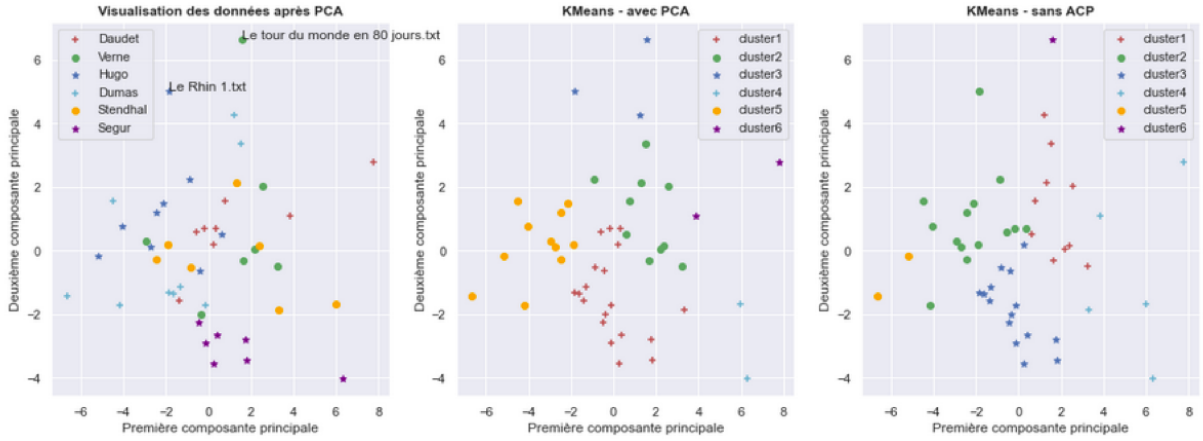

FIGURE 4 – KMeans Clustering - (Scikit-learn)

également les méthodes de clustering avec et sans ACP, afin de vérifier la plus-value de cette phase de réduction de données. Puis je compare les résultats des clusters créés [12] selon deux critères distincts ; la pureté (critère externe) définie comme le rapport entre la somme des éléments correctement labellisés par cluster et le nombre total d'éléments [5] :

$$purity = \frac{1}{N} \sum_{k} max_j(W_k \cap C_J)$$

Avec N le nombre total d'éléments, k le nombre de clusters et C le nombre de classes.

et le F1-Score :

$$F1 - score = 2 * \frac{precision * recall}{precision + recall}$$

Le Tableau 1 récapitule les différents résultats de pureté pour les versions d'algorithmes de clustering testées dans Weka, sur les données avec et sans ACP. Les méthodes KMeans et EM sont celles dont les résultats sont les meilleurs (*resp.* 51% et 49%), et on note une amélioration significative pour la méthode KMeans lorsqu'une ACP est réalisée en amont sur le jeu de données (66%).

Pour les deux meilleures méthodes citées ci-dessus (KMeans et EM avec ACP), les F1-scores sont repris dans le Tableau 2. C'est l'algorithme de KMeans qui permet une classification des auteurs la plus proche de la réalité, en effet, le F1 score atteint est de 53% (resp 46% pour EM).

12. J'utilise la fonction *classes_to_clusters* de Weka : le label associé au cluster créé est le label de l'item le plus représenté dans le cluster

ACP	Nb features	KMeans	EM	Hierachical clustering
Sans ACP	22	0.51	0.49	0.28
Avec ACP	6	0.66	0.51	0.28

TABLE 1 – Résultats de pureté pour les différentes méthodes de clustering

Modèle	Nb features	Precision	Recall	F1-Score
KMeans avec ACP	6	0.50	0.57	0.53
EM avec ACP	6	0.38	0.60	0.46

TABLE 2 – Precision, Recall et F1-scores pour 2 méthodes de clustering

6 Conclusion

L'utilisation des graphes d'interactions des personnages d'œuvres littéraires pour reconnaître les auteurs de celles-ci, permet d'obtenir un résultat maximum de pureté de 66% (KMeans avec ACP), supérieur à un modèle aléatoire. Dans une prochaine étape, il pourrait être intéressant de comparer ces résultats à ceux obtenus sur le l'approche stylographique de ce type de problème (pour la littérature française). Pour l'approfondissement de la méthode présentée, il serait utile de se concentrer plus spécifiquement sur la phase de reconnaissance d'entités nommées, qui génère beaucoup de « bruit » et est particulièrement sensible au style de l'auteur. Notamment, l'utilisation des titres (Monsieur, Mlle, Prince, etc), très usitée dans certains romans, gêne énormément le nommage des objets de l'œuvre. La quantification de ce bruit n'est pas non plus abordée dans ce projet.

Références

[1] Pennebaker JW Boyd RL. Did shakespeare write double falsehood ? identifying individuals by creating psychological signatures with text analysis. *Psychol Sci.*, 2015.

[2] Jacques Savoy. Elena ferrante unmasked. 09 2017.

[3] Rachel McCarthy and James O'Sullivan. Who wrote Wuthering Heights ? *Digital Scholarship in the Humanities*, 06 2020.

[4] Mariona Coll Ardanuy and Caroline Sporleder. Structure-based clustering of novels. EACL 2014 : Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLfL), 2014.

[5] Hinrich Schütze Christopher D. Manning, Prabhakar Raghavan. *Introduction to information retrieval*. Cambridge University Press, 2010.

7 Annexes

Auteur	Oeuvre
Alphonse Daudet	Les femmes d'artiste
Alphonse Daudet	La belle hivernaise
Alphonse Daudet	Le petit chose
Alphonse Daudet	Port Tarascon
Alphonse Daudet	Tartarin sur les Alpes
Alphonse Daudet	L'immortel
Alphonse Daudet	Le nabab 1
Alphonse Daudet	Le nabab 2
Jules Vernes	Voyage au centre de la terre
Jules Vernes	L'île mystérieuse
Jules Vernes	Les enfants du capitaine Grant
Jules Vernes	Cinq semaines en ballon
Jules Vernes	Le tour du monde en 80 jours
Jules Vernes	Vingt mille lieues sous les mer
Jules Vernes	De la terre à la lune
Victor Hugo	Le Rhin 1
Victor Hugo	Le Rhin 2
Victor Hugo	Les misérables 1
Victor Hugo	Les misérables 2
Victor Hugo	Les misérables 3
Victor Hugo	Quatrevingt treize
Victor Hugo	Napoléon le petit
Victor Hugo	Han d'Islande
Victor Hugo	Notre Dame de Paris

Auteur	Oeuvre
Alexandre Dumas	Gabriel Lambert
Alexandre Dumas	Les mille et un fantômes
Alexandre Dumas	Le chevalier de maison rouge
Alexandre Dumas	Le comte de Monte Cristo 1
Alexandre Dumas	Le comte de Monte Cristo 2
Alexandre Dumas	La tulipe noire
Alexandre Dumas	Les trois mousquetaires 1
Alexandre Dumas	Les trois mousquetaires 2
Alexandre Dumas	Vingt ans après
Stendhal	Les Cenci
Stendhal	Lucien Leuwen 1
Stendhal	Lucien Leuwen 2
Stendhal	Le rouge et le noir
Stendhal	La duchesse de Paliano
Stendhal	L'abesse de Castro
Stendhal	La chartreuse de Parme
La comtesse de Segur	Jean qui grogne Jean qui rit
La comtesse de Segur	L'auberge de l'ange gardien
La comtesse de Segur	Les malheurs de Sophie
La comtesse de Segur	Les vacances
La comtesse de Segur	Un bon petit diable
La comtesse de Segur	Les deux nigauds
La comtesse de Segur	Le général Dourakine

TABLE 3 – Liste des oeuvres littéraires sélectionnées

Feature label	Description
avg_deg	Degré moyen du graphe
median_deg	Degré médian du graphe
number_cliques	Nombre de cliques dans le graphe
density	Densité du graphe
proportion_of_isolates	Proportion de nœuds isolés dans le graphe
len_of_graph	Taille du graphe (i.e nombre de nœuds)
avg_centrality	Centralité moyenne du graphe
eigen_centrality_1char	Centralité la plus élevée dans le graphe
eigen_centrality_2char	2ème centralité la plus élevée
eigen_centrality_3char	3ème centralité la plus élevée
closeness_central_1char	Closeness centralité la plus élevée dans le graphe
closeness_central_2char	2ème closeness centralité la plus élevée
closeness_central_3char	3ème closeness centralité la plus élevée
betweenness_central_1char	Betweeness centralité la plus élevée dans le graphe
betweenness_central_2char	2ème betweeness centralité la plus élevée
betweenness_central_3char	3ème betweeness centralité la plus élevée
avg_clustering	Coefficient de clustering moyen dans le graphe
clustering_coef_main	Coefficient de clustering le plus élevé
avg_clustering_without_main	Coefficient de clustering le plus élevé hors le 1erpersonnage
relative_weight_main	Poids relatif maximal du graphe
relative_weight_second	2ème poids relatif maximal
relative_weight_ten	3ème poids relatif maximal

TABLE 4 – Détail des features du modèle