Quel workflow pour les sciences du texte ?

Antoine Widlöcher

Quel *workflow* pour les sciences du texte ?

Antoine Widlöcher
Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France
antoine.widlocher@unicaen.fr

RÉSUMÉ

Le triomphe des approches adossées à des méthodes d'apprentissage, dans de nombreuses branches de notre discipline, tend à occulter une part importante des domaines d'investigation pourtant intimement liée au traitement automatique des langues. Nous proposerons, pour commencer, de faire un pas dans la direction opposée, en faveur de ce que nous nommerons ici les *sciences du texte*, en les distinguant de l'ingénierie de la langue, dont l'omniprésence explique largement cette occultation. Nous voudrions ensuite contribuer à mettre en évidence la méthode propre à cette branche des savoirs, méthode commune pouvant permettre de faire sortir de l'isolement des travaux hétérogènes liés par un même rapport au texte. Nous voudrions enfin nous concentrer sur la phase de ce *workflow* qui demeure actuellement la plus difficile, celle de l'expérimentation sur corpus, et proposer un cadre pour la mise en place d'environnements d'expérimentation appropriés.

ABSTRACT

What workflow for text science ?

The triumphal success of approaches based on machine-learning methods, in many branches of our discipline, tends to marginalise a large part of the fields of research which are nevertheless intimately linked to natural language processing. First of all, we propose to take a step in the opposite direction, in favour of what we call here *text sciences*, distinguishing them from human language technologies, whose omnipresence largely explains this marginalisation. Then we would like to contribute to highlight the specific method of theses approaches, which share a common relationship with the text. Finally, we would like to focus on the phase of their *workflow* that currently remains the most difficult, that of corpus-based experimentation, and to propose a framework for setting up appropriate environments for experimentation.

MOTS-CLÉS : Sciences du texte, environnement d'expérimentation, expérimentation sur corpus.

KEYWORDS: Text science, experimental environement, corpus-based experimentation.

1 Sciences et ingénierie du texte

Une part très importante des travaux de notre communauté se porte désormais naturellement vers les approches adossées à des méthodes d'apprentissage, dont les succès fulgurants ne sont pas contestables. Que cette tendance forte ait d'ores et déjà transformé en profondeur notre discipline ne doit pas nous dispenser de méditer d'une part à la nature et à la portée de ces succès et d'autre part à la zone d'occultation qui résulte de cette forte mise en lumière. Pour tirer au clair la portée de ces succès incontestables, il nous semble important d'en souligner l'horizon applicatif. Qu'il s'agisse de classification de documents, de traduction, d'extraction d'information ou de résumé, pour citer

quelques exemples bien documentés des *applications* du TAL, la tentation d'y voir le texte comme un obstacle à surmonter, ne doit pas être minimisée. Le texte n'y est pas alors regardé comme une fin, mais comme un moyen d'atteindre certains objectifs applicatifs, la pertinence d'une approche étant alors potentiellement mesurée à l'aune d'un critère étranger au texte lui-même et à sa compréhension.

D'autres travaux, au contraire, s'appuyant aussi sur une machinerie computationnelle, visent l'étude du texte lui-même, la mise en lumière de ses lois. Là où les précédents, qui relèvent selon nous pour cette raison de l'*ingénierie du texte*, visent quelque chose derrière le texte, au moyen du texte, ces derniers, de l'ordre de la *science du texte*, visent le texte lui-même et sa compréhension. Si cette ligne de démarcation au sein des « disciplines du texte » évoque celle qu'identifie (Rastier, 2001) entre *arts* et *sciences* du texte, elle renvoie davantage ici à une différence de visée entre les différents paradigmes d'étude, au sein des approches scientifiques. Non sans rappeler aussi de vieilles querelles entre TAL et linguistique computationnelle, cette cartographie disciplinaire doit être précisée. On pourrait dire que les approches visant le texte lui-même et celles qui sont guidées par des impératifs applicatifs ont le TAL en commun. Celui-ci renvoie à un ensemble de méthodes de traitement des données textuelles utilisable dans une perspective ou dans l'autre. Ainsi, la linguistique computationnelle ne s'oppose pas au TAL mais le précise, par sa visée spécifique, son absence d'autres objets que le texte lui-même et sa compréhension. Mais elle n'épuise pas le concept des sciences du texte, dont relèvent tout autant, par exemple, des études littéraires ou philologiques et de nombreux chantiers ouverts à l'interface des humanités numériques, là où les sciences humaines rencontrent le besoin d'explorer computationnellement le texte, pour le comprendre. Pour ces disciplines, l'identification d'un phénomène textuel ne suffit pas ; il faut encore comprendre comment et pourquoi il se produit. Cette dimension intrinsèquement explicative y va de pair avec l'omniprésence de l'interprète humain.

Faire entendre la voix des sciences du texte, dans un contexte disciplinaire où l'omniprésence des impératifs de l'ingénierie du texte ne cesse d'en fragiliser l'existence, est-ce à dire pour autant qu'une hétérogénéité radicale devrait interdire toute communication entre ces disciplines ? Évidemment non. D'une part parce qu'une communauté de moyens (de formalisation, de calcul...) rend évidemment possible des avancées communes. D'autre part parce que la science du texte pourra toujours en droit éclairer son ingénierie, comme l'ont souvent montré les approches linguistiquement informées du TAL, qui n'ont certes pas actuellement le vent en poupe... Enfin, car les moyens puissants élaborés pour son ingénierie peuvent constituer aussi des moyens d'observation utiles à la science du texte, pourvu que sa visée explicative soit bien entendue, ce à quoi les nombreux travaux actuels consacrés à l'explicabilité dans les méthodes d'apprentissage pourraient évidemment contribuer. Reste que les travaux relevant de la science du texte, travaux visant exclusivement le texte et sa compréhension, doivent pouvoir exister. Leur marginalisation relative actuelle impose de repenser leurs spécificités et leur fond théorique commun, ne serait-ce que pour mettre en évidence leur omniprésence dans notre communauté et pour mettre en lumière les moyens dont ils doivent disposer pour s'y épanouir.

2 Quel *workflow* pour les sciences du texte ?

Par *sciences du texte*, nous désignons le complexe théorique et expérimental qui vise l'étude et la compréhension du texte et de ses lois, à différentes échelles (du caractère au corpus) et dans différentes perspectives (de la forme à l'interprétation). Leur scientificité repose sur les éléments suivants :

1. Leur **démarche expérimentale** s'appuie sur l'articulation entre des phases inductives et hypothético-déductives menées par confrontation aux données de l'expérience, c'est-à-dire aux données d'un **corpus**.

2. L'exigence de validation sur corpus suppose la capacité à identifier les données de l'expérience dont l'étude est menée, c'est-à-dire la capacité à **constituer des observables** dont le modèle sera modèle, dont la théorie sera théorie...
3. L'ensemble de leur démarche doit satisfaire, dans chacune de ses phases, aux **exigences de reproductibilité**. Cela suppose qu'un degré de formalisation suffisant soit atteint dans l'énoncé des modèles, des hypothèses et des paramètres d'expérimentation, pour que la communauté puisse vérifier que, dans les mêmes conditions, les mêmes causes conduisent aux mêmes effets et identifier, le cas échéant, erreurs et biais dans l'interprétation des résultats, conformément à l'**impératif de réfutabilité** qui prolonge naturellement celui de reproductibilité.

De ces contraintes épistémologiques résulte un schéma de *workflow* dans lequel s'inscriront naturellement les travaux en sciences du texte. La figure 1 donne sa forme essentielle.

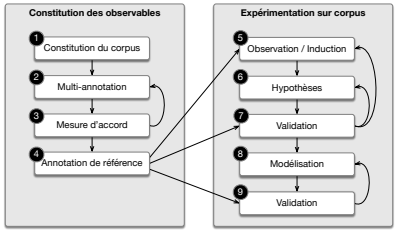

FIGURE 1 – Forme générale du *workflow* des sciences du texte

Nous distinguons ici deux grandes phases au sein de ce *workflow*. Les phases ❶ à ❹ correspondent à la constitution des observables, c'est-à-dire à la constitution d'un corpus de référence au sein duquel des occurrences du phénomène textuel ciblé ont été identifiées. Les phases ❺ à ❾ visent l'établissement d'un modèle du phénomène ciblé.

2.1 La constitution des observables

Les phases ❶ à ❹ sont très largement balisées, même si certains problèmes restent évidemment ouverts et si un défaut de systématicité est encore nettement observable à l'échelle de notre communauté. Des facteurs de blocage en résultent qui doivent être mis en évidence. La question de la constitution de corpus et celle de leur représentativité fait l'objet d'une littérature relativement abondante, dont la revue *Corpus* est par exemple le reflet dans la communauté francophone, depuis ses premiers numéros (Mellet, 2002). Un point important concerne aussi la diffusion des corpus. Sur ce point, on peut s'étonner que, si un assez net consensus se dégage concernant l'utilisation de schémas éprouvés de représentation des corpus textuels, et cela depuis fort longtemps, notamment autour de la TEI [1] (Ide & Véronis, 1995; Burnard *et al.*, 2006), le disponibilité de corpus de référence selon ce standard de fait demeure assez sporadique dans nos disciplines, alors que les sciences humaines et les disciplines littéraires, soucieuses d'établissement de données finement ciselées, se sont pour leur part nettement approprié les technologies associées. Si l'inscription dans la nébuleuse XML est acquise, l'absence de systématicité dans le recours à un vocabulaire commun est pourtant un évident facteur de ralentissement.

1. https://tei-c.org/

La méthodologie d'ensemble selon laquelle peuvent être menées les campagnes d'élaboration de ressources enrichies, annotées, fait l'objet d'une littérature très utile (Fort, 2016). L'importance de la multi-annotation ❷ a été largement soulignée pour garantir la fiabilité des données produites, en particulier sur les objets textuels peu étudiés ou difficiles à interpréter, et, au moins depuis (Artstein & Poesio, 2008), les moyens de mesurer l'émergence d'un consensus entre annotateurs ❸, condition *sine qua non* pour la constitution de données *de référence* ❹, sont étudiés en tant que tels. Différents environnements logiciels dédiés à l'annotation ont été proposées, parmi lesquels on peut notamment citer (Stenetorp *et al.*, 2012), (Widlöcher & Mathet, 2012) et plus récemment (Klie *et al.*, 2018). À mi-chemin entre l'annotation et l'exploration de corpus telle qu'elle sera définie ci-dessous, on peut également mentionner les travaux de (Landragin *et al.*, 2012), qui combinent ces deux phases.

Bien entendu, la diversité et l'hétérogénéité des phénomènes susceptibles d'être étudiés et annotés retardent l'élaboration de modèles communs de représentation des données enrichies, mais aussi, inévitablement, la mise en place de méthodes communes pour la comparaisons de structures annotées, préalable pourtant nécessaire à la mesure d'accord sur des données multi-annotées, dont l'absence bloque l'émergence de données de référence. De fait, si les moyens ne manquent pas pour la prise en charge de tâches d'annotation relevant de la pure catégorisation d'items textuels déjà identifiés, les propositions sont dramatiquement moins nombreuses pour des tâches imposant de surcroît le positionnement des objets dans le *continuum* textuel. Et si des propositions voient le jour, quoique très sporadiquement, pour la prise en charge des tâches dites d'*unitizing* (de repérage d'unités dans le texte), c'est-à-dire pour leur annotation puis pour la comparaison des productions de plusieurs annotateurs, le travail sur des structures plus complexes, notamment relationnelles (par exemple en syntaxe, rhétorique ou argumentation), impose souvent le recours à des méthodes *ad hoc*, élaborées pour chaque objet, en rendant fatalement difficile la confrontation des modèles et des théories.

2.2 Absence d'un cadre commun pour l'expérimentation sur corpus

Sur la seule question de la constitution des observables, un immense travail reste donc encore à accomplir, pour définir objets et méthodes communs. Nous voudrions toutefois nous concentrer sur les phases suivantes du *workflow*, celles de l'expérimentation sur corpus (❺ à ❾). Car si chacun, dans sa discipline, avance évidemment sur ce terrain, nous ne voyons pas cependant émerger un cadre commun qui permettrait, à l'échelle de la communauté, la reproduction simple des expériences menées ici ou là, la confrontation des approches, le partage des résultats. Que l'absence d'un tel cadre commun soit dommageable, tout le monde en conviendra. Nous voudrions savoir à quelles conditions son émergence pourrait être rendue possible et, inversement, quels éléments font blocage.

La question pourtant n'est pas neuve. Sans remonter aux origines de notre discipline – car en réalité les conditions que nous mentionnons sont presque imposées par la démarche scientifique elle-même – on retiendra notamment qu'il y a quinze ans déjà, (Enjalbert *et al.*, 2008), dans le prolongement de journées ATALA consacrées aux « architectures logicielles pour articuler les traitements sur corpus » (en 2005), notre communauté francophone se posait frontalement la question de la mise en place d'environnements d'expérimentation sur corpus, de la reproductibilité des expériences, du partage des ressources, de l'interopérabilité entre les systèmes... Et depuis lors au moins, différentes solutions méthodologiques et logicielles ont été proposées pour répondre à ces exigences. Nous voudrions ici proposer quelques repères dans cette nébuleuse, en cherchant surtout à identifier les paradigmes concurrents et les lignes de démarcation principales entre les différentes options envisagées, pour mieux saisir les raisons pour lesquelles principes et solutions communs tardent à émerger.

Pour nous orienter dans cette nébuleuse [2], il peut être utile de commencer par distinguer deux traditions restées jusqu'ici assez étanches l'une à l'autre, répondant à deux manières d'envisager les données textuelles. Nous proposons de formuler l'esprit de cette démarcation en nous appuyant sur la distinction souvent faite, notamment dans le champ des humanités numériques, entre *distant reading* (Moretti, 2013) et *close reading*, pour désigner l'hétérogénéité entre des approches considérant les données textuelles, souvent assez massives, d'une certaine hauteur et souvent au moyen de méthodes statistiques, et des approches restant davantage au contact des énoncés et des occurrences en contexte.

Relèvent clairement du premier paradigme les propositions faites dans le champs de l'analyse statistique des données textuelles (Lebart *et al.*, 1998) et notamment, surtout au niveau francophone, les travaux de la tradition lexicométrique issue de (Lafon, 1984), dont sont inspirés des environnements intégrés très complets d'exploration de corpus tels que la plate-forme TXM (Heiden, 2010).

Visant davantage l'exploration du corpus en restant au contact des énoncés, en pilotant généralement la recherche d'occurrences des phénomènes visés par l'expression de règles établies dans des formalismes dédiés à des niveaux d'analyse variés (échelles lexicale, syntagmatique, discursive...), d'autres environnements intégrés d'expérimentation sur corpus ont vu le jour, parmi lesquels ont peut citer notamment Gate (Cunningham *et al.*, 2002, 2011, 2013), Nooj (Silberztein, 2016), Unitex (Paumier *et al.*, 2021) et LinguaStream (Widlöcher & Bilhaut, 2008). L'enthousiasme suscité par ces environnements intégrés, puissants mais difficiles à prendre en main, semble avoir connu un certain infléchissement. Si Gate, Nooj et Unitex sont toujours maintenus et jouissent d'une communauté active (ce n'est pas le cas de LinguaStream), on a néanmoins le sentiment que la communication scientifique associée à ces plate-formes a sensiblement diminué ces dernières années, au-delà de cercles assez spécifiques, signe peut-être d'un relatif déphasage par rapport aux attentes de notre communauté. On ne voit pas, du moins, émerger un consensus large en faveur d'un environnement commun d'expérimentation en TAL qui reposerait sur ce principe.

On a vu au contraire se multiplier les approches *par librairie* visant, plutôt que l'élaboration d'un environnement intégré, l'exploitation depuis un langage d'interfaçage ou d'intégration tel que Python notamment, très apprécié pour le prototypage rapide et dans de nombreuses sciences expérimentales. À des outils dédiés à la langue comme NLTK (Bird *et al.*, 2009), régulièrement utilisé pour la recherche et l'enseignement, en vertu notamment de la pluralité des paradigmes d'analyse qu'il permet d'exploiter et le contrôle qu'il donne sur l'expression de règles dans différents formalismes, il convient d'ajouter d'une part des outils comme spaCy [3], certes dédiés à la matière textuelle, mais visant davantage la mise en production d'applications que l'expérimentation sur corpus, et d'autre part des outils dédiés à la science des données et à l'apprentissage machine, tel Scikit-learn (Pedregosa *et al.*, 2011), qui intègrent des éléments permettant de traiter des données textuelles, dont les spécificités sont d'ailleurs souvent écartées assez rapidement, au profit de représentations tabulaires et vectorielles évidemment plus en phase avec les méthodes courantes en apprentissage.

2.3 Quel cadre pour l'expérimentation sur corpus ?

La partie expérimentale de nos disciplines offre donc encore souvent le spectacle d'une collection d'approches difficiles à unifier dans un mouvement commun, où chaque travail avance son propre

2. Nous laissons de côté, dans ce rapide survol, des infrastructures d'assez bas niveau, comme notamment le *framework* UIMA (https://uima.apache.org/), et les questions importantes qu'elles posent en termes d'interopérabilité, pour nous concentrer en priorité sur les environnements plus immédiatement dédiés à l'analyse des textes.

3. https://spacy.io

formalisme, sa propre représentation du texte, des règles d'analyse... Conscient que toute tentative d'avancer à rebours de cette tendance naturellement entropique nous fait prendre le risque d'une fausse solution de plus, et donc finalement d'une augmentation du désordre, nous voudrions néanmoins envisager quelques pistes pour essayer d'y remédier. Nous les présenterons sous la forme d'une série de principes, non sans avoir d'abord souligné que nous maintenons largement les recommandations faites par (Widlöcher & Bilhaut, 2008), recommandations que nous complétons et que nous proposons d'amender, parfois en les radicalisant, parfois en relaxant certaines contraintes difficiles à satisfaire sans entrer en contradiction avec d'autres principes d'importance égale ou supérieure.

P1 - Hétéronomie fondamentale du chercheur De façon peut-être un peu provocante, nous voudrions aborder cette énumération des principes par des considérations de nature presque sociologique concernant certains *habitus* de notre communauté des sciences du texte. Nous pensons ici plus précisément à la tradition relativement forte de l'autonomie radicale du chercheur, par laquelle nos disciplines s'inscrivent d'ailleurs dans le prolongement des sciences humaines et de l'esprit, où l'autorité du savant suppose souvent la solitude. Nous entendons par là l'ambition (et souvent d'ailleurs la capacité admirable) du chercheur à maîtriser individuellement l'ensemble des phases du processus de construction intellectuelle, conceptuelle et expérimentale qu'implique l'étude des objets qu'il s'est fixés [4]. Or, pour admirable qu'elle soit, cette parfaite autonomie n'en demeure pas moins tout à fait exceptionnelle en pratique, et le risque est grand dès lors que cette ambition devienne contre-productive si l'on n'en mesure pas la limite. Si nous évoquons ce tropisme, c'est qu'il ne nous semble pas étranger à l'échec relatif des grands systèmes intégrés, qui reposent en partie sur l'hypothèse, trompeuse selon nous, que chacun pourra, en autonomie, mener ses expérimentations sur corpus. Nous voudrions promouvoir au contraire l'idée d'une hétéronomie fondamentale du chercheur, et prendre la juste mesure de la nécessité qui en résulte de clarifier les moyens d'une collaboration féconde entre les différents corps de métiers impliqués dans les sciences du texte. Être linguiste, statisticien ou algorithmicien, ce n'est résolument pas la même chose (dans une large majorité des cas), et, plutôt que de viser l'objectif illusoire d'une autonomie parfaite, nous devons plutôt nous interroger sur les moyens de rendre fertile la collaboration (*cf.* **P4 P5**).

P2 - Données textuelles de référence, données d'entrée et données de sortie Que le partage des données importe davantage que celui des outils, c'est là un fait qui a été maintes fois souligné. Quelles conséquences pratiques devons-nous en tirer ? D'abord, la nécessité de prendre au sérieux le mode de représentation faisant consensus pour la représentation des corpus textuels. De ce point de vue, il est clair que les technologies XML, tombées en désuétude chez les informaticiens, demeurent incontournables pour la représentation des données semi-structurées dont les corpus textuels sont la parfaite illustration. Il en résulte non seulement la nécessité de mettre en place des environnements logiciels capables de consommer de telles données en entrée (ce qui est souvent admis), mais aussi la nécessité de produire en sortie des données répondant à cette norme, notamment pour que les données textuelles enrichies issues de l'analyse demeurent compatibles avec les outils d'observation et les chaînes éditoriales définies pour la diffusion et la valorisation des données initiales. S'il n'est pas mécaniquement nécessaire que les processus d'analyse opèrent en conséquence à chaque étape sur des représentations XML des données, potentiellement coûteuses, il est en revanche nécessaire que toutes les représentations intermédiaires fassent systématiquement référence aux structures initiales, pour qu'à chaque instant, et surtout en fin de traitement, elle puissent y être projetées ou rapportées.

P3 - Complémentarité des *altitudes* d'observation Par *altitude* d'observation, nous renvoyons ici à la distinction évoquée ci-dessus entre *close* et *distant reading*. Si de nombreux chercheurs appré-

4. Ce dont l'usage des productions scientifiques à signature unique est le reflet éloquent.

hendent alternativement les données avec des méthodes distantes (statistiques, lexicométriques...) et des méthodes plus directement focalisées sur le repérage d'occurrences en contexte, il faut bien néanmoins reconnaître que l'adoption de ces différents points de vue relève de traditions, de méthodologies et conséquemment d'outils assez hétérogènes. S'il est raisonnable de supposer la complémentarité de ces altitudes d'observation, les premières étant notamment indispensables dans les phases inductives et de vérifications des hypothèses à petite échelle (celle du corpus ou du sous-corpus), quand les secondes interviennent dans les phases de modélisation de structures potentiellement complexes et dans le repérage d'occurrences à grande échelle (en contexte, à l'échelle de la phrase ou du discours), il est alors nécessaire de disposer de moyens efficaces de circulation entre ces niveaux. Cela impose d'abord que les descriptions d'objets analysés en contexte puissent être aisément collectées et synthétisées dans des représentations manipulables par les outils d'observations de plus haute altitude. Inversement, cela suppose que des éléments observés à haute altitude (des fréquences, des régularités, des attirances...) puissent être facilement reformulés en configurations observables *in situ*. On ne saurait trop insister sur l'importance du *retour au texte* pour les sciences du texte. Elle impose en particulier la disponibilité d'outils de visualisation adaptés aux modes conventionnels de représentation des données (*cf.* **P2**), pour que les objets soient observés dans leur contexte initial.

P4 - Abstraction progressive des formes de surface et variabilité des perspectives Avoir le texte pour matière n'implique pas que chaque étape et chaque niveau d'analyse doive reposer exclusivement sur sa forme initiale. Au contraire, pour les traitements computationnels, comme pour les travaux réalisés sans traitement mécanique, il est clair que certains niveaux d'analyse doivent pouvoir s'appuyer sur les résultats obtenus à d'autres niveaux, certains ordres classiques conduisant même d'ailleurs à des *pipelines* parfois figés à l'excès dont l'enchaînement [Tokenisation > POS Tagging > Analyse syntaxique > Analyse du discours] donne une bonne illustration. S'il nous semble important de garantir au contraire une assez grande liberté dans les enchaînements d'analyse à mettre en place (*cf. infra*), reste qu'une analyse d'un niveau quelconque doit pouvoir s'appuyer sur les sorties d'un niveau d'analyse préalable, et qu'une analyse subséquente devra pouvoir, à son tour, s'appuyer sur ses propres sorties. Il en résulte que chaque niveau d'analyse doit pouvoir exploiter, non seulement la matière textuelle initiale et ses formes de surface, mais surtout les représentations antérieurement calculées, cette forme d'indirection conduisant en pratique à une *abstraction progressive des formes de surface*. Chaque niveau d'analyse, humain ou computationnel, produit des *annotations*, souvent obtenues par abstraction depuis des annotations déjà produites, sur lesquelles les analyses subséquentes devront à leur tour pouvoir s'appuyer. Ces annotations doivent combiner la *localisation* dans le texte des phénomènes identifiés, en référence à la représentation initiale (*cf.* **P2**), et une *représentation symbolique* de leur interprétation, une caractérisation utilisable par les traitements subséquents. Cela n'implique pas que chaque niveau d'analyse doive tenir compte de toutes les représentations préalablement calculées. Au contraire, chacun devra pouvoir spécifier la *perspective* qui est la sienne, c'est-à-dire la manière dont il se rapporte au texte, en explicitant les abstractions sur lesquelles il s'appuie. Les avantages qui peuvent en résulter, en terme d'expressivité de chaque modèle d'analyse et en termes d'efficacité sur un plan combinatoire, sont potentiellement colossaux. La confrontation des points de vue (entre le linguiste, l'informaticien...) est toutefois évidemment nécessaire à l'exploitation de ces bénéfices (*cf.* **P1**).

P5 - Complémentarité des formalismes et modèles d'analyse L'étude d'objets textuels variés a naturellement conduit à l'émergence de multiples formalismes et modèles d'analyse, dont la pouvoir expressif et l'efficacité ont été établis pour les objets pour lesquels ils ont été pensés. Viser la mise en place d'un cadre expérimental commun, ce n'est évidemment pas proposer en la matière un réductionnisme total supposant l'omnipotence d'un formalisme particulier. Il est au contraire

nécessaire de faire jouer la complémentarité des formalismes et modèles d'analyse. Nous défendons du reste l'idée que, même si un formalisme et un modèle d'analyse ont généralement été élaborés pour l'étude d'objets particuliers, leur exploitation à d'autres niveaux peut s'avérer d'autant plus féconde que la variabilité des perspectives sur le texte donne une grande liberté dans sa lecture (*cf.* **P4**). L'exploitation des automates et expressions régulières sur des séquences quelconques, au-delà du niveau des chaînes de caractères pour lesquelles les modèles initiaux ont été pensés, illustre bien l'extension possible d'un domaine d'application. Pour que cette extension demeure possible, il est nécessaire que chaque formalisme et modèle d'analyse retenu ne soit pas inféodé à une certaine représentation du texte qu'il consomme, mais puisse au contraire opérer depuis une perspective quelconque sur le texte. Reste que l'identification d'un modèle approprié, en termes d'expressivité et d'efficacité, pour un problème textuel donné, demeure un problème complexe, imposant la collaboration entre les différents corps de métier (*cf.* **P1**).

P6 - Représentation unifiée des annotations, des extractions et des représentations synthétiques Dire qu'un formalisme et un modèle d'analyse quelconques doivent pouvoir opérer depuis une perspective quelconque à une échelle quelconque (*cf.* **P5**), c'est dire aussi que des enchaînements d'analyse doivent pouvoir être réalisés dans un ordre quelconque, que nous ne pouvons pas nous référer à des ordres classiques pour fixer les entrées/sorties de tel ou tel composant. Cela impose au contraire que les entrées/sorties de chaque niveau d'analyse soient encodées dans un modèle unifié, pouvant être produit et consommé à n'importe quel moment du processus d'analyse. En réponse au principe d'abstraction progressive et de variabilité des perspectives (*cf.* **P4**), chaque moment de l'analyse dédié au repérage d'occurrences en *close reading* (*cf.* **P3**) portera donc sur les annotations produites en amont, qui localisent dans les données initiales (*cf.* **P2**) et caractérisent les objets déjà reconnus, et produira de nouvelles annotations utilisables en aval, les unes et les autres étant représentées de manière homogène. Les extractions et représentations synthétiques élaborées en *distant reading* (*cf.* **P3**), elles aussi représentées de manière unifiée, s'appuieront elles aussi sur les annotations disponibles en amont (pour une perspective donnée) et les représentations synthétiques déjà établies, et seront elles-mêmes réutilisables en aval.

P7 - Déclarativité ciblée Les bonnes propriétés de la déclarativité pour la formalisation, l'étude et la capitalisation des règles d'analyse sont bien connues et nous la préconisons sans réserve pour l'ensemble des formalismes dédiés à la description des règles d'analyse (*cf.* **P5**). Le fait que l'approche déclarative masque intentionnellement les appareils procéduraux sous-jacents, pour l'application des règles, impose néanmoins clairement une concertation entre les corps de métier (*cf.* **P1**), ne serait-ce que pour que les conséquences algorithmiques restent sous contrôle. Au-delà de ce paramétrage des analyseurs, pour lequel la déclarativité doit être privilégiée, l'articulation de l'ensemble des traitements, dans des processus de type *pipeline* ou plus itératifs, pourra au contraire tirer bénéfice de l'adoption d'un paradigme plus impératif. En effet, pour le pilotage de la lecture des données d'entrée, pour la configuration des sorties, pour la gestion de flots d'exécution non strictement séquentiels, on tirera avantage du passage par un langage de programmation pour l'articulation des différentes phases d'analyse. Le recours à un langage d'intégration aura aussi l'avantage de simplifier l'utilisation combinée de librairies variées, pourvu que le langage retenu y donne effectivement accès.

P8 - Traces expérimentales La satisfaction des contraintes liées à la progression expérimentale, à la reproductibilité et à la réfutabilité impose évidemment pour commencer la disponibilité des données d'entrée et de sortie, qui doivent en conséquence être représentées dans des formats ouverts et documentés. La consultation des sorties, évidemment indispensable à l'évaluation du traitement mis en place, doit être soutenue par des outils de visualisation appropriés n'imposant idéalement ni l'installation d'un quelconque environnement logiciel complexe, ni la réitération de l'ensemble des

calculs ayant permis leur production. La mise en évidence du paramétrage du processus d'analyse bénéficiera d'abord du respect de l'exigence de déclarativité (*cf.* **P7**). Si l'articulation des différents traitements est pour sa part prise en charge programmatiquement, une API parfaitement claire devra être proposée. Toute expérimentation entreprise dans ce cadre devra pouvoir en conséquence produire une trace du cheminement suivi par le chercheur, trace où seront présentés de manière articulée les données d'entrée, les paramètres d'enchaînement, les paramètres d'analyse et les sorties, ainsi que, bien entendu, la justification des choix et l'interprétation des résultats.

3 Implémentation des ces principes dans la librairie Skhólion

Nous avons fait le choix ici de nous concentrer sur la présentation des principes qui nous semblent devoir être suivis pour la pratique expérimentale des sciences du texte. Pour rendre l'énoncé de ces principes plus concret et les illustrer, nous voudrions évoquer succinctement leur mise en œuvre au sein de la librairie Python Skhólion [5] que nous élaborons actuellement. Quelques illustrations de cette mise en œuvre sont données en annexes de cet article.

Dans l'esprit du principe **P1**, cette librairie vise à permettre la construction collective de dispositifs expérimentaux pour les sciences du texte. Elle doit permettre la collaboration efficace entre les différents acteurs de ces sciences et notamment entre le développeur et le spécialiste du texte, plutôt que de laisser à ce dernier le soin d'exploiter solitairement un environnement intégré puissant mais difficile à contrôler. En particulier, le principe selon lequel il plus aisé de vérifier la conformité d'un code donné à un problème posé, que d'élaborer *ex nihilo* la méthode de résolution, doit ici s'appliquer.

Conformément au principe **P2**, Skhólion opère sur des données textuelles d'entrée semi-structurées XML [6], auxquelles toutes les représentations produites feront référence [7], soit par la combinaison d'une expression XPath et de l'indication de la position de l'objet visé dans le nœud ciblé par l'expression, soit par référence à d'autres objets ainsi positionnés. En sortie, des données annotées sont produites par enrichissement des représentations initiales. En cours de traitement, des représentations variées sont utilisables, mais il est systématiquement possible de connaître l'ancrage des objets manipulés dans les données de référence.

La complémentarité des altitudes d'observation évoquée en **P3** est assumée, d'une part par la disponibilité de modèles d'analyse pour l'identification d'occurrences de phénomènes décrits dans différents formalismes, et d'autre part par la représentation des données, sans perte de leur ancrage, dans des structures adaptées à l'analyse de données. Nous nous appuyons notamment sur des *DataFrames* de la librairie Pandas (McKinney, 2010), qui permettent un accès direct aux outils puissants de cette librairie et des librairies sous-jacentes (en particulier NumPy (Harris *et al.*, 2020)), tout en permettant un pont vers les méthodes alimentées par des représentations tabulaires et vectorielles.

L'abstraction progressive des formes de surface du principe **P4** passe d'abord par la représentation des structures présentes dans les données initiales (notamment les structures typo-dispositionnelles). Toute unité porteuse de texte peut être segmentée en phrases, tokenisée et POS-taggée (un pont avec le Treetagger (Schmid, 1994) est assuré par défaut). L'ensemble des objets résultant peut être parcouru de différentes manières et utilisé pour produire des annotations, dont chacune, positionnée par rapport aux données de référence ou par rapport à des objets ainsi positionnés, est dotée d'une représentation

5. https://www.skholion.org
6. La librairie `lxml` (https://lxml.de) est largement utilisée.
7. La structuration initiale peut-être minimale, limitée par exemple à une décomposition en sections et paragraphes.

symbolique, sous la forme d'une structure de traits récursive. Les annotations produites pourront être exploitées par des analyseurs de plus haut niveau, qui pourront s'exprimer sur le texte qu'elles couvrent ou sur les représentations symboliques qu'elles portent, conformément à **P6**.

Conformément à **P7**, la manipulation du corpus, l'articulation des traitements et le paramétrage des sorties sont assurés de manière impérative, en Python, via l'API de Skhólion. En plus des analyseurs pouvant être directement écrits en Python, sur la base de cette API, pour le parcours des données et annotations disponibles, les modèles d'analyse proposés dans l'esprit de **P5**, **P6** et **P7**, encore en petit nombre pour le moment, permettent : 1) de construire une annotation correspondant à une structure présente dans les données d'entrée ; 2) de positionner librement une annotation dans le *continuum* textuel de toute unité disponible ; 3) de produire une annotation sur la base d'expressions régulières classiques sur la séquence de caractères de toute unité textuelle disponible et 4) d'utiliser la puissance des expressions régulières sur une séquence d'annotations produites en amont, en exprimant des contraintes sur les structures de traits associées, pour générer de nouvelles annotations. Ce premier ensemble a évidemment vocation à être étendu, conformément à **P5**, l'API de Skhólion devant simplifier l'intégration d'autres moyens d'analyse.

La démarche expérimentale est soutenue, dans l'esprit de **P8**, par l'utilisation systématique de formats ouverts pour la représentation des données, XML pour la représentation des données semi-structurées à dominante textuelle, JSON pour les autres données. En sortie de tout traitement, et notamment pour garantir un retour au texte conforme à **P3**, des représentations exploitant systématiquement les technologies du web (HTML, CSS, SVG et JavaScript) sont produites, qui peuvent être aisément consultées dans un navigateur, capitalisées et diffusées, sans aucune autre dépendance logicielle. Des visualisations sont proposées à l'échelle d'un texte ou à l'échelle du corpus, et des extractions d'objets ou de passages choisis sont aussi possibles. S'il est évidemment envisageable de travailler avec Skhólion dans un environnement de développement traditionnel, nous veillons à ce que l'utilisation de l'ensemble des outils proposés soit possible dans des Notebooks Jupyter (Kluyver *et al.*, 2016) et dans des environnements comme Jupyterlab, y compris pour la visualisation des résultats. De tels environnements, qui constituent le cadre privilégié d'utilisation de Skhólion, permettent de produire une trace du cheminement expérimental facile à capitaliser et à partager.

4 Conclusion

L'enthousiasme suscité dans notre communauté par les méthodes d'apprentissage et les LLM illustre parfaitement les potentialités et les risques où cet article trouve sa source. Car si la richesse des faits de langue que ces méthodes permettent de capturer justifie pleinement l'intérêt qu'on leur accorde, elles montrent aussi que bien des applications sont rendues possibles, qui ne mettent pas en pleine lumière les phénomènes linguistiques sous-jacents qu'elles exploitent. À travers ce plaidoyer pour les sciences du texte, nous voulions d'abord insister sur la nécessité de maintenir, au-delà de la réponse à des impératifs applicatifs (qui font souvent du texte un moyen), l'exigence de compréhension fine des phénomènes de langue (qui prend le texte pour fin). Les outils puissants auxquels l'ingénierie du texte a donné naissance peuvent aussi bien entendu contribuer à sa compréhension, pourvu que leur utilisation s'intègre dans un cadre expérimental dont cette compréhension est l'objectif clair. C'est ce cadre expérimental dont nous espérons pouvoir contribuer modestement à dessiner les contours, par la mise en lumière de principes pouvant guider sa mise en place et leur illustration dans une librairie naissante dédiée à l'expérimentation sur corpus, Skhólion.

Références

ARTSTEIN R. & POESIO M. (2008). Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, **34**(4), 555–596. DOI : http ://dx.doi.org/10.1162/coli.07-034-R2.

BIRD S., KLEIN E. & LOPER E. (2009). *Natural language processing with Python : analyzing text with the natural language toolkit*. O'Reilly.

BURNARD L., O'KEEFE K. O. & UNSWORTH J., Éds. (2006). *Electronic Textual Editing*. Modern Language Association.

CUNNINGHAM H., MAYNARD D., BONTCHEVA K. & TABLAN V. (2002). GATE : A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*.

CUNNINGHAM H., MAYNARD D., BONTCHEVA K., TABLAN V., ASWANI N., ROBERTS I., GORRELL G., FUNK A., ROBERTS A., DAMLJANOVIC D., HEITZ T., GREENWOOD M. A., SAGGION H., PETRAK J., LI Y. & PETERS W. (2011). *Text Processing with GATE (Version 6)*.

CUNNINGHAM H., TABLAN V., ROBERTS A. & BONTCHEVA K. (2013). Getting More Out of Biomedical Documents with GATE's Full Lifecycle Open Source Text Analytics. *PLoS Computational Biology*, **9**(2), e1002854. DOI : 10.1371/journal.pcbi.1002854.

ENJALBERT P., BONTCHEVA K. & HABERT B., Éds. (2008). *Plate-formes pour le traitement automatique des langues*. Volume 49(2) de Revue TAL. France : ATALA (Association pour le Traitement Automatique des Langues).

FORT K. (2016). *Collaborative Annotation for Reliable Natural Language Processing : Technical and Sociological Aspects*. Wiley.

HARRIS C. R., MILLMAN K. J., WALT S. J. V. D., GOMMERS R., VIRTANEN P., COURNAPEAU D., WIESER E., TAYLOR J., BERG S., SMITH N. J., KERN R., PICUS M., HOYER S., KERKWIJK M. H. V., BRETT M., HALDANE A., RÍO J. F. D., WIEBE M., PETERSON P., GÉRARD-MARCHANT P., SHEPPARD K., REDDY T., WECKESSER W., ABBASI H., GOHLKE C. & OLIPHANT T. E. (2020). Array programming with NumPy. *Nature*, **585**(7825), 357–362. Publisher : Springer Science and Business Media LLC, DOI : 10.1038/s41586-020-2649-2.

HEIDEN S. (2010). The TXM Platform : Building Open-Source Textual Analysis Software Compatible with the TEI Encoding Scheme. In R. OTOGURO, K. ISHIKAWA, H. UMEMOTO, K. YOSHIMOTO & Y. HARADA, Éds., *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation*, p. 389–398, Sendai, Japon : Institute for Digital Enhancement of Cognitive Development, Waseda University.

IDE N. & VÉRONIS J., Éds. (1995). *Text Encoding Initiative : Background and Context*. Text, Speech and Language Technology. Dordrecht : Kluwer.

KLIE J.-C., BUGERT M., BOULLOSA B., CASTILHO R. E. D. & GUREVYCH I. (2018). The INCEpTION Platform : Machine-Assisted and Knowledge-Oriented Interactive Annotation. In *Proceedings of the 27th International Conference on Computational Linguistics : System Demonstrations*, p. 5–9 : Association for Computational Linguistics.

KLUYVER T., RAGAN-KELLEY B., PÉREZ F., GRANGER B., BUSSONNIER M., FREDERIC J., KELLEY K., HAMRICK J., GROUT J., CORLAY S., IVANOV P., AVILA D., ABDALLA S. & WILLING C. (2016). Jupyter Notebooks – a publishing format for reproducible computational workflows. In F. LOIZIDES & B. SCHMIDT, Éds., *Positioning and Power in Academic Publishing : Players, Agents and Agendas*, p. 87 – 90 : IOS Press.

LAFON P. (1984). *Dépouillements et statistiques en lexicométrie*. Volume 24 de Travaux de linguistique quantitative. Genève : Paris : Slatkine ; Champion.

LANDRAGIN F., POIBEAU T. & VICTORRI B. (2012). ANALEC : a New Tool for the Dynamic Annotation of Textual Data. In *Proceedings of Language Resources and Evaluation (LREC 2012)*, p. 357–362, Istanbul, Turkey.

LEBART L., SALEM A. & BERRY L. (1998). *Exploring Textual Data*. Text, speech, and language technology. Kluwer Academic.

MCKINNEY W. (2010). Data Structures for Statistical Computing in Python. p. 56–61, Austin, Texas. DOI : 10.25080/Majora-92bf1922-00a.

MELLET S. (2002). Corpus et recherches linguistiques : Introduction. *Corpus*, (1). DOI : 10.4000/corpus.7.

MORETTI F. (2013). *Distant reading*. London ; New York : Verso.

PAUMIER S., GUENTHNER F., LAPORTE E., MALCHOK F., MARSCHNER C., MARTINEAU C., MARTÍNEZ C., MAUREL D., NAGEL S., NEME A., PETIT M., STIEHLER J. & VOLLANT G. (2021). UNITEX 3.3 Manuel d'utilisation.

PEDREGOSA F., VAROQUAUX G., GRAMFORT A., MICHEL V., THIRION B., GRISEL O., BLONDEL M., PRETTENHOFER P., WEISS R., DUBOURG V., VANDERPLAS J., PASSOS A., COURNAPEAU D., BRUCHER M., PERROT M. & DUCHESNAY E. (2011). Scikit-learn : Machine Learning in Python. *Journal of Machine Learning Research*, **12**, 2825–2830.

RASTIER F. (2001). *Arts et sciences du texte*. Formes sémiotiques. Paris : Presses universitaires de France, 1re édition.

SCHMID H. (1994). Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of the Conference on New Methods in Language Processing*, Manchester, UK.

SILBERZTEIN M. (2016). *Formalizing natural languages : the NooJ approach*. Collection Science cognitive et management des connaissances. London Hoboken : ISTE John Wiley and Sons.

STENETORP P., PYYSALO S., TOPIĆ G., OHTA T., ANANIADOU S. & TSUJII J. (2012). BRAT : a Web-based Tool for NLP-Assisted Text Annotation. In F. SEGOND, Éd., *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, p. 102–107, Avignon, France : Association for Computational Linguistics.

WIDLÖCHER A. & BILHAUT F. (2008). Articulation des traitements en TAL. Principes méthodologiques et mise en œuvre dans la plate-forme LinguaStream. *Revue TAL (Traitement Automatique des Langues)*, **49**(2), 73–101. Place : France Publisher : ATALA (Association pour le Traitement Automatique des Langues).

WIDLÖCHER A. & MATHET Y. (2012). The Glozz Platform : a Corpus Annotation and Mining Tool. In C. CONCOLATO & P. SCHMITZ, Éds., *ACM Symposium on Document Engineering (DocEng'12)*, p. 171–180, Paris, France : ACM.

Annexes - Quelques illustrations du cadre proposé par Skhólion

Les exemples fournis ci-après ont pour unique objectif de donner une idée des informations accessibles depuis l'API et de la relative simplicité de mise en œuvre des traitements et des outils de visualisation proposés. Les algorithmes présentés ne sont pas toujours optimaux mais permettent d'illustrer en particulier l'exploitation des niveaux de segmentation et la possiblité de s'appuyer sur les annotations antérieurement produites.

```python
#
# Affichage simple d'un texte (CorpusItem) issu du corpus.
#

from skholion.corpus.map import CorpusMap
from skholion.xxml.navigator import Navigator
from skholion.gui.browser import Browser

corpus_map = CorpusMap("./corpus_data/")
fortune = corpus_map.get_corpus_item_by_key_name("Zola_La_Fortune_des_Rougon")

output_dir = "navigator/"
navigator_dir = output_dir + fortune.get_file_name_prefix()
navigator = Navigator(xml_corpus_item_source_path=fortune.get_html_quick_view_full_path(),
                      navigator_dir_output_path=navigator_dir)
navigator.make_all()

browser = Browser()
browser.open_local_path(navigator.get_main_file_path())
```

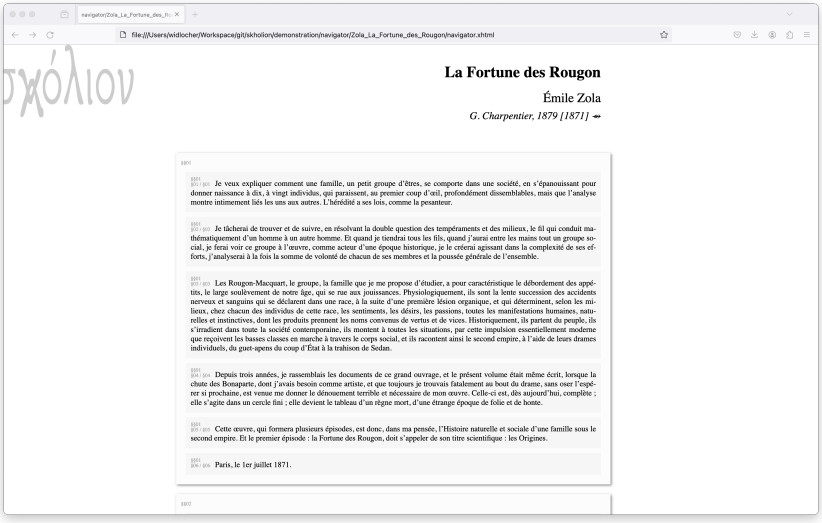

FIGURE 2 – Visualisation simple d'un item de corpus dans un navigateur web

```python
#
# Annotation par expressions régulières simples appliquées au contenu textuel d'un unique
# CorpusItem, puis affichage du texte annoté et des représentations symboliques associées.
#

from skholion.corpus.map import CorpusMap
from skholion.corpus.quickview import CorpusItemQuickView
from skholion.gui.color import ColorManager
from skholion.xxml.annotation.offset import OffsetAnnotator, OffsetAnnotation
from skholion.metamodel.characterization import Characterization
from skholion.xxml.navigator import Navigator
from skholion.gui.browser import Browser

import re

corpus_map = CorpusMap("./corpus_data/")
fortune = corpus_map.get_corpus_item_by_key_name("Zola_La_Fortune_des_Rougon")
fortune_text = fortune.get_anchored_text()
fortune_text_content = fortune_text.get_content()

offset_annotator = OffsetAnnotator(fortune.get_html_quick_view_full_path(),
                                   CorpusItemQuickView.CHARACTERIZATIONS_INJECTION_XPATH)

color_manager = ColorManager()

searched_items = ["(Rougon)", "(Macquart)", "(Lantier)", "(Mouret)", "(Saccard)", "(Coupeau)", "(Quenu)"]
for si in searched_items :
    patron = re.compile(si)
    for occurrence_position, match in enumerate(patron.finditer(fortune_text_content)):
        start_offset, end_offset = match.span()
        color = color_manager.get_color(match.group(1))
        characterization = Characterization("Famille", {"forme": match.group(1),
                                                        "occurrence" : str(occurrence_position+1),
                                                        "contexte":{"gauche":fortune_text_content[start_offset-100:start_offset],
                                                                    "droite":fortune_text_content[end_offset:end_offset+100]}})
        annotation = OffsetAnnotation(annotation_context_anchored_item=fortune_text,
                                      annotation_start_offset_in_context=start_offset,
                                      annotation_end_offset_in_context=end_offset,
                                      annotation_characterization=characterization,
                                      annotation_xml_type="span",
                                      annotation_xml_attributes={"style":"background-color:%s;" % color},
                                      annotation_text_content=match.group(1))
        offset_annotator.add_annotation(annotation)

offset_annotator.annotate()
output_xhtml_path = "/tmp/corpus_item.tmp.xhtml"
offset_annotator.dump(output_xhtml_path)

color_map_path = "/tmp/color_map.tmp.xhtml"
color_manager.write_html_color_map_file(output_path=color_map_path)

output_dir = "navigator/"
navigator_dir = output_dir + fortune.get_file_name_prefix()
navigator = Navigator(xml_corpus_item_source_path=output_xhtml_path,
                      offset_annotation_set=offset_annotator.offset_annotations_inserted_with_success,
                      color_map_source_path=color_map_path,
                      navigator_dir_output_path=navigator_dir)
navigator.make_all()

browser = Browser()
browser.open_local_path(navigator.get_main_file_path())
```

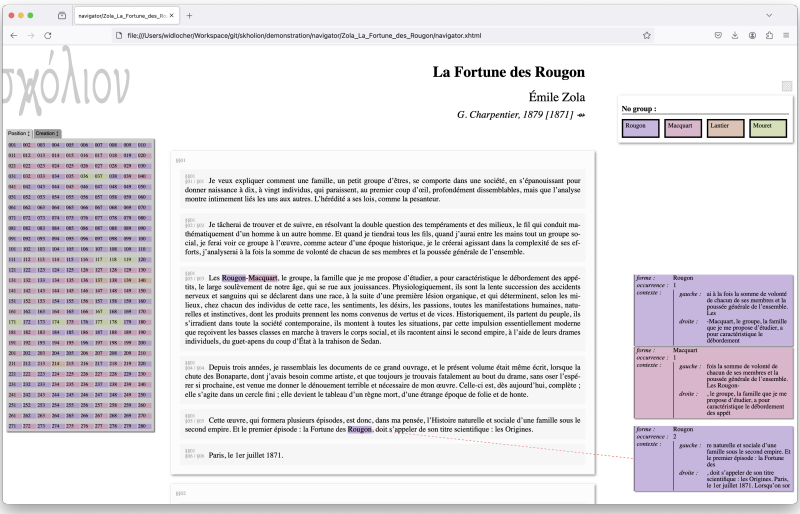

FIGURE 3 – Annotation par expressions régulières au niveau caractère, puis visualisation des annotations et des structures de traits associées

```python
#
# Annotation par expressions régulières simples appliquées au contenu textuel des phrases
# d'un ensemble de Corpusitems, puis affichage à l'aide d'un CorpusNavigator permettant
# de naviguer entre les textes.
#

from skholion.corpus.map import CorpusMap
from skholion.corpus.quickview import CorpusItemQuickView
from skholion.xxml.annotation.offset import OffsetAnnotator, OffsetAnnotation
from skholion.metamodel.characterization import Characterization
from skholion.xxml.navigator import Navigator, CorpusNavigator
from skholion.gui.color import ColorManager
from skholion.gui.browser import Browser

import re

corpus_map = CorpusMap("./corpus_data/")
germinal = corpus_map.get_corpus_item_by_key_name("Zola_Germinal")
fortune = corpus_map.get_corpus_item_by_key_name("Zola_La_Fortune_des_Rougon")
argent = corpus_map.get_corpus_item_by_key_name("Zola_L_Argent")
corpus = [germinal, fortune, argent]

corpus_level_color_manager = ColorManager()

for corpus_item in corpus :

    corpus_item_level_color_manager = ColorManager()
    text = corpus_item.get_sentences_segmented_anchored_text()
    offset_annotator = OffsetAnnotator(corpus_item.get_html_quick_view_full_path(),
                                       CorpusItemQuickView.CHARACTERIZATIONS_INJECTION_XPATH)

    for section in text:
        for paragraph in section:
            for sentence in paragraph:
                sentence_content = sentence.get_content()
                searched_items = ["(Rougon)", "(Macquart)", "(Lantier)", "(Mouret)", "(Saccard)", "(Coupeau)", "(Quenu)"]
                for si in searched_items :
                    patron = re.compile(si)
                    for match in patron.finditer(sentence_content):
                        color = corpus_level_color_manager.get_color(match.group(1))
                        start_offset, end_offset = match.span()
                        characterization = Characterization("Famille", {"forme": match.group(1), "phrase" : sentence_content})
                        annotation = OffsetAnnotation(annotation_context_anchored_item=sentence,
                                                      annotation_start_offset_in_context=start_offset,
                                                      annotation_end_offset_in_context=end_offset,
                                                      annotation_characterization=characterization,
                                                      annotation_xml_type="span",
                                                      annotation_xml_attributes={"style":"background-color:%s;" % color},
                                                      annotation_text_content=match.group(1))
                        offset_annotator.add_annotation(annotation)

                searched_items = ["(hériter)", "(hérédit\\w)", "(famille)", "(file)", "(fils)", "(mère)", "(père)"]
                for si in searched_items :
                    patron = re.compile(si)
                    for match in patron.finditer(sentence_content):
                        color = corpus_item_level_color_manager.get_color(match.group(1), group_key="Parenté")
                        start_offset, end_offset = match.span()
                        characterization = Characterization("Famille", {"forme": match.group(1), "motif" : si})
                        annotation = OffsetAnnotation(annotation_context_anchored_item=sentence,
                                                      annotation_start_offset_in_context=start_offset,
                                                      annotation_end_offset_in_context=end_offset,
                                                      annotation_characterization=characterization,
                                                      annotation_xml_type="span",
                                                      annotation_xml_attributes={"style":"background-color:%s;" % color},
                                                      annotation_text_content=match.group(1))
                        offset_annotator.add_annotation(annotation)

    offset_annotator.annotate()
    output_xhtml_path = "/tmp/corpus_item.tmp.xhtml"
    offset_annotator.dump(output_xhtml_path)

    corpus_item_level_color_map_path = "/tmp/color_map.tmp.xhtml"
    corpus_item_level_color_manager.write_html_color_map_file(output_path=corpus_item_level_color_map_path)

    output_dir = "navigator/"

    navigator_dir = output_dir + corpus_item.get_file_name_prefix()
    navigator = Navigator(xml_corpus_item_source_path=output_xhtml_path,
                          offset_annotation_set=offset_annotator.offset_annotations_inserted_with_success,
                          color_map_source_path=corpus_item_level_color_map_path,
                          navigator_dir_output_path=navigator_dir)
    navigator.make_all()

corpus_level_color_map_path = "/tmp/color_map.tmp.xhtml"
corpus_level_color_manager.write_html_color_map_file(output_path=corpus_level_color_map_path)
corpus_navigator = CorpusNavigator(navigator_source_and_output_path=output_dir,
                                   color_map_source_path=corpus_level_color_map_path)
corpus_navigator.make_all()

browser = Browser()
browser.open_local_path(corpus_navigator.get_main_file_path())
```

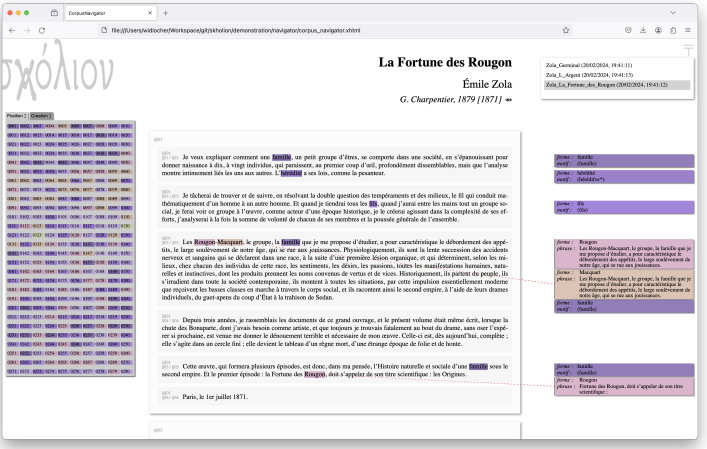

FIGURE 4 – Application d'expressions régulières au contenu textuel des phrases d'un corpus composé de plusieurs textes

```python
#
# On annote tous les verbes à l'indicatif, puis on annote les séquences ininterrompues
# de verbes au présent.
#

from skholion.corpus.map import CorpusMap
from skholion.corpus.quickview import CorpusItemQuickView
from skholion.metamodel.characterization import Characterization
from skholion.xxml.annotation.offset import OffsetAnnotator
from skholion.linguistics.partofspeech import PartOfSpeech, VerbalPartOfSpeech
from skholion.analysis.anchoreditemannotator import AnchoredItemAnnotator
from skholion.analysis.regexoa import RegexOnAnnotationsSolver
from skholion.analysis.unitgrouper import UnitGrouper
from skholion.gui.color import ColorManager
from skholion.xxml.navigator import Navigator
from skholion.gui.browser import Browser

corpus_map = CorpusMap("./corpus_data/")
corpus_item = corpus_map.get_corpus_item_by_key_name("Zola_La_Fortune_des_Rougon")
text = corpus_item.get_sentences_and_tokens_segmented_pos_tagged_anchored_text()
quick_view_html_path = corpus_item.get_html_quick_view_full_path()

offset_annotator = OffsetAnnotator(quick_view_html_path, CorpusItemQuickView.CHARACTERIZATIONS_INJECTION_XPATH,
                                   characterization_injection_mode=OffsetAnnotator.CHARACTERIZATION_INJECTION_MODE_COMPACT)

color_manager = ColorManager()
for section in text :
    for paragraph in section :
        for sentence in paragraph :
            for token in sentence :
                if (token.part_of_speech[PartOfSpeech.FEATURE_NAME_TAG]==PartOfSpeech.TAG_VERB
                and token.part_of_speech[VerbalPartOfSpeech.FEATURE_NAME_MOOD]==VerbalPartOfSpeech.MOOD_INDICATIVE):
                    main_tense = token.part_of_speech[VerbalPartOfSpeech.FEATURE_NAME_MAIN_TENSE]
                    characterization = Characterization("Verbe", {"type":"verb", "content":token.content, "main-tense": main_tense})
                    color = color_manager.get_color(main_tense)
                    token_annotation = AnchoredItemAnnotator.get_unit_from_anchored_item(anchored_item=token,
                                                                                         annotation_characterization=characterization,
                                                                                         annotation_xml_attributes={"style":f"background-color:{color};"})
                    offset_annotator.add_annotation(token_annotation)

roas = RegexOnAnnotationsSolver(offset_annotator.offset_annotations)
annotations_input_sequence = roas.prepare_annotation_set()
pattern = "Annotation:{(@/main-tense:%s)}+" % VerbalPartOfSpeech.MAIN_TENSE_PRESENT
pattern = roas.prepare_pattern(pattern)
matches = roas.find_all(pattern, annotations_input_sequence)

color = color_manager.get_color("present-sequence")
for match in matches :
    unit_1, unit_2, match_characterization = match
    characterization = Characterization("Present-sequence", {"type":"present-sequence"})
    new_annotation = UnitGrouper.get_unit_from_unit_to_unit(unit_1=unit_1,
                                                            unit_2=unit_2,
                                                            annotation_characterization=characterization,
                                                            annotation_xml_attributes={"style":f"background-color:{color};padding:2px;"})
    offset_annotator.add_annotation(new_annotation)

offset_annotator.annotate()

output_xhtml_path = "/tmp/corpus_item.tmp.xhtml"
offset_annotator.dump(output_xhtml_path)
color_map_path = "/tmp/color_map.tmp.xhtml"
color_manager.write_html_color_map_file(output_path=color_map_path)
output_dir = "navigator/"
navigator_dir = output_dir + corpus_item.get_file_name_prefix()
navigator = Navigator(xml_corpus_item_source_path=output_xhtml_path,
                      offset_annotation_set=offset_annotator.offset_annotations_inserted_with_success,
                      color_map_source_path=color_map_path,
                      navigator_dir_output_path=navigator_dir)
navigator.make_all()
browser = Browser()
browser.open_local_path(navigator.get_main_file_path())
```

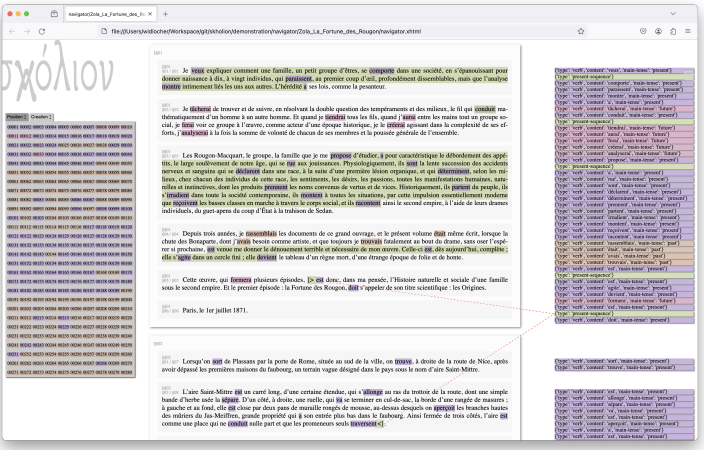

FIGURE 5 – Annotation de tokens POS-taggés, puis application de motifs REGEXOA (*REGEX On Annotations*) pour l'annotation d'une séquence d'annotations de plus bas niveau

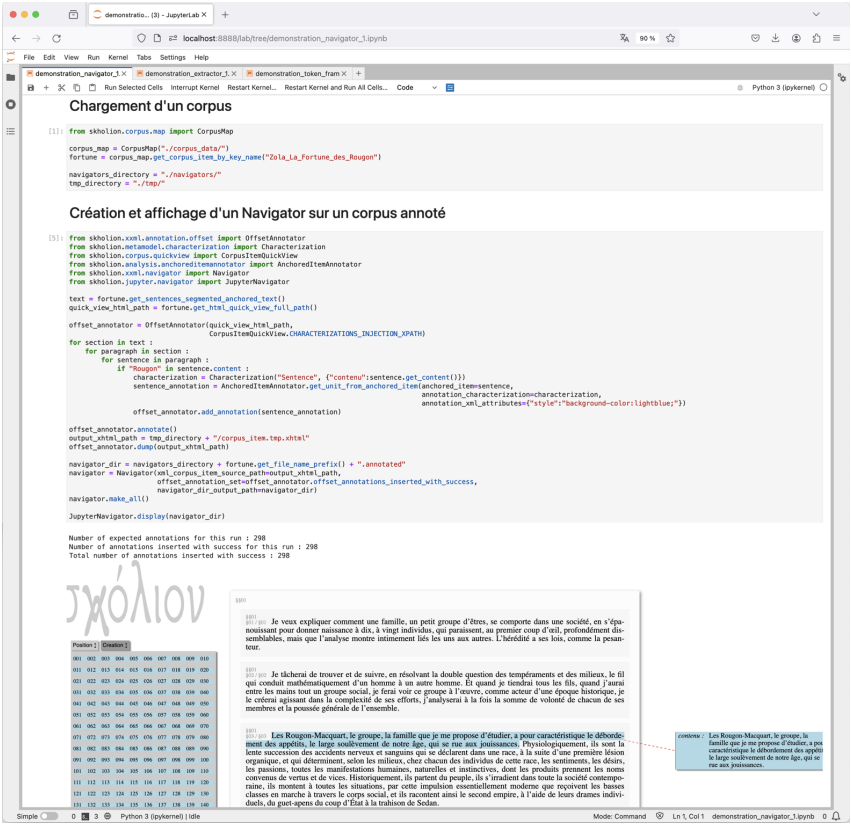

FIGURE 6 – Annotation de phrases et visualisation du texte annoté dans JupyterLab

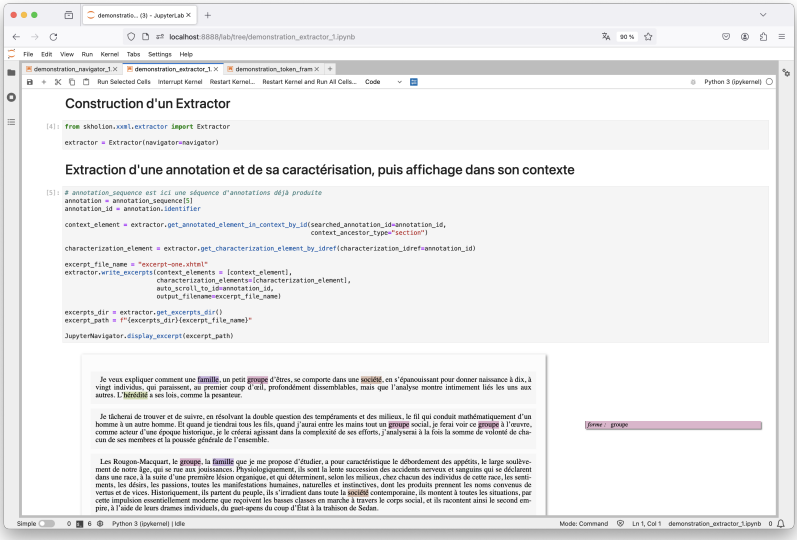

FIGURE 7 – Extraction d'une annotation et visualisation dans son contexte

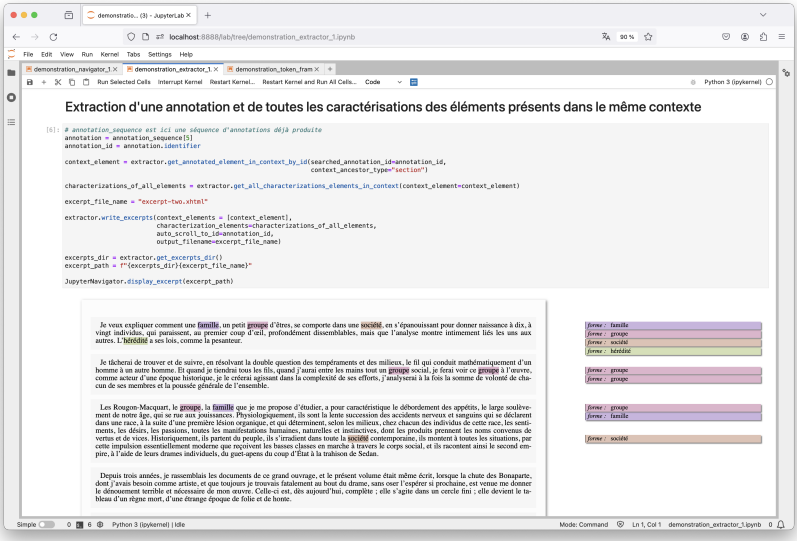

FIGURE 8 – Extraction d'une annotation et visualisation dans son contexte, en intégrant les descriptions des autres objets présents dans ce contexte

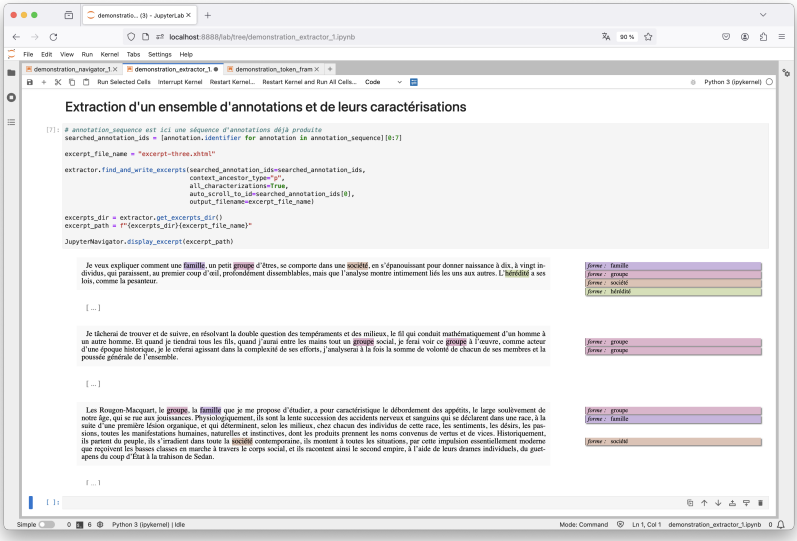

FIGURE 9 – Extraction d'une sélection d'annotations

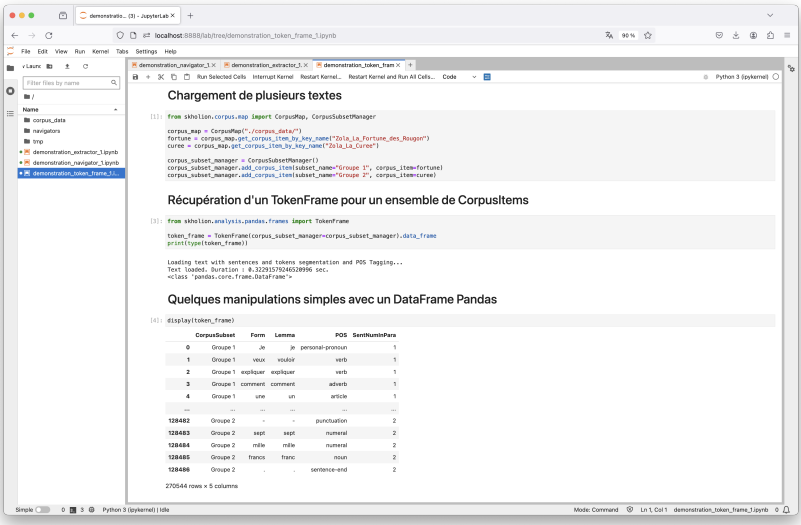

FIGURE 10 – Récupération d'un TokenFrame et manipulation du DataFrame Pandas sous-jacent

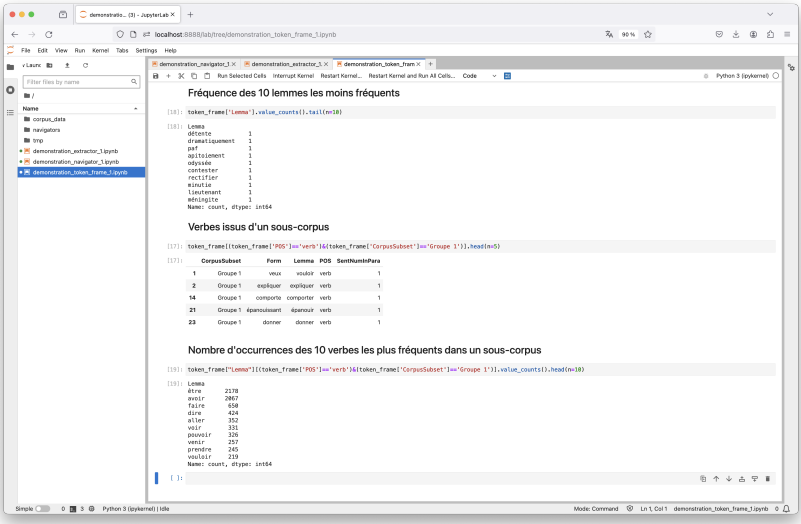

FIGURE 11 – Exemples de manipulation d'un DataFrame avec Pandas