# Chapitre 1

# Annotation des sections et entités juridiques

*Ce chapitre traite de la détection de sections et des mentions (occurrences) d'entités dans les décisions jurisprudentielles françaises. Ce problème est important car il vise une structuration des documents par le balisage des sections organisant le document, des méta-données de référence de l'affaire, et des citations des normes juridiques employées. L'utilité pour le métier de juriste est améliorée de plusieurs façons. L'identification des méta-données d'indexation des décisions est automatisée. La lecture des décisions et leur compréhension manuelles ou automatiques est facilitée. Aussi, en complétant le problème par la désambigüisation des méta-données et des normes, les critères de recherche de décisions peuvent être enrichis. Les principales contributions du travail discuté ici sont : la constitution et l'annotation manuelles d'un corpus d'évaluation, l'expérimentation et la discussion de l'impact, sur le problème, de divers aspects de la conception d'un système d'extraction d'information par étiquetage de séquence (l'ingénierie manuelle et automatique des caractéristiques des objets à classer, le choix du schéma d'étiquetage, l'augmentation des données d'entrainement). Les résultats montrent l'efficacité de modèle à base de champs aléatoires conditionnels à chaîne linéaire (CRF) pour les différentes tâches.*

## 1.1 Introduction

Bien que les décisions ne soient pas structurées, leur contenu est organisé en sections dont les principales sont : l'entête, le corps, et le dispositif. Chacune de ces sections décrit des informations spécifiques de l'affaire :

- l'entête contient de nombreuses méta-données de référence comme la date, le lieu, les participants, etc.
- le corps détaille les faits, les procédures antérieures, les conclusions des parties et le raisonnement des juges ;

- le dispositif est la synthèse du résultat final c'est-à-dire qu'on y retrouve les réponses aux demandes des parties.

Certaines informations spécifiques se retrouvent très souvent dans une même section, e.g. méta-données (localisation, date), prétentions des parties, décisions finales. Compte tenu de la répartition standard de certaines informations, certaines tâches d'extraction d'information peuvent être abordées comme des traitements spécifiques à appliquer à certaines sections. Ce chapitre traite dans un premier temps des modèles utilisés pour appliquer cette phase de segmentation des décisions en sections. Par la suite, les entités, et données sur les demandes et résultats, pourront plus facilement être extraites. Nous nous focaliserons en particulier ici sur la détection des mentions d'entités telles que la date à laquelle le jugement a été prononcé, le type de juridiction, sa localisation (ville), les noms des juges, des parties, et les règles de loi citées (normes). La Table 1.1 liste les différentes entités cibles et fournit des exemples illustrant leurs occurrences dans les décisions avec lesquelles nous avons travaillé.

On pourrait s'attendre à ce qu'une institution comme la justice respecte un modèle strict et commun à tous les tribunaux pour la rédaction des décisions pour permettre de facilement pouvoir les lire et les analyser. Malheureusement, même si les décisions décrivent des informations de même nature, le modèle employé semble varier entre les juridictions. C'est ce que l'on remarque déjà au niveau de la transition entre sections. Au vu de leur rôle, il est évident que les sections devraient être séparées par des marqueurs bien précis. Une approche intuitive de sectionnement consisterait par conséquent à définir un algorithme capable de reconnaître automatiquement ces marqueurs de transition par l'utilisation d'expressions régulières. Cependant, les marqueurs retrouvés ne sont généralement pas standards. Les indicateurs de transitions sont en effet souvent différents d'une décision à l'autre ; ils peuvent correspondre à des titres ou des motifs à base de symboles (astérisques, tirets, etc.). Il arrive même parfois que la transition soit implicite et que l'on ne s'en rende compte que par la forme ou le contenu des lignes, au cours de la lecture. Même les marqueurs explicites sont hétérogènes. Lors de l'emploi de titres par exemple, la transition de l'entête à l'exposé du litige peut être indiquée par des titres comme « Exposé », « FAITS ET PROCÉDURES », « Exposé de l'affaire », « Exposé des faits », etc. Quant au dispositif, il est introduit généralement par l'expression « PAR CES MOTIFS » avec souvent quelques

| Entités | Label | Exemples | #mentions[a] Médiane[b] | Total[c] |
|---|---|---|---|---|
| Numéro de registre général (R.G.) | **rg** | « 10/02324 », « 60/JAF/09 » | 3 | 1318 |
| Ville | **ville** | « NÎMES », « Agen », « Toulouse » | 3 | 1304 |
| Juridiction | **juridiction** | « COUR D'APPEL » | 3 | 1308 |
| Formation | **formation** | « 1re chambre », « Chambre économique » | 2 | 1245 |
| Date de prononcé | **date** | « 01 MARS 2012 », « 15/04/2014 » | 3 | 1590 |
| Appelant | **appelant** | « SARL K. », « Syndicat ... », « Mme X ... » | 2 | 1336 |
| Intimé | **intime** | - // - | 3 | 1933 |
| Intervenant | **intervenant** | - // - | 0 | 51 |
| Avocat | **avocat** | « Me Dominique A., avocat au barreau de Papeete » | 3 | 2313 |
| Juge | **juge** | « Monsieur André R. », « Mme BOUSQUEL » | 4 | 2089 |
| Fonction de juge | **fonction** | « Conseiller », « Président » | 4 | 2062 |
| Norme | **norme** | « l' article 700 NCPC », « articles 901 et 903 » | 12 | 7641 |
| Non-entité | **O** | *mot ne faisant partie d'aucune mention d'entité* | - | - |

[a] nombre de mentions d'entités dans le corpus annoté pour les expérimentations
[b] nombre médian de mentions par document dans le corpus annoté
[c] nombre total d'occurrences dans le corpus annoté
* Les statistiques sur les sommes d'argent ne concernent que 100 documents annotés (max=106, min=1, moyenne=17.77), contre 500 documents pour les autres entités.

Tableau 1.1 – Exemples d'entités et statistiques sur la base d'exemples annotés manuellement

variantes qui peuvent être très simples (par exemple « Par Ces Motifs ») ou exceptionnelles (par exemple « P A R C E S M O T I F S : »). Dans certaines décisions, cette expression est remplacée par d'autres expressions comme « DECISION », « DISPOSITIF », « LA COUR », etc. Par ailleurs, lors de l'utilisation de symboles, il arrive qu'un même motif sépare différentes sections et même des paragraphes dans une même section. Des différences similaires apparaissent aussi pour les entités. Les noms de parties sont généralement placés après un mot particulier comme « APPELANTS » ou « DEMANDEUR » pour les demandeurs (appelants en juridiction de 2e degré), « INTIMES » ou « DEFENDEUR » pour les défendeurs (ou intimés), et « INTERVENANTS » pour les intervenants. Les noms des individus, sociétés et lieux commencent par une lettre majuscule, et sont entièrement en majuscules. Cependant, certains mots communs peuvent apparaître aussi en majuscules (par ex. APPELANTS, DÉBATS, ORDONNANCE DE CLÔTURE). Les entités peuvent contenir des chiffres (identifiants, dates,

...), des caractères spéciaux ( « / », « - »), des initiales (par ex. « A. ») ou abréviations. Dans l'entête, les entités apparaissent généralement dans le même ordre (par ex. les appelants avant les intimés, les intimés avant les intervenants). Cependant, on rencontre une multitude de types d'entités dans l'entête, contrairement aux autres sections où seules les normes nous intéressent. De plus, le texte est mieux structuré dans l'entête que dans les autres sections. Ces nombreuses différences entre décisions rendent inadéquates les expressions régulières.

Notre étude consiste à analyser l'application du Modèle Caché de Markov (HMM) et des Champs Aléatoires Conditionnels (CRF) aux problèmes de sectionnement et reconnaissance d'entités juridiques. Ces deux tâches sont ainsi représentées sous la forme d'un problème d'étiquetage de séquences. L'idée est de découper un texte en segments atomiques distincts (*token*) qui peuvent être des mots, des phrases, des paragraphes, etc. Le texte est ainsi représenté sous forme de séquences et chaque objet d'intérêt (section ou entité) comprend un ou plusieurs segments. Un label est défini pour chaque type d'entité (par ex. `PER` pour les noms de personnes).

## 1.2 Extraction d'information par étiquetage de séquence

Parmi les approches d'extraction d'information Chau *et al.* [2002], on retrouve principalement :

- Les **systèmes à recherche lexicale** sont conçus sur la base d'une liste d'entités préalablement connues, et leurs synonymes dans le domaine d'intérêt. Par exemple, dans le domaine juridique, un lexique pourrait contenir les identifiants de règles juridiques et les noms des juges. La liste des entités peut être fournie par des experts ou apprise à partir d'un ensemble de données annotées manuellement (phase d'apprentissage). Cependant, il s'avère très difficile de maintenir une telle liste car le domaine change régulièrement (nouvelles lois par ex.). De plus, les mentions d'entités peuvent avoir plusieurs variantes. Par exemple, la même règle juridique « Article 700 du code de procédure civile » peut être citée seule et en entier (« article 700 du code de procédure civile »), ou abrégée (« article 700 CPC »), ou encore avec d'autres règles (« articles 700 et 699 du code de procédure civile »).

De plus, ces approches sont sujettes aux problèmes d'ambiguïté, par exemple lorsque différentes entités comprennent les mêmes mots. Ces problèmes ont largement limité ces premiers systèmes [Palmer & Day, 1997].

- Les **systèmes à base de règles** décrivent la variété des mentions d'entités en fonction de la régularité du contexte, de la structure et du lexique. Il existe plusieurs plate-formes et langages permettant de formaliser l'écriture des règles. Par exemple, dans le formalisme JAPE de Gate, Wyner [2010] détecte les énoncés de décisions à l'aide d'une règle qui sélectionne les phrases contenant un terme de jugement (*affirm*, *grant*, etc.) et suivies d'un nom de juge :

```
Rule: DecisionStatement
Priority: 10
(
{Sentence contains JudgementTerm}
):termtemp
{JudgeName}
->
:termtemp.DecisionStatement = {rule = ‘‘DecisionStatement’’}.
```

  Ces systèmes présentent l'avantage de reposer sur des expressions déclaratives qui facilitent la maintenance (erreurs faciles à tracer et à expliquer) et l'expression directe des connaissances du domaine en règles [Waltl *et al.* , 2018]. Bien que parfois suffisant pour traiter des corpus modestes et spécialisés, ces systèmes sont très souvent limités en pratique. La définition manuelle de règles exige notamment des efforts considérables, en particulier pour le traitement de grands corpus. Par ailleurs, un ensemble donné de règles est difficilement réutilisable dans d'autres domaines ou sur des données n'intégrant pas exactement les subtilités linguistiques exprimées par les règles. Quelques approches adaptatives ont néanmoins été conçues pour surmonter ces limites tout en bénéficiant toujours de la facilité à expliquer le comportement des systèmes à base de règles [Siniakov, 2008; Chiticariu *et al.* , 2010].

- Les **systèmes basés sur l'apprentissage automatique** exécutent des classifieurs multi-classes sur des segments de texte. Par exemple, un

algorithme traditionnel de classification comme le modèle bayésien naïf peut être entraîné pour détecter les noms de gènes en classifiant les mots d'un article scientifique [Persson, 2012]. Par ailleurs, les algorithmes d'étiquetage de séquences tels que le CRF classifient les mots tout en modélisant les transitions entre les labels [Finkel *et al.* , 2005]. Dans ce registre, les architectures d'apprentissage profond réalisent actuellement les meilleures performances sur de multiples tâches d'extraction d'information en général et de reconnaissance d'entités nommées en particulier [Lample *et al.* , 2016].

Certains travaux ont combiné différentes approches pour extraire les entités à partir de documents juridiques, par exemple, par la description de l'information contextuelle en utilisant des règles pour répondre au problème d'ambiguïté des méthodes à recherche lexicale [Mikheev *et al.* , 1999; Hanisch *et al.* , 2005]. Mais les systèmes basés sur l'apprentissage automatique sont les plus efficaces actuellement pour l'extraction d'information, en particulier les modèles graphiques probabilistes.

Trois principaux aspects doivent être traités lors de la conception des systèmes à étiquetage de séquence : la sélection du modèle d'étiquetage, l'ingénierie des caractéristiques des segments à étiqueter, et le choix d'une représentation de segment (encore appelé schéma d'étiquetage).

### 1.2.1 Les modèles graphiques probabilistes HMM et CRF

Nous avons choisi d'analyser l'application des modèles CRF et HMM car les comparaisons avec d'autres approches démontrent bien que les modèles probabilistes obtiennent les meilleurs résultats lors de l'extraction d'information dans les documents juridiques. Par exemple, dans Kríž *et al.* [2014], le modèle HMM a été comparé à l'Algorithme de Perceptron à Marges Inégales (PAUM) de Li *et al.* [2002] pour reconnaître les institutions et références d'autres décisions de justice, ainsi que les citations d'actes juridiques (loi, contrat, etc.) dans les décisions judiciaires de la République Tchèque. Les deux modèles ont donné de bonnes performances avec des scores $F_1$ de 89% et 97% pour le HMM utilisant les tri-grammes comme descripteurs de mots, et des scores $F_1$ de 87% et 97% pour le PAUM en utilisant des 5-grammes de lemmes et les rôles grammaticaux (*Part-Of-Speech tag*) comme descripteurs.

Considérons un texte $T$ comme étant une séquence d'observations $t_{1:n}$,

avec chaque $t_i$ étant un segment de texte (mot, ligne, phrase, etc.). En considérant une collection de labels, l'étiquetage de $T$ consiste à affecter les labels appropriés à chaque $t_i$. La segmentation de $T$ est un étiquetage particulier qui implique de découper $T$ en des groupes qui ne se chevauchent pas (des partitions). Les tâches de sectionnement et d'annotation des entités, prises séparément, sont des problèmes de segmentation.

#### 1.2.1.1 Les modèles cachés de Markov (HMM)

Un modèle HMM[1] est une machine à états finis définie par un ensemble d'états $\{s_1, s_2, ..., s_m\}$. Un modèle HMM a pour fonction d'affecter une probabilité jointe $P(T, L) = \prod_{i=1}^{n} P(l_i|l_{i-1})P(t_i|l_i)$ à des paires de séquences d'observations $T = t_{1:n}$ et de séquences de labels $L = l_{1:n}$. Étant donné qu'un HMM est un modèle génératif, chaque label $l_i$ correspond à l'état $s_j$ dans lequel la machine a généré l'observation $t_i$. Il y a autant de labels candidats que d'états. Le processus d'étiquetage de $T$ consiste à déterminer la séquence de labels $L^*$ qui maximise la probabilité jointe ($L^* = \arg\max_L P(T, L)$). Une évaluation de toutes les séquences possibles de labels est nécessaire pour déterminer $L^*$. Pour éviter la complexité exponentielle $O(m^n)$ d'une telle approche, $n$ étant la longueur du texte et $m$ le nombre de labels candidats, l'algorithme de décodage Viterbi [Viterbi, 1967], basé sur de la programmation dynamique, permet d'obtenir une estimation de $L^*$. Cet algorithme utilise des paramètres estimés par apprentissage sur un corpus de textes annotés manuellement :

- un alphabet ou vocabulaire $\{o_1, o_2, ..., o_k\}$ ;
- un ensemble d'états $\{s_1, s_2, ..., s_m\}$ ;
- la probabilité que $s_j$ génère la première observation $\pi(s_j), \forall j \in [1..m]$ ;
- la distribution de probabilité de transition $P(s_i|s_j), \forall i, j \in [1..m]$ ;
- la distribution de probabilité d'émission $P(o_i|s_j), \forall i \in [1..k], \forall j \in [1..m]$.

Les probabilités de transition et d'émission peuvent être inférées en utilisant une méthode de maximum de vraisemblance comme l'algorithme d'espérance maximale. L'algorithme Baum-Welch [Welch, 2003] en est une spécification conçue spécialement pour le HMM.

1. Rabiner [1989] fournit plus de détails sur le modèle HMM.

L'avantage du HMM réside dans sa simplicité et sa vitesse d'entraînement. Cependant, il est difficile de représenter les segments à l'aide de multiples descripteurs distincts. Il est tout aussi difficile de modéliser la dépendance entre des observations distantes parce que l'hypothèse d'indépendance entre observations est très restrictive (i.e. l'état courant dépend uniquement des états précédents et de l'observation courante).

#### 1.2.1.2 Les champs conditionnels aléatoires à chaîne linéaire (CRF)

Même si l'algorithme Viterbi est aussi utilisé pour appliquer le modèle CRF à l'étiquetage de séquences, la structure du CRF diffère de celle du HMM. Au lieu de maximiser la probabilité jointe $P(L, T)$ comme le HMM, un modèle CRF [Lafferty *et al.* , 2001] cherche la séquence de labels $L^*$ qui maximise la probabilité conditionnelle suivante :

$$P(L|T) = \frac{1}{Z} \exp\left(\sum_{i=1}^{n}\sum_{j=1}^{q} \lambda_j f_j(l_{i-1}, l_i, t_{1:n}, i)\right)$$

où $Z = \sum_{l_{1:n} \in \mathrm{L(T)}} \exp\left(\sum_{i=1}^{n}\sum_{j=1}^{q} \lambda_j f_j(l_{i-1}, l_i, t_{1:n}, i)\right)$ est le facteur de normalisation, L(T) étant l'ensemble des séquences possibles de labels pour $T$, $q$ le nombre de fonction $f_j(\cdot)$.

Les fonctions potentielles $f_j(\cdot)$ sont les caractéristiques utilisées par les modèles CRF. Deux types de fonctions caractéristiques sont définies : les caractéristiques de transition qui dépendent des labels aux positions courantes et précédentes ($l_{i-1}$ et $l_i$ resp.) et de $T$ ; et les caractéristiques d'état qui sont des fonctions de l'état courant $l_i$ et de la séquence $T$. Ces fonctions $f_j(\cdot)$ sont définies à l'aide de fonctions à valeurs binaires ou réelles $b(T, i)$ qui combinent les descripteurs d'une position $i$ dans $T$ [Wallach, 2004]. Pour labelliser les références aux règles de loi par exemple, un CRF pourrait inclure par exemple les fonctions potentielles pour étiqueter « 700 » dans ce contexte « ... l'article 700 du code de procédure civile ... » :

$$f_1(l_{i-1}, l_i, t_{1:n}, i) = \begin{cases} b_1(T, i) & \text{si } l_{i-1} = \text{NORME} \wedge l_i = \text{NORME} \\ 0 & \text{sinon} \end{cases}$$

$$f_2(l_{i-1}, l_i, t_{1:n}, i) = \begin{cases} b_2(T, i) & \text{si } l_i = \text{NORME} \\ 0 & \text{sinon} \end{cases}$$

avec

$$b_1(T,i) = \begin{cases} 1 & \text{si } (t_{i-1} = \text{article}) \wedge (POS_{i-1} = \text{NOM}) \\ & \wedge (NP_{i-1} = \text{<unknown>}) \wedge (NS_{i-1} = \text{@card@}) \\ 0 & \text{sinon} \end{cases}$$

$$b_2(T,i) = \begin{cases} 1 & \text{si } (t_i = 700) \wedge (POS_i = \text{NUM}) \wedge (NP1_i = \text{article}) \wedge (NS1_i = \text{code}) \\ 0 & \text{sinon} \end{cases}$$

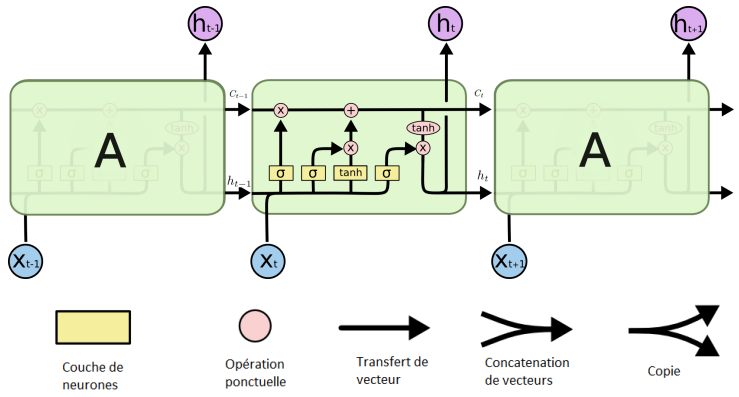
$t_i$ étant l'observation (le mot) en position $i$ dans $T$, $POS_i$ étant le rôle grammatical de $t_i$ ($NUM$ = valeur numérique, $NOM$ = nom), et $NP1_i$ et $NS1_i$ sont les lemmes des mots avant et après $t_i$, respectivement. Les symboles <*unknown*> et @*card*@ encodent les lemmes inconnus et ceux des nombres respectivement. Pouvant être activées au même moment, les fonctions $f_1$ et $f_2$ définissent des descripteurs se chevauchant. Avec plusieurs fonctions activées, la croyance dans le fait que $l_i = NORME$ est renforcée par la somme $\lambda_1 + \lambda_2$ des poids affectés respectivement à $f_1$ et $f_2$ [Zhu, 2010]. Un modèle CRF active une fonction $f_j$ lorsque ses conditions sont satisfaites (celles activant $b_j(T,\cdot)$) et $\lambda_j > 0$. Les diverses fonctions pondérées $f_j$ sont définies par des descripteurs caractérisant les segments, et les labels des données d'entraînement. La phase d'apprentissage consiste principalement à estimer le vecteur de paramètres $\lambda = (\lambda_1, ..., \lambda_F)$ à partir de textes annotés manuellement $\{(T_1, L_1), ..., (T_M, L_M)\}$, $T_k$ étant un texte et $L_k$ la séquence de labels correspondants. La valeur optimale de $\lambda$ est celle qui maximise la fonction objectif $\sum_{k=1}^{M} \log P(L_k|T_k)$ sur les données d'entraînement. En général, outre le maximum de vraisemblance, cette optimisation est résolue à l'aide de l'algorithme de descente de gradient dont l'exécution peut être accélérée à l'aide de l'algorithme L-BFGS de Liu & Nocedal [1989].

#### 1.2.1.3 CRF et réseaux de neurones artificiels

**Définitions** Un réseau de neurones artificiels est un algorithme d'apprentissage automatique dont le fonctionnement est inspiré de celui du cerveau [McCulloch & Pitts, 1943; Rosenblatt, 1958]. Un neurone reçoit des valeurs $x = [x_1, x_2, \ldots, x_l]$ en entrée, puis calcule un élément de sortie $y$ par une fonction de la combinaison pondérée des entrées (poids $W = [w_0, w_1, \ldots, w_l]$). Un réseau de neurones comprend des neurones structurés en couches. Les sorties d'une couche servent d'entrées à la couche sui-

vante et les nœuds d'une couche ne sont pas connectés. Les poids sont déterminés lors d'une phase d'entraînement qui les ajuste afin de minimiser l'erreur entre les valeurs attendues $y$ en sorties et celles prédites $\hat{y}$ par le modèle. Traditionnellement, les réseaux de neurones sont à « propagation avant » (*feed-forward*) i.e. l'unique sens de propagation de l'information est de la couche d'entrée successivement vers a couche de sortie. Mais comme nous l'avons vue précédemment, la prise en compte du contexte (état précédent ou suivant) est très importante dans la modélisation des séquences. Les réseaux à propagation avant ne sont pas adaptés pour prédire la sortie à l'instant $t$ à partir de ses connaissances des instants précédents.

Les réseaux récurrents de neurones (*recurrent neural networks - RNN*) [Jordan, 1986; Elman, 1990] sont une architecture conçue pour modéliser les données séquentielles $X = X_{1:n}$. Le principe est de passer à l'instant $t$ en entrée du réseau, la sortie ou état du réseau de l'instant précédent $t-1$ en plus de l'observation courante $X_t$. Un LSTM est une variante particulière de RNN dont l'état est définie par la sortie $h_t$ du réseau et la sortie $C_t$ qui permet de gérer la mémoire à plus long terme (Figure 1.1 Page 10).

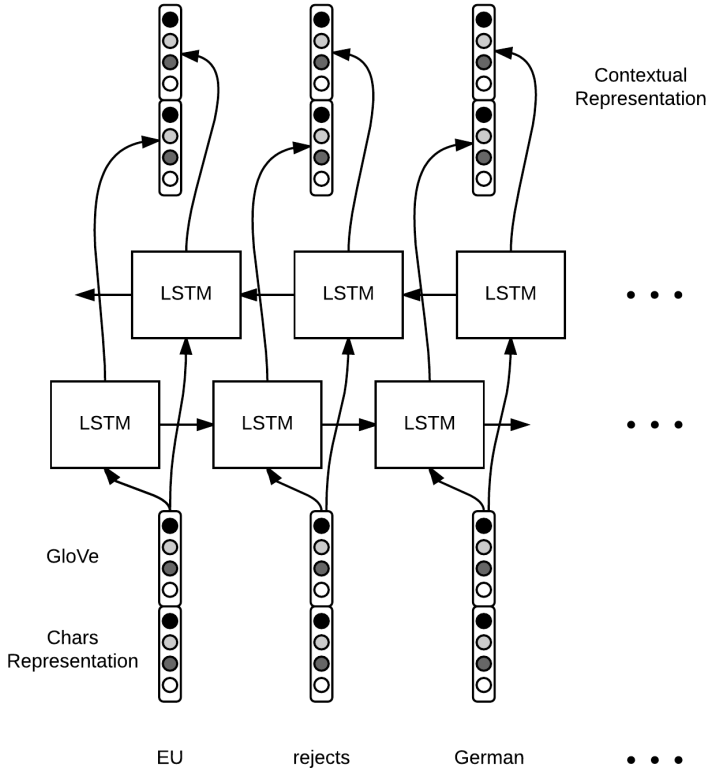



Source : `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`

Figure 1.1 – Structure interne d'un LSTM dans une couche de réseau LSTM

Les entrées et sorties d'un LSTM sont des vecteurs de nombres réels. L'entrée $X_i$ est la représentation vectorielle indépendante de la séquence entrée de l'observation en position $i$, par exemple celle du mot $t_i$ du texte $T = t_{1:n}$. La sortie $h_i$ représente le contexte de l'observation en position $i$ dans la séquence entrée. On peut ainsi chaîner des LSTM pour modéliser les textes entrées et y appliquer un CRF pour l'étiquetage. C'est le prin-

cipe du BiLSTM-CRF de Lample *et al.* [2016] une des architectures les plus efficaces pour la l'étiquetage d'entités nommées. En effet, le BiLSTM comprend deux couches enchaînat des LSTM (Figure 1.2 Page 11). Une couche permet d'apprendre le contexte "gauche" des mots, les états étant propager dans le sens du début vers la fin du texte. L'autre couche apprend le contexte droit en propageant les informations dans le sens inverse.




Source : `https://guillaumegenthial.github.io/sequence-tagging-with-tensorflow.html`

Figure 1.2 – Apprentissage de la représentation contextuelle avec une double couche de réseaux LSTM (BiLSTM)

**Représentation** Les réseaux de neurones permettent d'apprendre des caractéristiques grâce à des méthodes de plongement de mots telles que Word2Vec [Le & Mikolov, 2014a] et Glove [Pennington *et al.* , 2014].

**Entraînement**

**Décodage**

### 1.2.2 Représentation des segments atomiques

La représentation des segments à labelliser occupe une place importante pour l'obtention de bons résultats avec les modèles décrits précédemment. Elle consiste généralement à décrire la forme et le contexte de chaque segment en lui assignant des attributs [Nadeau & Sekine, 2007; Sharnagat, 2014]. Ils peuvent être booléens (« le mot est il en majuscule ? »), numériques (nombre de caractères du mot), nominaux (par ex. le rôle grammatical d'un mot), ou définis par des expressions régulières (par ex. pour les numéros R.G. on peut avoir `dd/ddddd` où `d` désigne un chiffre). Ces descripteurs mettent en évidence des régularités relatives à l'occurrence des entités. Par exemple, préciser qu'un mot débute par une lettre majuscule permet d'indiquer les noms propres. La définition de tels descripteurs consiste ainsi à fournir au modèle des indices l'aidant à mieux distinguer les différents types d'entités.

Etant donné que les descripteurs dépendent généralement de l'intuition du concepteur du système d'étiquetage, il est difficile mais nécessaire d'identifier des descripteurs appropriés. Après avoir défini des candidats, il n'est pas sûr qu'en les combinant tous ensemble, on obtienne les meilleures performances. Une sélection de caractéristiques peut alors s'avérer nécessaire. Cette sélection peut améliorer les performances d'étiquetage, et accélérer l'extraction des descripteurs, l'entraînement du modèle, ainsi que son application à de nouveaux textes [Kitoogo & Baryamureeba, 2007]. Elle peut aussi fournir une meilleure compréhension du comportement des modèles entraînés [Klinger & Friedrich, 2009]. Deux principales approches se distinguent. D'une part, les méthodes « filtrantes » (*filters*), comme l'information mutuelle, comparent individuellement les descripteurs à l'aide de scores qui ne sont pas nécessairement basés sur la performance. D'autre part, les méthodes « enveloppantes » (*wrappers*) comparent des sous-ensembles de descripteurs sur la base des performances d'évaluation qu'elles permettent d'obtenir (par exemple la $F_1$-mesure obtenue sur un ensemble d'exemples). Même si les méthodes filtrantes sont plus rapides, elles sont en général moins performantes car elles ne per-

mettent pas d'éviter les redondances, et ne prennent pas en compte l'effet de la combinaison de caractéristiques.

La définition manuelle des caractéristiques suivie de la sélection est souvent qualifiée de méthode forcée car elle dépend fortement de la capacité du concepteur du système à identifier les descripteurs appropriés.

### 1.2.3 Schéma d'étiquetage

Nous traitons d'entités dont les occurrences comprennent un ou plusieurs éléments atomiques. Pour améliorer les résultats d'un modèle d'étiquetage, certaines parties des entités peuvent être mises en évidence à travers une représentation appropriée de segments. La Figure 1.3 Page 13 illustre l'utilisation la différence entre des schémas appelés IO, BIO, IEO et BIEO, sur un extrait de décision de justice pour l'annotation du nom d'un juge et de sa fonction.

| | *composée* | *de* | *Madame* | *Martine* | *JEAN* | *,* | *Président* | *de* | *chambre* | *,* | *de* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| IO | O | O | I-JUGE | I-JUGE | I-JUGE | O | I-FONCTION | I-FONCTION | I-FONCTION | O | O |
| BIO | O | O | B-JUGE | I-JUGE | I-JUGE | O | B-FONCTION | I-FONCTION | I-FONCTION | O | O |
| IEO | O | O | I-JUGE | I-JUGE | E-JUGE | O | I-FONCTION | I-FONCTION | E-FONCTION | O | O |
| BIEO | O | O | B-JUGE | I-JUGE | E-JUGE | O | B-FONCTION | I-FONCTION | E-FONCTION | O | O |

Figure 1.3 – Illustration des schémas d'étiquetage IO, BIO, IEO, BIEO

Nous comparons dans cette étude quelques schémas d'étiquetage dont certains sont décrits par Konkol & Konopík [2015]. Le principe de ces schémas est d'étiqueter différemment des segments atomiques en fonction de leur position dans l'entité. Pour cela, le label associé à l'entité est préfixé par l'une des lettres suivantes :

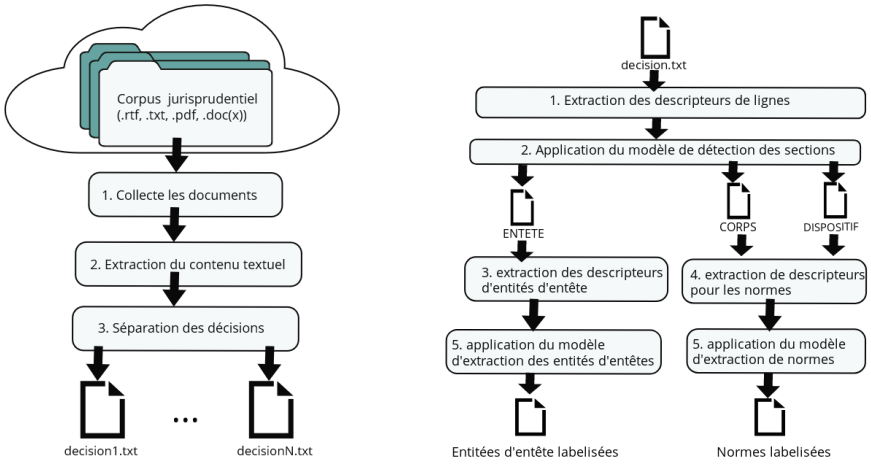
- B : début (*beginning*) ;
- I : intérieur (*inside*) ;
- E (ou L, ou M) : fin (*end* ou *last* ou *middle*) ;
- S (ou U, ou W) : singleton ou entité à segment unique (*single* ou *unit* ou *whole*) ;
- O : hors de toute entité (*outside*).

Le schéma IO utilisé par défaut ne met l'accent sur aucune partie et affecte le même label à tous les segments d'une même entité. D'autres schémas distinguent soit le premier élément (BIO), soit le dernier (IEO), soit les

deux (BIEO). Les schémas IEO et BIO ont des variantes IEO1, BIO1, IOE2, et BIO2. Les modèles IOE2, et BIO2 utilisent resp. les préfixes E- et B- pour étiqueter les entités à mot unique, contrairement à IEO1 et BIO1 qui utilisent plutôt le préfixe I- dans ce cas. Le modèle BIEO est souvent étendu sous la forme BIESO (ou BILOU) dans le cas où l'on souhaite distinguer les entités à un seul segment (par ex. ville ou numéro R.G.). Il est possible d'aller plus loin en mettant l'accent sur les mots avant (O-JUGE) et après (JUGE-O) l'entité (JUGE par exemple) et en indiquant le début (BOS-O, *begininning of sentence*) et la fin (O-EOS, *end of sentence*) du texte ou de la phrase. Le format ainsi obtenu est appelé BMEWO+ [Baldwin, 2009].

Un autre intérêt des schémas plus complexes que IO est de pouvoir distinguer des entités du même type qui se suivent sans être explicitement séparées (par exemple, des appelants mentionnés sur des lignes consécutives). Cet aspect est notamment important dans les décisions de justice par exemple lorsque des noms de parties sont listés dans la section ENTETE en n'étant séparés que d'un simple retour à la ligne.

## 1.3 Architecture proposée




Après la collecte et le pré-traitement des documents, l'étiqueteur de ligne est d'abord appliqué pour détecter les sections, puis les étiqueteurs d'entités peuvent être appliqués simultanément dans les sections.

Figure 1.4 – Application des modèles entraînés pour l'étiquetage de sections et entités.

Nous proposons de travailler uniquement avec le contenu textuel des

documents. Ce contenu est extrait des documents téléchargés en éliminant les éléments inutiles, principalement des espaces vides. Ces éléments sont typiques des documents formatés (`.rtf`, `.doc(x)`, `.pdf`). Ils ne fournissent pas une indication standard sur le début des sections. Le choix de ne pas exploiter le formatage des documents permet d'avoir à gérer un nombre plus faible de diversités entre les textes tout en appliquant le même processus de traitement à tout document indépendamment de son format d'origine. Une simple architecture d'étiquetage de sections et d'entités juridiques a été conçue avec cette uniformisation des documents comme point d'entrée (Figure 1.4 Page 14). Ainsi, les documents sont collectés puis prétraités suivant leur format d'origine (extraction du texte et séparation des décisions apparaissant dans le même document). Ensuite, après le sectionnement des décisions, les entités sont identifiées dans les différentes sections. Par ailleurs, comme segment atomique à étiqueter nous avons choisi les lignes pour la détection des sections, et les mots pour les entités.

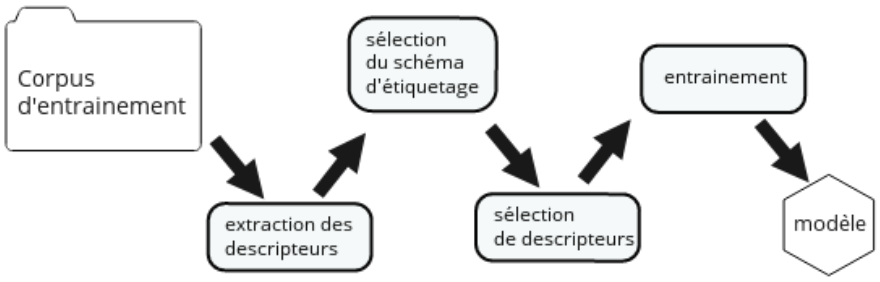



Figure 1.5 – Entrainement des modèles.

Les modèles HMM et CRF étant tous les deux supervisés, ils doivent être entraînés sur des exemples manuellement annotés pour estimer leurs paramètres. Nous proposons de sélectionner le schéma d'étiquetage et les sous-ensembles minimaux de caractéristiques manuellement définies, avant d'entraîner les modèles HMM et CRF (Figure 1.5 Page 15).

## 1.3.1 Définition de descripteurs candidats

### 1.3.1.1 Descripteurs pour la détection des sections

Nous considérons donc la ligne comme élément à étiqueter lors du sectionnement. Nous n'avons pas travaillé au niveau des mots afin d'éviter que des mots de la même ligne ne soient classés dans des sections différentes. L'étiquetage des phrases a aussi été évité car en découpant les documents en phrases telles qu'elles sont entendues en français, on a généralement des segments qui s'étendent d'une section à une autre (absence de ponctuation). De plus, l'entête en particulier d'avantage l'apparence d'un formulaire.

Plusieurs critères peuvent être utilisés pour différencier les sections, à savoir : la longueur des lignes (plus longues dans le corps, plus courtes dans l'entête), les premiers termes de certaines lignes (typiques de chaque section) et le nombre total de lignes. Un HMM n'adapte qu'un descripteur assimilé à l'élément à étiqueter. D'autres descripteurs peuvent être la position de l'élément à étiqueter (numéro de ligne) ou le début de la ligne. Le descripteur capturant la longueur de ligne peut être absolu (nombre exact de mots dans la ligne), ou relatif (une catégorie de la longueur). Sur la base des quantiles de la distribution des longueurs de lignes sur un ensemble de décisions, nous avons défini trois catégories : `LQ1` ($longueur \leq 5$), `LQ2` ($5 < longueur \leq 12$) et `LQ2` ($12 < longueur \leq 14$). Nous avons également catégorisé les parties de documents afin de capturer une position de ligne relative.

Lors de l'extraction des caractéristiques, le document est considéré comme divisé en $N$ parties (10 dans nos expériences). La position relative d'une ligne est donc le numéro de la partie contenant la ligne particulière. En résumé, les caractéristiques sont décrites comme suit (avec leurs étiquettes entre parenthèses) :

- forme de la ligne : la ligne entière, ses premiers mots (`t0, t1, t2`), sa longueur absolue (`absLength`) et sa longueur relative (`relLength`);
- contexte de ligne : le numéro de ligne (`absNum`) et le numéro de la partie de document contenant la ligne (`relNum`), les deux premiers mots des lignes précédente (`p0, p1`) et suivantes (`n0, n1`), ainsi que leurs longueurs absolues et relatives (`pLength`, `pRelLength`, `nLength`, `nRelLength`).

#### 1.3.1.2 Descripteurs pour la détection d'entités

La détection d'entités consiste, dans notre cas, à entraîner soit un modèle CRF, soit un modèle HMM pour étiqueter les différents segments de texte (mot, ponctuation, numéro, identifiant) suivant qu'ils appartiennent ou non à la mention d'une entité. Les deux modèles nécessitent des caractéristiques, dont certaines peuvent être définies sur la base de régularités directement observables dans les textes. Il est également possible d'obtenir des descripteurs à partir du résultat d'autres tâches d'analyse de texte.

Sur la base des observations de décision, nous avons défini la morphologie des mots pour les normes et méta-données d'entête :

- forme du mot : le mot (`token`), son lemme (`lemma_W0`), « commence-t-il par une lettre majuscule ? » (`startsWithCAP`), « est-il entièrement en majuscule ? » (`isAllCAP`), « est-ce une initiale solitaire ? » comme par exemple « B. » (`isLONELYINITIAL`), « contient-il un caractère de ponctuation ? » (`PUN-IN`), « n'est-ce qu'une ponctuation ? » (`isALLPUN`), « contient-il un caractère numérique ? » (`DIGIT-IN`), « ne contient-il que des chiffres ? » (`isALLDIGIT`) ;
- contexte de mot : les mots précédents (`w-2, w-1`) et suivants (`w1, w2`) et leurs lemmes ($\texttt{lemmaW}_i$). La lemmatisation homogénéise les variantes du même mot. Les mots adjacents sont choisis pour indiquer les termes couramment utilisés pour introduire des entités.

Plus particulièrement pour les méta-données d'entête, nous avons défini des descripteurs supplémentaires pour capter le contexte du mot : numéro de ligne (`lineNum`), position de l'élément dans la ligne (`numInLine`), « le document contient-il le mot clé *intervenant* ? » (`intervenantInText`), le texte vient-il après le mot clé « APPELANT » (`isAfterAPPELANT`), « INTIME » (`isAfterINTIME`), « INTERVENANT » (`isAfterINTERVENANT`). Nous avons également pris en compte les dernières lignes, où le mot était précédemment rencontré dans le texte ( `lastSeenAt` ), ainsi que le nombre de fois où il a été trouvé ( `nbTimesPrevSeen` ), car les noms des parties sont souvent répétés à des emplacements différents. Nous avons également défini une caractéristique spéciale pour les normes : « le mot est-il un mot clé de règles juridiques ? » (`isKEYWORD`). Pour ce dernier descripteur, nous avons établi une courte liste de mots-clés généralement utilisés pour citer des règles juridiques (*article, code, loi, contrat, décret, convention, civil, pénal*, etc.).

Nous avons étendu ces caractéristiques avec les rôles grammaticaux (*Part-of-Speech* et les modèles thématiques (*topic model*).

**Rôles grammaticaux** : certaines entités ont tendance à contenir des rôles grammaticaux particuliers. Par exemple, les noms d'individus sont composés de noms propres (Chang et Sung, 2005). Nous avons extrait le rôle grammatical du mot courant (`POS`) ainsi que celui de ses voisins (`POSW-2, POSW-1, POSW1, POSW2`).

**Modèles thématiques** : comme Polifroni & Mairesse [2011] et Nallapati *et al.* [2010], nous utilisons des associations mot-thème pour décrire les mots. Il s'agit de modéliser un ensemble de $N$ thèmes et d'utiliser leurs identifiants comme descripteurs. Il serait peut-être intéressant d'utiliser la probabilité déduite du modèle thématique, mais l'inférence sous-jacente au modèle LDA [Blei *et al.* , 2003] n'est pas déterministe (la distribution de probabilité change pour le même mot entre différentes inférences). Néanmoins, l'ordre des sujets ne changeant pas de manière significative, nous avons utilisé l'identifiant du thème le plus pertinent pour le mot (`topic0`) ainsi que ceux de ses voisins (`w-2topic0`, `w-1topic0`, `w1topic0`, `w2topic0`).

## 1.3.2 Sélection des descripteurs

### 1.3.2.1 Sélection pour le modèle CRF

Nous avons étudié deux approches enveloppantes qui semblent toujours converger et qui ne nécessitent pas de définir manuellement la taille du sous-ensemble cible.

La première méthode, qui est la recherche bidirectionnelle (BDS) de Liu & Motoda [2012], combine la sélection séquentielle en avant (SFS) et la sélection séquentielle en arrière (SBS) en parallèle (Algorithme 1). La SFS recherche un sous-ensemble optimal, en commençant par un ensemble vide et en ajoutant le descripteur qui améliore le mieux l'efficacité du sous-ensemble sélectionné. Le critère d'efficacité dans notre cas est défini par la $F_1$-mesure (Eq. 1.4.1). Contrairement à la SFS, la SBS commence par l'ensemble des candidats et supprime successivement les plus mauvais descripteurs. Une caractéristique ne peut être ajoutée dans $Y_{F_{k+1}}$ que si elle est présente dans $Y_{B_k}$.

La seconde méthode, qui est l'algorithme de sélection séquentielle avant à flottement SFFS de Pudil *et al.* [1994], étend la SFS en surmontant son incapacité à réévaluer l'utilité d'un descripteur après son rejet (Algorithme

**Algorithme 1 :** Recherche bidirectionnelle BDS

**Données :** Données annotées, $X$ liste de tous les descripteurs candidats
**Résultat :** Meilleur sous-ensemble de descripteurs

Démarrer la SFS avec $Y_{\mathcal{F}_0} = \emptyset$;
Démarrer la SBS avec $Y_{\mathcal{B}_0} = X$;
$k = 0$;
**tant que** $Y_{\mathcal{F}_k} \neq Y_{\mathcal{B}_k}$ **faire**
    $x^+ = \underset{x \in Y_{\mathcal{B}_k} \setminus Y_{\mathcal{F}_k}}{\text{argmax}} F_1(Y_{\mathcal{F}_k} + x)$ ; $Y_{\mathcal{F}_{k+1}} = Y_{\mathcal{F}_k} + x^+$ //SFS;
    $x^- = \underset{x \in Y_{\mathcal{B}_k} \setminus Y_{\mathcal{F}_{k+1}}}{\text{argmax}} F_1(Y_{\mathcal{F}_k} - x)$ ; $Y_{\mathcal{B}_{k+1}} = Y_{\mathcal{B}_k} - x^-$ //SBS;
    $k = k + 1$ ;
**retourner** $Y_{\mathcal{F}_k}$;

2). En effet, le SFFS effectue des tests en arrière à chaque itération.

#### 1.3.2.2 Sélection pour le modèle HMM

Pour sélectionner les meilleurs descripteurs pour les modèles HMM, nous avons testé individuellement les différents candidats. La caractéristique donnant le meilleur résultat sur l'ensemble de données annotées est sélectionnée.

## 1.4 Expérimentations et discussions

L'objectif de cette section est de discuter des différents aspects liés à la performance des modèles CRF et HMM. Il est question de discuter l'effet des descripteurs candidats définis, de comparer des algorithmes de sélection de caractéristiques et des schémas d'étiquetage. Nous discutons par la suite l'origine des erreurs (confusion, nombre d'exemples d'entraînement), et comparons les descripteurs définis manuellement par rapport à l'utilisation de réseaux de neurones.

**Algorithme 2 :** Sélection séquentielle avant à flottement

**Données :** Données annotées, $X$ liste de tous les descripteurs candidats
**Résultat :** Meilleur sous-ensemble de descripteurs

1 $Y_0 = \emptyset$;
2 $k = 0$;
3 **répéter**
4 $\quad x^+ = \underset{x \notin Y_k}{\operatorname{argmax}} F_1(Y_k + x)$ ; $Y_k = Y_k + x^+$;
5 $\quad x^- = \underset{x \in Y_k}{\operatorname{argmax}} F_1(Y_k - x)$;
6 $\quad$ **si** $F_1(Y_k - x^-) > F_1(Y_k)$ **alors**
7 $\quad\quad Y_{k+1} = Y_k - x^-$;
8 $\quad\quad X = X - x^-$;
9 $\quad\quad k = k + 1$;
10 $\quad\quad$ Rentrer à 5;
11 $\quad$ **sinon**
12 $\quad\quad$ Rentrer à 4;
13 **jusqu'à** $X = \emptyset$ *ou* $X = Y_k$;
14 **retourner** $Y_k$;

## 1.4.1 Conditions d'expérimentations

### 1.4.1.1 Annotation des données de référence

Pour évaluer les méthodes de TAL, Xiao [2010] suggère de choisir un jeu d'exemples suffisant en assurant au mieux l'équilibre dans la variété des données et la représentativité du langage. Nous avons essayé de suivre cette recommandation en sélectionnant aléatoirement des décisions à annoter. Au total, 503 documents ont été rassemblés et annotés manuellement à l'aide de la plateforme GATE Developer [2]. Cet outil permet de marquer les passages à annoter en les surlignant à l'aide du pointeur de la souris ; ce qui allège l'annotation manuelle. Des balises XML sont rajoutées autour des passages sélectionnés, en arrière plan dans le document.

Chaque document annoté comprend en moyenne 260 lignes et 3900 mots environ. Les deux dernières colonnes du Tableau 1.1 Page 3 pré-

2. https ://gate.ac.uk/family/developer.html

sentent la distribution des entités labellisées dans le jeu de données. En se basant sur un sous-ensemble de 13 documents labellisés par 2 annotateurs différents, nous avons calculé des taux d'accord inter-annotateur en utilisant la statistique Kappa de Cohen [1960]. Ces mesures d'accord inter-annotateur ont été calculées au niveau des caractères parce que certains mots peuvent être coupés par des annotations incorrectes (par ex. <*juridiction*> *cour d'appe* </*juridiction*> *l* contre <*juridiction*> *cour d'appel* </*juridiction*>), ou bien les annotateurs pourraient ne pas être d'accord si une apostrophe devrait être incluse ou pas dans l'annotation (par ex. *l'*<*norme*>*article 700* contre <*norme* >*l'article 700*). Les taux de Kappa de 0,705 et 0,974 ont été obtenus pour l'annotation des entités et des sections respectivement. D'après la catégorisation de Viera *et al.* [2005], le niveau d'accord observé est *substantiel* pour les entités (0,61 – 0,80) et *presque parfait* pour les sections (0,81 – 0,99).

#### 1.4.1.2 Mesures d'évaluation

Nous avons utilisé la précision, le rappel et la $F_1$-mesure comme mesures d'évaluation car elles sont généralement utilisées comme références en extraction d'information. La $F_1$-mesure se calcule à l'aide de la formule suivante :

$$F_1 = 2 \times \frac{Precision \times Rappel}{Precision + Rappel}. \tag{1.4.1}$$

L'évaluation peut être faite au niveau des segments atomiques ou des entités selon que l'on soit plus intéressé respectivement par l'étiquetage du maximum de segments atomiques ou par la labellisation complète d'un maximum d'entités.

**Evaluation au niveau atomique (*token-level)*** : cette évaluation mesure la capacité d'un modèle à labelliser les segments atomiques des entités. Les valeurs de précision et rappel sont calculées sur les données de test pour chaque label $l$ comme suit :

$$Precision_l = \frac{\text{nombre de segments correctement labellisés par le modèle avec } l}{\text{nombre de segments labellisés par le modèle avec } l}$$

$$Rappel_l = \frac{\text{nombre de segments correctement labellisés par le modèle avec } l}{\text{nombre de segments manuellement labellisés avec } l}$$

**Evaluation au niveau entité (*entity-level*)** : cette évaluation mesure le taux d'entités parfaitement identifiées c'est-à-dire seulement celles dont les segments atomiques ont été tous correctement labellisés. Les valeurs de précision et rappel sont calculées sur les données de test pour chaque classe d'entité $e$ comme suit :

$$Precision_e = \frac{\text{nombre d'entités de type } e \text{ parfaitement détectées par le modèle}}{\text{nombre d'entités détectées et classifiées } e \text{ par le modèle}}$$

$$Rappel_e = \frac{\text{nombre d'entités de type } e \text{ parfaitement détectées par le modèle}}{\text{nombre d'entitées manuellement classifiées } e}$$

**Evaluation globale (*overall-level*)** : l'évaluation globale donne les performances générales d'un modèle sans distinction des classes ou labels. Elle est réalisée aux deux niveaux décrits précédemment mais indépendamment du label d'élément ou du type d'entité. La précision et le rappel sont calculées au niveau des entités comme suit :

$$Precision = \frac{\text{nombre d'entités correctement labellisées par le modèle}}{\text{nombre d'entités labellisées par le modèle}}$$

$$Rappel = \frac{\text{nombre d'entités correctement labellisées par le modèle}}{\text{nombre d'entités manuellement labellisées}}.$$

Ces métriques sont calculées de la même façon au niveau atomique.

#### 1.4.1.3 Outils logiciels

Nous avons utilisé les modèles HMM et CRF tels qu'implémentés dans la librairie Mallet [McCallum, 2012]. Les modèles étudiés ont été entraînés par la méthode d'espérance maximale pour ceux basés sur le HMM, et par la méthode L-BFGS pour ceux basés sur le CRF. Le découpage des textes en mots (*tokenisation*), la lemmatisation, et l'annotation des rôles grammaticaux (*Part-of-Speech tagging*) ont été effectués à l'aide de la fonctionnalité d'annotation de textes français de TreeTagger[3] [Schmid, 1994]. L'implémentation dans Mallet du LDA [Blei *et al.* , 2003] a permis d'inférer 100 thèmes à partir d'un corpus lemmatisé d'environ 6k documents. Le Tableau 1.2 Page 23 présente des mots représentatifs trouvés dans les premiers thèmes inférés. L'extraction des autres descripteurs a été implémentée pour cette expérimentation.

3. `http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger`

| Id thème | Mots représentatifs |
|---|---|
| 0 | préjudice dommage somme subir réparation titre faute payer intérêt responsabilité |
| 1 | société salarié groupe mirabeau pouvoir demande article licenciement cour titre |
| 2 | harcèlement travail salarié moral employeur fait attestation faire santé agissements |
| 3 | vente acte prix vendeur acquéreur notaire condition clause vendre immeuble |
| 4 | travail poste reclassement employeur médecin licenciement salarié inaptitude visite |
| 5 | monsieur nîmes avocat appel barreau arrêt madame disposition prononcer président |
| 6 | mademoiselle madame non mesure décision tutelle surendettement comparant |
| 7 | transport marchandise jeune sed éducateur bateau navire transporteur responsabilité |
| 8 | congé salarié conversion emploi plan convention employeur sauvegarde reclassement |
| 9 | marque site contrefaçon sous droit auteur joseph produit propriété photographie |
| 10 | pierre patrick bordeaux bruno catherine civil article corinne cour avocat |

Tableau 1.2 – Mots représentatifs des 10 premiers thèmes sur les 100 inférés

Les valeurs de précision, rappel, et $F_1$-mesure ont été calculées à l'aide du script d'évaluation de la campagne CoNLL-2002 [4]. Elles sont indiquées en pourcentage dans les tableaux de résultats d'évaluation des sections suivantes.

### 1.4.2 Sélection du schéma d'étiquetage

Dans le but d'évaluer comment la représentation de segments affecte les performances, nous avons implémenté quatre représentations (IO, IEO2, BIO2, BIEO). Nous avons réalisé un simple découpage des données en deux ensembles : 25% pour l'entraînement et 75% pour les tests. Les performances reportées dans le Tableau 1.3 Page 24 sont les performances globales sur la base de test. Seul l'élément (mot/ligne) est utilisé comme descripteur. La durée d'entraînement est très longue, particulièrement pour la détection d'entités dans l'entête avec le CRF. Il semble évident que cette durée croisse avec le nombre de labels candidats de la section et la complexité du schéma d'étiquetage. En effet, BIEO exige beaucoup plus de temps, et IO exige le temps d'entraînement le plus bas, et le schéma IOE semble être plus rapide que BIO même s'ils ont le même nombre de labels. Nous remarquons aussi que les représentations complexes n'améliorent pas significativement les résultats par rapport au simple IO qui demande pourtant beaucoup moins de temps.

4. `http://www.cnts.ua.ac.be/conll2002/ner/bin/conlleval.txt`

| Tâche | Modèle | Niveau atomique[a] | | | Niveau entité[a] | | | Durée[b] | Schéma |
|---|---|---|---|---|---|---|---|---|---|
| | | Précision | Rappel | $F_1$ | Précision | Recall | $F_1$ | | |
| Sections | CRF | 91.75 | 91.75 | 91.75 | 64.49 | 56.55 | 60.26 | 4.685 | IO |
| | | 88.95 | 88.95 | 88.95 | 48.12 | 38.26 | 42.63 | 11.877 | IEO2 |
| | | 87.09 | 87.09 | 87.09 | 46.79 | 37.20 | 41.45 | 12.256 | BIO2 |
| | | 86.00 | 86.00 | 86.00 | 58.98 | 41.86 | 48.97 | 35.981 | BIEO |
| | HMM | 32.64 | 32.64 | 32.64 | 22.16 | 18.91 | 20.41 | 6.564 | IO |
| | | 32.92 | 32.92 | 32.92 | 17.73 | 16.09 | 16.87 | 7.827 | IEO2 |
| | | 32.39 | 32.39 | 32.39 | 31.93 | 26.65 | 29.05 | 8.391 | BIO2 |
| | | 33.06 | 33.06 | 33.06 | 32.47 | 27.53 | 29.80 | 8.7 | BIEO |
| Entités d'entête | CRF | 86.86 | 78.96 | 82.73 | 80.84 | 65.17 | 72.17 | 70.525 | IO |
| | | 87.77 | 79.65 | 83.51 | 82.46 | 65.19 | 72.82 | 228.751 | IEO2 |
| | | 87.41 | 78.14 | 82.51 | 81.66 | 66.80 | 73.49 | 230.865 | BIO2 |
| | | 87.72 | 79.55 | 83.44 | 84.38 | 68.35 | 75.53 | 475.249 | BIEO |
| | HMM | 79.12 | 67.75 | 73.00 | 61.48 | 35.05 | 44.64 | 6.345 | IO |
| | | 78.82 | 68.69 | 73.40 | 66.63 | 40.16 | 50.11 | 8.298 | IEO2 |
| | | 80.68 | 67.48 | 73.49 | 70.37 | 45.32 | 55.14 | 7.908 | BIO2 |
| | | 80.05 | 69.01 | 74.12 | 74.73 | 50.77 | 60.46 | 9.973 | BIEO |
| Normes | CRF | 95.60 | 92.96 | 94.26 | 88.06 | 83.50 | 85.72 | 28 | IO |
| | | 95.40 | 93.18 | 94.27 | 88.75 | 85.65 | 87.17 | 32.136 | IEO2 |
| | | 95.20 | 93.30 | 94.24 | 85.65 | 83.13 | 84.37 | 50.769 | BIO2 |
| | | 95.46 | 91.57 | 93.47 | 88.83 | 84.71 | 86.72 | 50.566 | BIEO |
| | HMM | 89.83 | 88.78 | 89.30 | 73.74 | 75.02 | 74.37 | 41.389 | IO |
| | | 88.20 | 89.23 | 88.71 | 78.01 | 81.27 | 79.61 | 44.086 | IEO2 |
| | | 89.25 | 87.83 | 88.53 | 73.89 | 76.63 | 75.24 | 46.634 | BIO2 |
| | | 87.39 | 88.10 | 87.74 | 77.76 | 82.35 | 79.99 | 45.52 | BIEO |

[a] Résultats sur une simple division du jeu de données en 25% pour l'entraînement et 75% pour les tests (entraînement limité à 100 itérations maximum)
[b] Durée d'entraînement en secondes avant l'arrêt de l'entraînement

Tableau 1.3 – Comparaison des schémas d'étiquetage.

### 1.4.3 Sélection des descripteurs

Pour comparer les méthodes BDS et SFFS, nous exploitons le schéma IO. Durant nos expérimentations, la méthode SFFS a exécuté 185 entraînements pour le modèle CRF d'identification des sections. La méthode BDS quant à elle a duré plus de 15h pour 600 itérations d'entraînement-test. Malgré la sauvegarde des scores $F_1$ pour éviter d'exécuter plusieurs fois l'entraînement pour les mêmes sous-ensembles de descripteurs, le processus de sélection est toujours resté très long pour les deux algorithmes. Nous avons testé individuellement chacun des descripteurs candidats pour les modèles HMM. Les résultats sont reportés dans le Tableau 1.4 Page 25.

Le résultat le plus remarquable est la forte réduction du nombre de descripteurs par les algorithmes. En général, la moitié est éliminée par la sélection BDS, tandis que la méthode SFFS élimine beaucoup plus de candidats (par exemple en ne sélectionnant que 4 descripteurs parmi les

| Tâche | Modèle | niveau atomique[a] | | | niveau entité[a] | | | Sous-ensemble |
|---|---|---|---|---|---|---|---|---|
| | | Précision | Rappel | $F_1$ | Précision | Rappel | $F_1$ | sélectionné |
| Sections | CRF | 99.31 | 99.31 | 99.31 | 90.28 | 90.68 | 90.48 | BDS[b1] |
| | | 99.55 | 99.55 | 99.55 | 85.69 | 85.84 | 85.76 | SFFS[b2] |
| | | 99.36 | 99.36 | 99.36 | 88.16 | 88.39 | 88.27 | TOUS[b0] |
| | | 91.75 | 91.75 | 91.75 | 64.49 | 56.55 | 60.26 | token |
| | HMM | 90.99 | 90.99 | 90.99 | 4.18 | 3.63 | 3.89 | absLength |
| | | 86.97 | 86.97 | 86.97 | 4.08 | 3.30 | 3.65 | relLength |
| | | 37.59 | 37.59 | 37.59 | 18.81 | 18.81 | 18.81 | token |
| Entités d'entête | CRF | 94.00 | 91.42 | 92.69 | 92.26 | 88.76 | 90.47 | BDS[c1] |
| | | 94.10 | 91.93 | 93.00 | 92.64 | 88.96 | 90.76 | SFFS[c2] |
| | | 94.20 | 91.86 | 93.02 | 93.05 | 89.59 | 91.28 | TOUS[c0] |
| | | 86.86 | 78.96 | 82.73 | 80.84 | 65.17 | 72.17 | token |
| | HMM | 76.90 | 80.41 | 78.61 | 62.66 | 52.16 | 56.93 | token |
| | | 66.48 | 69.67 | 68.04 | 39.34 | 28.36 | 32.96 | lemma_W0 |
| | | 39.63 | 37.50 | 38.54 | 15.49 | 5.35 | 7.95 | POS |
| Normes | CRF | 95.91 | 96.72 | 96.31 | 91.14 | 90.45 | 90.80 | BDS[d1] |
| | | 95.68 | 95.45 | 95.57 | 90.34 | 88.27 | 89.29 | SFFS[d2] |
| | | 95.07 | 96.69 | 95.87 | 90.87 | 90.64 | 90.76 | TOUS[d0] |
| | | 95.60 | 92.96 | 94.26 | 88.06 | 83.50 | 85.72 | token |
| | HMM | 89.21 | 94.25 | 91.66 | 72.67 | 77.28 | 74.90 | **token** |
| | | 90.31 | 92.81 | 91.54 | 69.24 | 69.46 | 69.35 | lemma_W0 |

[a] Résultats sur un simple découpage des données de 25% pour l'entraînement, 75% pour le test avec 100 itérations d'entraînement au maximum pour le CRF, et 80% pour l'entraînement et 20% pour le test avec 50 itérations au maximum pour l'entraînement du HMM

[b0] **Tous les candidats définis pour les sections (16 descripteurs)** : { relNum, relLength, pRelLength, absLength, t0, t1, t2, absNum, pLength, nRelLength, n0, nLength, p0, p1, n1, token }

[b1] **Selection par BDS pour les sections (07 descripteurs)** : {p0, n0, relNum, absLength, t0, t1, t2}

[b2] **Selection par SFFS pour les sections (06 descripteurs)** : { n0, nRelLength, relNum, t0, t1, t2 }

[c0] **Tous les candidats définis pour les méta-données d'entête (34 descripteurs)** : { isLONELYINITIAL, isALLCAP, isALLDIGIT, DIGIT-IN, intervenantInText, lineNum, lastSeenAt, nbTimesPrevSeen, isAfterAPPELANT, isAfterINTIME, isAfterINTERVENANT, startsWithCAP, PUN-IN, isALLPUN, POSW2, w2topic0, numInLine, POSW-1, lemmaW2, lemmaW-2, POSW-2, w-2topic0, POSW1, w1topic0, token, POS, lemma_W0, topic0, w2, w-1topic0, lemmaW-1, w-1, w1, lemmaW1 }

[c1] **Selection par BDS pour les méta-données d'entête (17 descripteurs)** : { POSW1, isAfterAPPELANT, numInLine, w-2topic0, POSW2, isAfterINTERVENANT, isAfterINTIME, POSW-2, isLONELYINITIAL, token, lemma_W0, lemmaW-2, isALLPUN, w-1, w1, w2, isALLCAP }

[c2] **Selection par SFFS pour les entités d'entête (10 descripteurs)** : { numInLine, w-2topic0, lemmaW-2, isAfterINTERVENANT, isAfterINTIME, w-1, w1, w2, isALLCAP, token }

[d0] **Tous les candidats définis pour les normes (28 descripteurs)** : { isALLPUN, isALLDIGIT, DIGIT-IN, isKEYWORD, POSW2, w2topic0, PUN-IN, POSW-1, isLONELYINITIAL, startsWithCAP, isALLCAP, lemmaW-2, POSW-2, w-2topic0, POS, topic0, POSW1, w1topic0, w2, lemmaW2, token, lemma_W0, w-2, w-1topic0, w-1, lemmaW-1, w1, lemmaW1 }

[d1] **Selection par BDS pour les normes (14 descripteurs)** : { POSW1, w-2topic0, isKEYWORD, lemmaW2, DIGIT-IN, token, lemmaW1, lemmaW-2, POS, isALLPUN, w-1, w2, PUN-IN, w-2 }

[d2] **Selection par SFFS pour les normes (04 descripteurs)** : {POSW1, lemmaW-2, w-1, DIGIT-IN}

Tableau 1.4 – Performances des sous-ensembles sélectionnés de descripteurs.

14 candidats définis pour l'annotation des normes).

Par ailleurs, les algorithmes de sélection forment des combinaisons in-

attendues. Par exemple, dans le cas de la détection de sections, la ligne suivante semble être beaucoup plus indicatrice que la première. Il est aussi intéressant de noter que les descripteurs basés sur notre observation apparaissent dans les sous-ensembles sélectionnés (par ex. `isAfterIntervenant`, `isKEYWORD`). Remarquons aussi que la longueur absolue des lignes (`absLength`) joue un rôle important dans l'identification des sections vu qu'il a été sélectionné à la fois pour le CRF et le HMM (sélection BDS). Avec ces sous-ensembles sélectionnés, les modèles sont plus performants que lorsqu'ils exploitent seulement le segment ou l'ensemble tout entier des candidats. Cette amélioration des résultats n'est pas très importante au regard de la longue durée d'exécution des algorithmes. Ainsi, un algorithme plus rapide et plus efficace devrait être utilisé.

### 1.4.4 Evaluation détaillée pour chaque classe

Nous discutons ici la capacité des modèles à identifier individuellement chaque type d'entité et de section. Les expérimentations ont été réalisées avec tous les descripteurs pour les modèles CRF. Seuls `absLength` et `token` ont été utilisés comme descripteurs dans les modèles HMM pour l'identification des sections et des entités respectivement. Le schéma d'étiquetage est IO. Le nombre d'itérations maximal a été fixé à 500 pour assurer la convergence lors de l'entraînement. Les Tableaux 1.5 et 1.6 présentent les résultats d'une validation croisée à 5 itérations, respectivement aux niveaux atomique et entité.

D'un point de vue général (évaluation globale), les modèles HMM se comportent assez bien au niveau élément avec un seul descripteur, particulièrement pour l'identification des sections et des normes. Le modèle HMM est capable de labelliser les normes car plusieurs d'entre elles sont répétées entre les décisions. De plus, la citation des normes est quasi standard (`article [IDENTIFIANT] [TEXTE D'ORIGINE]`). Le modèle HMM n'est cependant pas aussi efficace pour détecter entièrement les mots des entités d'où le faible score enregistré au niveau entité. Quant aux modèles CRF, leurs résultats sont très bons sur toutes les tâches et à tous les niveaux d'évaluation malgré quelques limites observées sur l'identification des parties.

| | HMM | | | CRF | | |
|---|---|---|---|---|---|---|
| | *Precision* | *Rappel* | $F_1$ | *Precision* | *Rappel* | $F_1$ |
| **I-corps** | 92.46 | 95.25 | 93.83 | 99.57 | 99.69 | 99.63 |
| **I-dispositif** | 53.44 | 48.46 | 50.83 | 98.63 | 97.59 | 98.11 |
| **I-entete** | 97.91 | 91.93 | 94.83 | 99.51 | 99.55 | 99.53 |
| **Evaluation globale** | 90.63 | 90.63 | 90.63 | 99.48 | 99.48 | 99.48 |
| **I-appelant** | 34.46 | 16.87 | 22.65 | 84.34 | 76.27 | 80.1 |
| **I-avocat** | 85.17 | 98.75 | 91.46 | 98.02 | 98.15 | 98.09 |
| **I-date** | 75.67 | 72.45 | 74.02 | 98 | 96.6 | 97.3 |
| **I-fonction** | 88.81 | 64.46 | 74.7 | 95.23 | 95.13 | 95.18 |
| **I-formation** | 79.38 | 94.38 | 86.23 | 98.8 | 99.45 | 99.12 |
| **I-intervenant** | 82.07 | 38.04 | 51.98 | 83.38 | 68.26 | 75.07 |
| **I-intime** | 50.4 | 68.09 | 57.93 | 82.54 | 83.33 | 82.93 |
| **I-juge** | 73.4 | 88.73 | 80.34 | 97.55 | 97.23 | 97.39 |
| **I-juridiction** | 85.15 | 98.37 | 91.28 | 98.91 | 99.69 | 99.3 |
| **I-rg** | 68.53 | 22.14 | 33.47 | 97.81 | 97.44 | 97.62 |
| **I-ville** | 91.5 | 82.41 | 86.72 | 98.94 | 99.15 | 99.04 |
| **Evaluation globale** | 76.21 | 82.26 | 79.12 | 95.13 | 94.51 | 94.82 |
| **I-norme** | 88.23 | 93.7 | 90.89 | 97.14 | 96.09 | 96.62 |

Tableau 1.5 – Précision, Rappel, $F_1$-mesures pour chaque type d'entité et section au niveau atomique.

| | HMM | | | CRF | | |
|---|---|---|---|---|---|---|
| | *Precision* | *Rappel* | $F_1$ | *Precision* | *Rappel* | $F_1$ |
| **corps** | 0.99 | 0.99 | 0.99 | 89.57 | 90.1 | 89.83 |
| **dispositif** | 12.05 | 7.33 | 9.11 | 98.02 | 97.82 | 97.92 |
| **entete** | 10.47 | 10.5 | 10.48 | 92.11 | 92.48 | 92.29 |
| **Evaluation globale** | 7.22 | 6.27 | 6.71 | 93.22 | 93.47 | 93.34 |
| **appelant** | 17.84 | 5.6 | 8.52 | 84.05 | 77.29 | 80.53 |
| **avocat** | 44.29 | 39.15 | 41.56 | 90.97 | 90.3 | 90.63 |
| **date** | 66.87 | 62.15 | 64.43 | 97.96 | 96.6 | 97.27 |
| **fonction** | 89.84 | 64.13 | 74.84 | 96.89 | 96.94 | 96.92 |
| **formation** | 61.5 | 65.86 | 63.61 | 98.4 | 98.95 | 98.68 |
| **intervenant** | 14.29 | 4 | 6.25 | 62.5 | 40 | 48.78 |
| **intime** | 30.28 | 27.47 | 28.8 | 79.31 | 78.93 | 79.12 |
| **juge** | 73.54 | 83.21 | 78.07 | 96.58 | 96.35 | 96.47 |
| **juridiction** | 81.31 | 87.66 | 84.37 | 98.86 | 99.54 | 99.2 |
| **rg** | 68.53 | 22.41 | 33.77 | 97.57 | 98.02 | 97.79 |
| **ville** | 89.52 | 84.7 | 87.05 | 98.85 | 99.15 | 99 |
| **Evaluation globale** | 64.59 | 54.56 | 59.15 | 93.77 | 92.93 | 93.35 |
| **norme** | 71.94 | 78.45 | 75.05 | 92.66 | 91.38 | 92.01 |

Tableau 1.6 – Précision, Rappel, $F_1$-mesures pour chaque type d'entité et section au niveau entité.

### 1.4.5 Discussions

#### 1.4.5.1 Confusion de classes

Certaines erreurs sont probablement dues à la proximité des entités de types différents. D'après la matrice de confusion des méta-données d'entête (Figure 1.6 Page 29), les *intervenants* sont parfois mal classifiés comme *intimé* en majorité (17 %), mais aussi comme *appelant* (4 %) ou *avocat* (2 %) probablement parce qu'il s'agit d'entités mentionnées les unes à la suite des autres dans l'entête (les *intervenants* sont généralement mentionnés juste après les *avocats* des *intimés*). De plus, les intervenants apparaissent dans une très faible proportion de documents annotés. Par ailleurs, une quantité considérable d'*appelants* sont aussi classifiés comme *intimés* (16 %). Ce qui signifie que la transition entre la liste des appelants et celle des intimés est difficilement identifiable avec les descripteurs de mots que nous avons définis.

La proximité crée aussi des confusions entre les sections CORPS et DISPOSITIF qui se suivent (Figure 1.7 Page 30).

#### 1.4.5.2 Redondance des mentions d'entités

Il est aussi intéressant de remarquer que certaines entités sont répétées dans le document. Par exemple, les noms des parties apparaissent précédemment à une mention qui donne plus de détails. Certaines normes sont aussi citées plusieurs fois et en alternant souvent les formes abrégées et longues (par exemple, la juridiction, la date, les normes). Bien que les différentes occurrences d'une même méta-données ne soient pas toujours identiques, de telles redondances aident à réduire le risque de manquer une entité. Cet aspect peut être exploité afin de combler l'imperfection des modèles.

#### 1.4.5.3 Impact de la quantité d'exemples annotés

Des expérimentations ont été menées pour évaluer les variations des modèles lorsque l'on augmente le nombre de données d'entraînement. Pour cela, nous avons évalué différentes tailles de la base d'entraînement. Les données ont été divisées en $75\% - 25\%$ pour resp. l'entraînement et le test. 20 fractions de l'ensemble d'entraînement ont été utilisées (de 5% à 100%). A chaque session entraînement-test, le même jeu de test a été

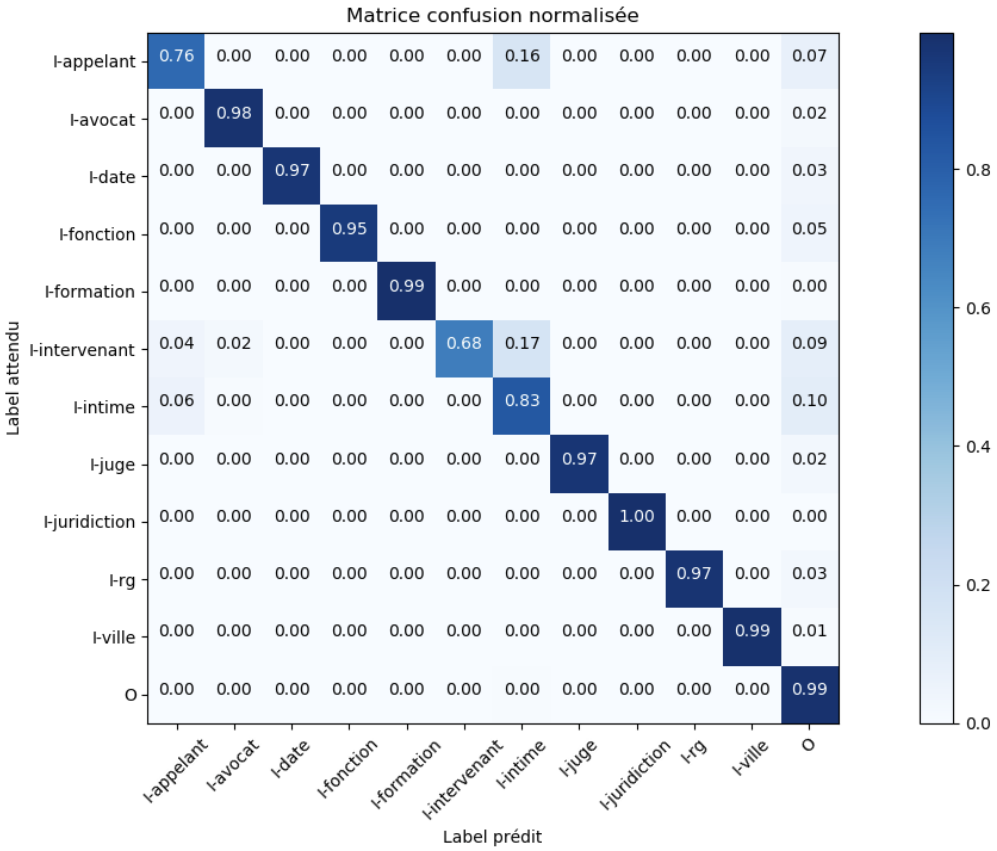



Figure 1.6 – Matrice de confusion entre méta-données d'entête avec le modèle CRF

employé pour les différentes fractions de l'ensemble d'entraînement. Les courbes d'apprentissage des modèles CRF et HMM sont représentées resp. sur les Figures 1.8a (Page 31) et 1.8b (Page 31) .

Il apparaît que les scores $F_1$ croissent avec le nombre de données d'entraînement pour les CRF et HMM, mais cette amélioration devient très faible au-delà de 60% de données d'entraînement quelle que soit la tâche. Il est possible que les exemples ajoutés par la suite partagent la même structure que celle de ceux qui ont été ajoutés auparavant. Ainsi, cette étude doit être étendue à la sélection des exemples les plus utiles. Raman & Ioerger [2003] ont démontré les avantages des algorithmes de sélection d'exemples combinés à celle des caractéristiques pour la classification. Les mêmes méthodes sont probablement applicables à l'étiquetage

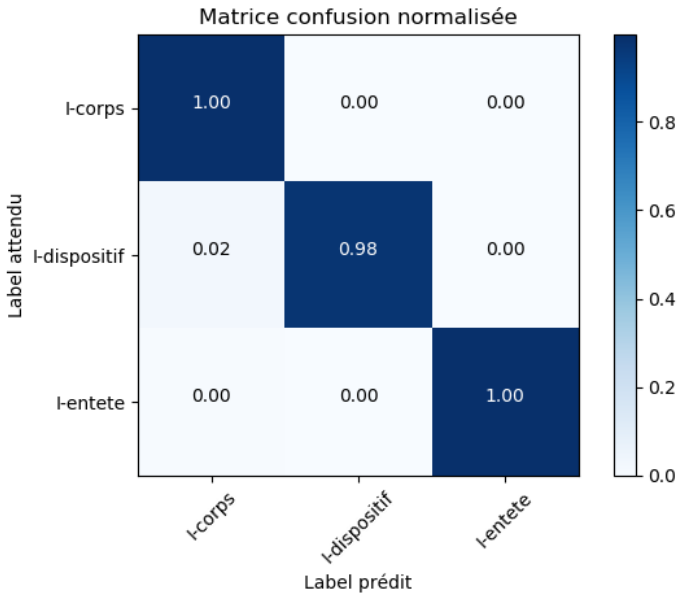



Figure 1.7 – Matrice de confusion entre lignes des sections avec le modèle CRF

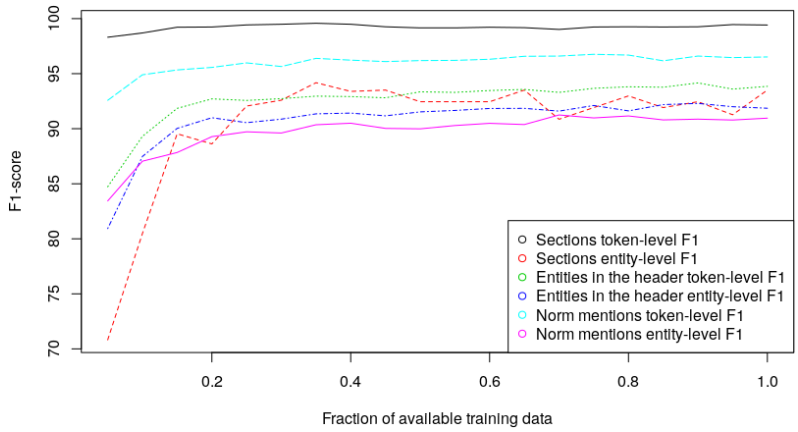
de séquences.

#### 1.4.5.4 Descripteurs manuels vs. réseau de neurones

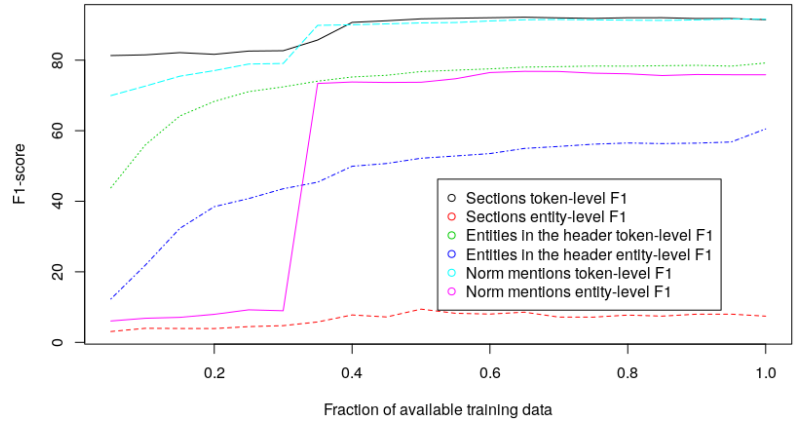
| | CRF + descripteurs manuels | | | BiLSTM-CRF | | |
|---|---|---|---|---|---|---|
| | *Precision* | *Rappel* | $F_1$ | *Precision* | *Rappel* | $F_1$ |
| **appelant** | 82.49 | 69.42 | 74.72 | 80.26 | 71.53 | 75.04 |
| **avocat** | 90.15 | 89.02 | 89.56 | 84.93 | 87.88 | 86.36 |
| **date** | 95.34 | 91.46 | 93.12 | 95.04 | 90.79 | 92.63 |
| **fonction** | 95.87 | 95.08 | 95.44 | 92.69 | 93.48 | 93.03 |
| **formation** | 96.91 | 91.31 | 93.7 | 91.05 | 89.47 | 89.84 |
| **intervenant** | 51.42 | 32.71 | 36.8 | 31.48 | 20 | 23.11 |
| **intime** | 76.01 | 79.15 | 77.22 | 67.7 | 75.43 | 70.83 |
| **juge** | 95.67 | 94.07 | 94.84 | 95.44 | 95.56 | 95.46 |
| **juridiction** | 98.55 | 98.25 | 98.33 | 97.95 | 99.22 | 98.57 |
| **rg** | 95.46 | 95.29 | 95.27 | 91.13 | 97.26 | 93.92 |
| **ville** | 98.33 | 93.01 | 94.71 | 91.43 | 95.34 | 93.3 |
| **norme** | 91.08 | 90.27 | 90.67 | 91.43 | 92.65 | 92.03 |
| **Evaluation globale** | 92.2 | 90.09 | 91.12 | 89.21 | 90.43 | 89.81 |

Tableau 1.7 – Comparaison entre le CRF avec des descripteurs définis manuellement et le BiLSTM-CRF au niveau entité.

L'ingénierie manuelle des caractéristiques est difficile car arbitraire.




(a) CRF




(b) HMM

Figure 1.8 – Évolution du score F1 en fonction de l'augmentation du nombre de données d'entraînement.

Nous avons comparé les performances de nos descripteurs avec celles des réseaux de neurones qui apprennent une représentation des segments. Pour cela nous avons choisi le BiLSTM-CRF de Lample *et al.* [2016] qui fait partie des meilleures approches récentes. La comparaison a été effectuée pour la détection des entités avec le schéma d'étiquetage BIEO et une validation croisée à 9 itérations. Le BiLSTM-CRF prend en entrée les plongements sémantiques Word2Vec [Mikolov *et al.* , 2013] des mots. Pour cela, nous avons entraîné des vecteurs de mots à partir d'un corpus jurisprudentiel de plus de 800K documents provenant de `www.legifrance.gouv.`

`fr` avec l'implémentation [5] de Le & Mikolov [2014a]. Les vecteurs obtenus ont une dimension de 300. Etant donné que les décisions sont des documents particulièrement longs, leur contenu a été découpé en des morceaux de texte dont la taille n'excède pas 300 mots. Les résultats obtenus par le BiLSTM-CRF sont assez proches de ceux que nous observons avec les descripteurs manuellement définis (Tableau 1.7 Page 30). Etant donné que ces derniers permettent de mieux détecter certaines entités comme les *intervenants*, les *avocats* ou les numéro *R.G.*, et vice-versa pour les *normes* ou les *appelants* chez le BiLSTM-CRF, une combinaison des deux types de descripteurs pourrait améliorer les résultats actuels.

## 1.5 Conclusion

L'application des modèles HMM et CRF dans le but de détecter des sections et des entités dans les décisions de justice est une tâche difficile. Ce chapitre a examiné les effets de divers aspects de la conception sur la qualité des résultats. En résumé, malgré une importante réduction du nombre de descripteurs, l'amélioration des résultats semble être insignifiante lorsque l'on sélectionne séparément la représentation du segment et le sous-ensemble de caractéristiques. Cependant, opter pour la bonne configuration en évaluant les approches de sélection combinées avec diverses représentations de segment pourrait peut-être offrir de meilleurs résultats. En raison de la longue durée de recherche du sous-ensemble optimal de descripteurs, il serait préférable d'utiliser un algorithme de sélection beaucoup plus rapide que les méthodes BDS et SFFS que nous avons expérimentées. De plus, même si les résultats s'améliorent avec l'augmentation de la taille de l'échantillon d'apprentissage, la mesure globale $F_1$ semble atteindre une limite très rapidement. Étant donné que certaines entités ne sont pas très bien détectées, il peut être avantageux d'ajouter des exemples appropriés afin de traiter ces problèmes spécifiques.

L'application des modèles pose deux difficultés majeures : l'annotation d'un nombre suffisant d'exemples et la définition de caractéristiques discriminantes. Les efforts d'annotation peuvent être réduits avec un système automatique à faible performance d'étiquetage. Il suffirait alors de vérifier manuellement ces annotations afin de corriger les erreurs commises par

5. `https://code.google.com/archive/p/word2vec/`

le système sur de nouvelles décisions à l'aide d'un outil d'aide à l'annotation. En ce qui concerne la définition des caractéristiques, dans la mesure où notre approche actuelle est réalisée manuellement par l'analyse de quelques documents, il est possible que de tels descripteurs ne s'adaptent pas parfaitement à un nouvel ensemble de données (différents pays, différentes langues, différentes juridictions). Pour éviter les énormes efforts requis pour définir les fonctionnalités manuellement, il serait préférable d'utiliser des descripteurs appris automatiquement à partir de corpus étiquetés ou non, comme des mots incorporés.

Il serait intéressant de poursuivre les travaux proposés sur la tâche de reconnaissance d'entités nommées. L'étude de modèles couplant les approches à descripteurs définis manuellement (e.g., CRF), avec des approches sans définition manuelle (de type apprentissage profond e.g., BiLSTM CRF) semble particulièrement intéressante. Une étude comparative approfondie des limitations des deux approches serait alors souhaitable. Bien que les approches à base d'apprentissage profond apparaissent (légèrement) moins performantes dans nos tests, l'étude de ces approches prometteuse est bien entendu à recommander. Des travaux sur l'impact des techniques de plongement lexical sur la performance de ces systèmes méritent notamment d'être menées.

Pour l'indexation des décisions dans une base de connaissances, il est aussi important de définir des méthodes de désambiguïsation et de résolution pour les entités à occurrences multiples, en plus de la correspondance des entités extraites avec des entités de référence, comme l'ont expérimenté Dozier *et al.* [2010] et Cardellino *et al.* [2017]. Ces travaux peuvent être poursuivis par d'autres applications telles que l'anonymisation automatique qui aiderait à publier plus rapidement l'énorme volume de décisions prononcées régulièrement.

# Chapitre 2

# Identification des demandes

## 2.1 Introduction

Introduire, chaque chapitre, de manière explicite par la contribution, discuter l'impacte de chaque méthode explorée, contribution sur la constitution des datasets, accompagnement des experts. dans un encadré

Au cœur de l'analyse des décisions de justice se trouve le concept de demande. Il s'agit d'une réclamation ou requête effectuée par une ou plusieurs parties aux juges. Une partie peut demander des dommages-intérêts en réparation d'un préjudice subi ou à l'issu d'un divorce, des indemnités auxquelles elle pense avoir droit, ou encore une étude d'expert, etc. Les demandes sont fondamentales car l'argumentation au cours d'une affaire a deux buts : faire accepter ses demandes, et faire rejeter celles de la partie adverse. L'extraction des demandes et des résultats correspondants, dans un corpus, permet ainsi de récolter des données informant de la manière dont sont jugés des types de demandes d'intérêt. Les informations qui nous intéressent sont la catégorie de la demande, le quantum (montant) demandé, le sens du résultat (par ex. la demande a-t-elle été acceptée ou rejetée ?), et le quantum obtenu (décidé par les juges). Pour pouvoir extraire les demandes et les résultats, il est nécessaire de comprendre comment ceux-ci sont exprimés et co-référencés dans les décisions jurisprudentielles. Leur énoncé peut comporter des expressions plus ou moins complexes, dont souvent des références à des jugements antérieurs, des agrégations ou des restrictions (Figure 2.1 Page 35).

Jennifer M., Catherine M. et Sandra M. ... demandent à la Cour de :
- les recevoir régulièrement appelantes incidentes du jugement du 23/05/2014 ;
- infirmer le dit jugement en toutes ses dispositions ; ...
Statuant à nouveau ...
- les condamner au paiement d'une somme de 3 000,00 € pour procédure abusive et aux entiers dépens ;

(a) Exemples d'énoncés de demandes

La cour, ...
CONFIRME le jugement entreprise en toutes ses dispositions.
Y ajoutant
CONSTATE que Amélanie Gitane P. épouse M. est défaillante à rapporter la preuve d'une occupation trentenaire lui permettant d'invoquer la prescription acquisitive de la parcelle BH 377 située [...].
DEBOUTE Amélanie Gitane P. épouse M. de sa demande en dommages et intérêts.
CONDAMNE Amélanie Gitane P. épouse M. aux dépens d'appel.
DIT n'y avoir lieu à l'application de l'article 700 du Code de Procédure Civile.

(b) Exemple d'énoncés de résultats

Figure 2.1 – Enoncés simples, ou comprenant des références et des agrégations (extraits de la décision 14/01082 de la cour d'appel de Saint-Denis (Réunion))

## 2.1.1 Données cibles à extraire

### 2.1.1.1 Catégorie de demande

Une catégorie $c$ de demande regroupe les prétentions qui sont de même nature par le fait qu'elles partagent deux aspects : l'objet demandé (par ex. dommages-intérêts, amende civile, déclaration de créance) et le fondement c'est-à-dire les règles ou normes ou principes juridiques qui fondent la demande (par ex. article 700 du code de procédure civile). Des noms particuliers sont utilisés pour identifier les catégories (Tableau 2.1).

### 2.1.1.2 Sens du résultat

Le sens du résultat est l'interprétation de la décision des juges sur une demande. Nous le notons $s_r$. En général, le sens peut être positif si la demande a été acceptée, et négatif si elle a été rejetée. Il arrive aussi que le résultat soit reporté à un jugement futur ; il s'agit dans ce cas d'un sursis à statuer.

### 2.1.1.3 Quantum demandé

Le quantum demandé quantifie l'objet de la demande. Nous le notons $q_d$. Par exemple, dans l'exemple de la Figure 2.1a, "3000 €" est le quantum

| Label | Expression nominative | Objet | Fondement |
|---|---|---|---|
| acpa | amende civile pour abus de procédure | amende civile | Articles 32-1 code de procédure civile + 559 code de procédure civile |
| concdel | dommages-intérêts pour concurrence déloyale | dommages-intérêts | Article 1382 du code civil |
| danais | dommages-intérêts pour abus de procédure | dommages-intérêts | Articles 32-1 code de procédure civile + 1382 code de procédure civile |
| dcppc | déclaration de créance au passif de la procédure collective | déclaration de créance | L622-24 code de commerce |
| doris | dommages-intérêts pour trouble de voisinage | dommages-intérêts | principe de responsabilité pour trouble anormal de voisinage |
| styx | frais irrépétibles | dommages-intérêts | Article 700 du code de procédure civile |

*Les labels ont été définis particulièrement pour cette étude, et par conséquent, ils n'existent pas dans le langage juridique.*

Tableau 2.1 – Exemples de catégories de demandes

demandé au titre des dommages-intérêts pour procédure abusive. Bien que cette étude ne porte que sur des sommes d'argent, le quantum peut être d'une autre nature comme par exemple une période dans le temps (garde d'enfant, ou emprisonnement, etc.). Toutes les catégories demandes n'ont pas de quantum (par ex. une demande de divorce) et seul le sens du résultat sera la donnée à extraire dans ce cas.

#### 2.1.1.4 Quantum obtenu ou résultat

Le quantum obtenu quantifie le résultat ou la décision des juges. Nous le notons $q_r$. Il ne peut qu'être inférieur ou égal au quantum demandé. Si la demande est rejetée, $q_r$ est nul même si cela n'est pas explicitement mentionné dans le document. A noter qu'il doit être de la même nature que le quantum demandé (somme d'argent ou durée).

### 2.1.2 Expression, défis et indicateurs d'extraction

Les demandes sont en général décrites à la fin de la section d'exposé des faits, procédures, moyens et prétentions des parties (section Litige). Elles rentrent donc dans les "moyens et prétentions des parties" qui regroupent les demandes et les arguments des parties. Quant aux résultats, ils sont décrits dans la section Dispositif et dans la section Motifs (raisonnement des juges). Les demandes sont exprimées dans différents paragraphes qui correspondent soit à une partie, soit à un groupe de par-

ties partageant les mêmes demandes (par ex. des époux). Les paragraphe sont parfois organisés en liste dont chaque élément exprime une ou plusieurs demandes, ou fait référence à un jugement antérieur. Les résultats ont aussi la forme de liste dans la section Dispositif. Par contre, dans les motifs de la décision, les raisonnements sont organisés en paragraphes, et ordonnés catégorie après catégorie. Le résultat est donné à la fin du groupe de paragraphes associé à la catégorie.

Cette pseudo-structure n'est pas standard et impose de nombreux défis à relever. En effet, une décision jurisprudentielle porte sur plusieurs demandes de catégories différentes ou similaires. Il est important de faire correspondre un quantum demandé extrait au sens et quantum du résultat qui font référence à la même demande. La séparation des demandes et des résultats rend difficile cette mise en correspondance. Ce problème peut aussi être causé par la redondance des quanta ; par exemple, les résultats exprimés dans les Motifs sont résumés dans le Dispositif. D'autre part, les références aux jugements antérieurs exigent de résoudre des références aux résultats de jugements antérieurs qui sont, généralement, rappelés dans le même document. Notons aussi que les difficultés liées aux agrégations (par ex. "*infirmer ... en toutes ces dispositions*") et aux restrictions/sélections (par ex. "*infirme le jugement ... sauf en ce qu'il a condamné M. A. ...*") méritent d'être résolues. Par ailleurs, les catégories de demandes sont nombreuses [1] mais ne sont pas toutes présentes à la fois dans les décisions. Tous ces aspects rendent difficile l'annotation manuelle des données de référence et la modélisation d'une approche d'extraction adéquate. Nous avons cependant identifié des indicateurs qui pourraient être utiles.

On pourrait au préalable annoter les candidats potentiels de quanta. Nous nous sommes intéressés aux demandes dont les quanta sont des sommes d'argent. Les mentions de somme d'argent sont généralement de la forme « `[valeur] [monnaie]` » (par ex. `3000 €`, `15 503 676 francs`, `un euro`, `339.000 XPF`). Des centimes apparaissent parfois (par ex. `dix huit euros et soixante quatorze centimes`, `26'977 € 19`). Ainsi, il est possible d'annoter les sommes d'argent à l'aide d'une expression régulière. Même s'il est difficile de reconnaître des sommes d'argent écrites en lettre, il faut remarquer que l'équivalent en chiffre est généralement mentionné tout près (par ex. `neuf mille cinq cent soixante six euros et quatre vingt sept centimes (9566,87 € )`).

1. plus de 500 selon la nomenclature des affaires civiles NAC+

La terminologie utilisée est aussi un bon indicateur pour reconnaître des demandes et des résultats. En effet, le vocabulaire utilisé est très souvent propre aux catégories de demandes. Par exemple le dernier élément de la Figure 2.1a comprend le terme "*pour procédure abusive*" qui est près d'une somme d'argent (*3000 €*) ; il est donc probable que ce type de terme assez particulier soit un bon indicateur de la position des quanta. Par ailleurs, des verbes particuliers sont utilisés pour exprimer les demandes et résultats : infirmer, confirmer, constater, débouter, dire ...

### 2.1.3 Formulation du problème

Nous avons tenu compte de deux principaux aspects du problème :

1. Une décision comprend plusieurs demandes de catégories similaires ou différentes ;
2. Il existe un grand nombre de catégories (500+) ; ce qui rend difficile l'annotation d'exemples de référence pour couvrir toutes ces catégories.

L'idée est de pouvoir ajouter progressivement de nouvelles catégories. Nous avons par conséquent opté pour une extraction par catégorie. Cette stratégie permet par ailleurs d'ajouter facilement de nouvelles classes sans avoir à redéfinir les classes déjà entraînées. Une exécution du système d'extraction permet ainsi d'extraire les demandes d'une seule catégorie. Le problème est décomposé en deux tâches :

**Tâche 1 :** Détecter les catégories présentes dans le document pour appliquer l'extraction uniquement à ces catégories ;

**Tâche 2 :** Pour chaque catégorie $c$ identifiée, extraire les demandes :

1. identification des valeurs d'attributs : quanta demandés ($q_d$), quanta obtenus ($q_r$), et sens du résultat ($s_r$) ;
2. mise en correspondance des attributs pour former les triplets ($q_d, s_r, q_r$) correspondants aux paires demande-résultat.

## 2.2 Travaux connexes

Chacune des tâches précédentes se rapproche d'une tâche couramment traitée en fouille de texte. En effet, la détection de catégories dans les dé-

cisions peut être modélisée comme un problème de classification de documents. La tâche d'extraction se rapproche plus quant à elle des problématiques comme l'extraction d'évènements, le remplissage de champs, ou encore l'extraction de relations et la résolution de référencement.

### 2.2.1 Extraction d'éléments structurés

Les demandes ressemblent aux structures telles que les relations ou les évènements. En effet, les champs définis par ACE [2008], pour les relations, et ACE [2005] pour les évènements, se rapprochent de ceux visés lors de l'extraction des demandes comme l'illustre le Tableau 2.2. Plus précisément, une catégorie de demandes correspond à un type d'évènement ou de relation entre deux entités. Les arguments qui participent à l'évènement « demande » ou à la relation « demande-résultat » sont le quantum demandé et le quantum résultat. Le sens du résultat représente la classe de la structure « demande ».

| | **Relation [ACE, 2008]** | **Événement [ACE, 2005]** | **Analogie chez les demandes** |
|---|---|---|---|
| **Type** | Org-Aff.Student-Alum | Die | Catégorie="Dommages-intérêts pour procédure abusive" |
| **Passage** (*extend*) | *Card graduated from the University of South Carolina* | "Il est mort hier d'une insuffisance rénale." | (*Figure 2.1*) |
| **Déclencheur (*trigger*)** | - | "mort" | "procédure abusive" |
| **Participants ou Arguments (*arguments*)** | Arg1="Card" Arg2="the University of South Carolina" | Victim-Arg="il" Time-Arg="hier" | Quantum-demandé="3000€" Quantum-obtenu="0 €" |
| **Classes (*attributes*, *classes*)** | Asserted | Polarity=POSITIVE, Tense=PAST | Sens-résultat="Rejeté" |

Tableau 2.2 – Exemples d'analogie entre relations, évènements et demandes

### 2.2.2 Approches d'extraction d'éléments structurés

L'extraction d'éléments structurés repose généralement sur une approche modulaire du problème qui le décompose en tâches plus simples. D'une part, on dispose de l'identification des déclencheurs [2] et des arguments.

2. terme-clé indiquant la présence d'un évènement [ACE, 2005].

D'autre part, une mise en correspondance relie les arguments et déclencheurs qui participent à la même relation ou au même évènement. Les classes peuvent être déterminées par classification du passage associé. Cette décomposition a permis à de nombreuses méthodes de voir le jour.

L'approche traditionnelle consiste en une chaîne de traitements enchaînant des modules adaptés à une tâche simple. La sortie d'une étape est l'entrée de la suivante. C'est ainsi que Ahn [2006] définit un enchaînement de modèles de classification (k-plus-proches-voisins [Cover & Hart, 1967] vs. classificateur d'entropie maximum [Nigam *et al.* , 1999]), pour extraire des champs d'évènements dans le corpus d'ACE [ACE, 2005]. Bien que les différents modules soient plus faciles à développer, ce type d'architecture souffre de l'accumulation et la propagation d'erreurs d'une étape à la suivante, ainsi que de la non exploitation de l'interdépendance entre les tâches. Par conséquent, l'inférence jointe des champs est préconisée. Celle-ci peut-être réalisée par une modélisation graphique probabiliste ou neuronale. Par exemple, pour l'extraction d'évènements, Yang & Mitchell [2016] estiment la probabilité conditionnelle jointe du type d'entité $t_i$, les rôles des arguments $r_{i.}$ et les types d'entités qui remplissent ces rôles $a.$ : $p_\theta(t_i, r_{i.}, a.|i, N_i, x)$, $i$ étant un déclencheur candidat, $N_i$ l'ensemble des entités candidates qui sont de potentiels arguments pour $i$, et $x$ est le document. Par ailleurs, Nguyen *et al.* [2016] illustrent l'utilisation des réseaux de neurones profonds avec une couche pour la prédiction du déclencheur, une autre pour le rôle des arguments, et la dernière encode la dépendance entre les labels de déclencheurs et les rôles d'arguments. [PERFORMANCE DE LEUR METHODE]

L'annotation d'ACE [2005] est un marquage des champs dans le texte, et par conséquent, la position ou l'occurrence des champs est indiquée (« annotation au niveau du segment de mots »). Comme dans notre cas, les données peuvent être annotées dans un tableau, hors des textes d'où elles sont issues. Il est donc nécessaire de retrouver leur position sans supervision. Palm *et al.* [2017] proposent dans cette logique une architecture de réseaux de neurones point-à-point qu'ils ont expérimentés sur des corpus de requêtes de recherche de restaurant et films [Liu *et al.* , 2013] ou de réservation de billets d'avion [Price, 1990]. Ils se sont intéressés au problème de remplissage de champs en apprenant la correspondance entre les textes et les valeurs de sorties. Leur modèle est basé sur les réseaux de pointeurs [Vinyals *et al.* , 2015] qui sont des modèles séquence-à-séquence avec attention, dans lesquels la sortie est une position de la séquence d'entrée.

Le modèle proposé consiste en un encodeur de la phrase et des contextes, plusieurs décodeurs (un pour chaque champ). L'application de cette architecture à l'extraction des demandes serait confrontée à deux obstacles majeurs auxquels il faut répondre au préalable. Premièrement, les décisions judiciaires ont des contenus de plusieurs centaines à plusieurs milliers de lignes contrairement aux requêtes manipulées par Palm *et al.* [2017] dont la plus longue ne comprend que quelques dizaines de mots. La complexité des architectures neuronales de TALN augmente rapidement en espace et en temps avec la longueur des documents manipulés. Deuxièmement, nous disposons de très peu de données annotées ; entre 23 et 198 documents annotés dans notre cas contre plusieurs milliers pour les expérimentations de Palm *et al.* [2017].

L'avantage de l'utilisation des réseaux de neurones vient de leur capacité à apprendre automatiquement des caractéristiques pertinentes contrairement aux modèles probabilistes qui exigent très souvent une ingénierie manuelle des caractéristiques. Par contre, il est beaucoup plus facile d'utiliser les modèles probabilistes sur des corpus de faible taille et de longs textes comme c'est le cas pour notre problème.

### 2.2.3 Extraction de la terminologie d'un domaine

L'identification des attributs peut être facilitée grâce à leur proximité avec des termes-clés caractéristiques des catégories de demandes au même titre que les « déclencheurs » aident à identifier les évènements. Ne disposant pas au préalable de la liste des termes pertinents pour l'extraction des demandes, il est possible de les apprendre. Il existe à cet effet plusieurs métriques statistiques de pondération de termes généralement employées en recherche d'information et en classification de texte comme méthodes de sélection de caractéristiques. Ces métriques sont qualifiées de poids globaux car calculées à partir des occurrences dans un corpus, à la différence des poids locaux (Tableau 2.4) calculés à partir des occurrences dans un document. Quelques métriques sont formulées ici en utilisant les notations du Tableau 2.3 définies pour une base d'apprentissage.

#### 2.2.3.1 Métriques non-supervisées

Les métriques non-supervisées affectent un score à un terme en rapport avec l'importance de ce dernier dans le corpus global $D$. Parmi ces

| Notation | Description |
|---|---|
| $t$ | un terme |
| $d$ | un document |
| $\|t\|$ | longueur de $t$ (nombre de mots) |
| $c$ | la catégorie (domaine ciblé) |
| $\overline{c}$ | la classe complémentaire ou négative |
| $D$ | ensemble global des documents de taille $N = \|D\|$ |
| $D_c$ | ensemble des documents de $c$ de taille $\|D_c\|$ |
| $D_{\overline{c}}$ | ensemble des documents de $\overline{c}$ de taille $\|D_{\overline{c}}\|$ |
| $N$ | nombre total de documents |
| $N_t$ | nombre de documents contenant $t$ |
| $N_{\overline{t}}$ | nombre de documents ne contenant pas $t$ |
| $N_{t,c}$ | nombre de documents de $c$ contenant $t$ |
| $N_{\overline{t},c}$ | nombre de documents de $c$ ne contenant pas $t$ |
| $N_{t,\overline{c}}$ | nombre de documents de $\overline{c}$ contenant $t$ |
| $N_{\overline{t},\overline{c}}$ | nombre de documents de $\overline{c}$ ne contenant pas $t$ |
| $DF_{t\|c}$ | proportion de documents contenant $t$ dans le corpus de $c$ ($DF_{t\|c} = \frac{N_{t,c}}{\|D_c\|}$) |
| $DF_{c\|t}$ | proportion de documents appartenant à $c$ dans l'ensemble de ceux qui contiennent $t$ |

Tableau 2.3 – Notation utilisée pour formuler les métriques

métriques, on retrouve par exemple la fréquence inverse de document (*inverse document frequency*) $idf$ [Sparck Jones, 1972] et ses variantes $pidf$ [Wu & Salton, 1981] et $bidf$ [Jones *et al.* , 2000] accordent plus d'importance aux termes rares. Elles considèrent en fait qu'un terme rare est plus efficace pour la distinction entre des documents. Par conséquent, elles sont efficaces en recherche d'information mais moins indiquées en classification de textes où le but est plutôt de séparer des catégories [Wu *et al.* , 2017]. Elles se formulent comme suit :

$$idf(t) = \log_2\left(\frac{N}{N_t}\right), pidf(t) = \log_2\left(\frac{N}{N_t} - 1\right), bidf(t) = \log_2\left(\frac{N_{\overline{t}} + 0.5}{N_t + 0.5}\right)$$

Il est possible de prendre explicitement en compte le fait que les termes peuvent comprendre plusieurs mots (n-grammes) et avoir des tailles différentes (nombre de mots). La C-value [Frantzi *et al.* , 2000], par exemple,

distingue la fréquence du terme et de ses sous-termes (termes imbriqués) par la formule :

$$\text{C-value}(t) = \begin{cases} \log_2(|t|) \cdot (N_t - \frac{1}{|T_t|} \cdot \sum_{b \in T_t} N_b), & \text{si } t \text{ est imbriqué} \\ \log_2(|t|) \cdot N_t, & \text{sinon,} \end{cases}$$

$T_t$ étant l'ensemble des termes candidats qui contiennent $t$.

#### 2.2.3.2 Métriques supervisées

Les métriques supervisées mesurent l'information contenue dans les labels des documents de la base d'apprentissage. Pour un terme $t$, elles expriment généralement la différence de proportion qui existe entre les occurrences de $t$ dans $D_c$ et ses occurrences dans $D_{\bar{c}}$. Elles sont ainsi mieux adaptées à la distinction entre catégories. Parmi les nombreuses métriques existantes, nous avons expérimenté les suivantes :

**La différence de fréquence** $\Delta_{DF}$ consiste simplement à calculer la différence entre les proportions de documents contenant $t$ respectivement dans $c$ et $\bar{c}$ :

$$\Delta_{DF}(t, c) = DF_{t|c} - DF_{t|\bar{c}}$$

**Le gain d'information** $ig$ [Yang & Pedersen, 1997] estime la quantité d'information apportée par la présence ou l'absence d'un terme $t$ sur l'appartenance d'un document à une classe $c$ :

$$ig(t, c) = \begin{array}{l} \frac{N_{t,c}}{N} \log_2 \left( \frac{N_{t,c} N}{N_t} \right) + \frac{N_{\bar{t},c}}{N} \log_2 \left( \frac{N_{\bar{t},c} N}{N_{\bar{t}} |D_c|} \right) \\ \quad + \frac{N_{t,\bar{c}}}{N} \log_2 \left( \frac{N_{t,\bar{c}} N}{N_t |D_{\bar{c}}|} \right) + \frac{N_{\bar{t},\bar{c}}}{N} \log_2 \left( \frac{N_{\bar{t},\bar{c}} N}{N_{\bar{t}} |D_c|} \right) \end{array}$$

**La fréquence de pertinence** $rf$ [Lan *et al.* , 2009] a comme intuition de considérer que plus la fréquence d'un terme $t$ est élevée dans $D_c$ relativement à sa fréquence dans $D_{\bar{c}}$, plus il contribue à distinguer les documents de $c$ de ceux de $\bar{c}$. Elle est calculée par la formule :

$$rf(t, c) = \log \left( 2 + \frac{N_{t,c}}{max(1, N_{t,\bar{c}})} \right)$$

**Le coefficient du $\chi^2$** [Schütze *et al.* , 1995] estime le manque d'indépendance entre $t$ et $c$. Par conséquent, une grande valeur de $\chi^2(t,c)$ indique une relation étroite entre $t$ et $c$. Elle est calculée par la formule :

$$\chi^2(t,c) = \frac{N((N_{t,c}N_{\bar{t},\bar{c}}) - (N_{t,\bar{c}}N_{\bar{t},c}))^2}{N_t N_{\bar{t}} |D_c||D_{\bar{c}}|}$$

**Le coefficient de corrélation** *ngl* de Ng, Goh et Low [Ng *et al.* , 1997] est la racine carré positive du $\chi^2$ [Schütze *et al.* , 1995] :

$$ngl(t,c) = \frac{\sqrt{N}|(N_{t,c}N_{\bar{t},\bar{c}}) - (N_{t,\bar{c}}N_{\bar{t},c})|}{\sqrt{N_t N_{\bar{t}} |D_c||D_{\bar{c}}|}}.$$

L'intuition est de ne regarder que les termes qui proviennent de $D_c$ et qui indiquent l'appartenance à $c$.

**Le coefficient** *gss* de Galavotti, Sebastiani, et Simi [Galavotti *et al.* , 2000] est une fonction simplifiée du *ngl* [Ng *et al.* , 1997] :

$$gss(t,c) = (N_{t,c}N_{\bar{t},\bar{c}}) - (N_{t,\bar{c}}N_{\bar{t},c}).$$

Le facteur $N$ a été éliminé car il est le même pour tous les termes. Le facteur $\sqrt{N_t N_{\bar{t}}}$ est supprimé car il accentue les termes extrêmement rares qui ne sont pas efficaces pour la classification de textes. Le facteur $\sqrt{|D_c||D_{\bar{c}}|}$ est éliminé car il accentue les catégories extrêmement rares, ce qui tend à réduire l'efficacité micro-moyennée (efficacité calculée globalement sur le corpus de test sans distinction du label des éléments).

**Le test de Marascuilo** (*mar*) qui se calcule par la formule :

$$mar(t,c) = \frac{\left( \begin{array}{l} (N_{t,c} - N_t N_{t,c}/N)^2 \\ \quad + (N_{t,\bar{c}} - N_t |D_{\bar{c}}|/N)^2 \\ \quad\quad + (N_{\bar{t},c} - |D_c| N_{\bar{t}}/N)^2 \\ \quad\quad\quad + (N_{\bar{t},} - N_{\bar{t}} |D_{\bar{c}}|/N)^2 \end{array} \right)}{N}$$

C'est un test de proportion multivariée. Nous proposons de l'utiliser pour tester la présence d'un terme $t$ dans différents corpus. Autrement dit, il s'agit de tester l'homogénéité des textes du corpus

contenant $c$. Lorsque $mar(t,c) \geq 3.84$ on accepte l'hypothèse selon laquelle la proportion de textes pour lesquels $t$ prédit $c$ est significative avec un risque d'erreur de 5%.référence**

**Le « delta lissé d'$idf$ »** , $dsidf$ [Paltoglou & Thelwall, 2010], est une version lissée du delta $idf$ ($didf$) de Martineau *et al.* [2009] ($didf(t,c) = \log_2\left(\frac{|D_{\bar{c}}|N_{t,c}}{|D_c|N_{t,\bar{c}}}\right)$). $dsidf$ se formule comme suit :

$$dsidf(t,c) = \log_2\left(\frac{|D_{\bar{c}}|(N_{t,c}+0.5)}{|D_c|(N_{t,\bar{c}}+0.5)}\right)$$

.

**Le delta BM25 d'$idf$** , $dbidf$ [Paltoglou & Thelwall, 2010], est une autre variante plus sophistiquée du $didf$ qui se calcule comme suit :

$$dbidf(t,c) = \log_2\left(\frac{(|D_{\bar{c}}| - N_{t,\bar{c}}+0.5)|(N_{t,c}+0.5)}{(|D_c| - N_{t,c}+0.5)(N_{t,\bar{c}}+0.5)}\right)$$

#### 2.2.3.3 Discussions

A l'exception de la C-value, ces métriques ne tiennent pas explicitement compte de la taille des termes dans les situations où on souhaiterait manipuler des termes de tailles différentes. Brown [2013] propose que soit affecté à un n-gramme $t$ le poids $\left(\frac{N_t}{N}\right)^{0.27}|t|^{0.09}$, une formule obtenue empiriquement pour l'identification du langage d'un document. Par ailleurs, la méthode C-value [Frantzi *et al.* , 2000] propose un produit similaire avec le logarithme de la longueur à la place des puissances. Il est par conséquent évident que le produit lissé de la longueur du terme (puissance ou logarithme) avec les métriques décrites précédemment, permet de favoriser les longs termes qui, bien que rares, sont très souvent plus pertinents que certains termes plus courts. Aussi, le temps pour calculer ces différentes métriques devient rapidement long, surtout pour des $n$-grammes de mots de taille variée (nombre de mots). Pour compter rapidement les occurrences des n-grammes des corpus, nous avons utilisé la librairie SML [3] [Harispe *et al.* , 2013] lors des expérimentations.

3. `http://www.semantic-measures-library.org`

## 2.3 Méthode

### 2.3.1 Détection des catégories par classification

Étant donné l'ensemble $D_{\bar{c}}$ des documents ne comprenant aucune demande de la catégorie d'intérêt $c$, nous proposons de modéliser la tâche de détection des catégories en une tâche de classification de documents. Pour chaque catégorie $c$, un modèle de classification binaire est entraîné pour déterminer si un document $d$ contient une demande de la catégorie $c$. Nous avons particulièrement expérimenté quatre algorithmes traditionnellement utilisés comme approches de base. Il s'agit du Bayésien Naïf, de l'arbre de décision C4.5, des $k$-plus-proches-voisins [Cover & Hart, 1967], de la machine à vecteurs de support (SVM). Ces algorithmes sont décrits en détail dans le chapitre 3 qui est axé sur la classification des documents. Les labels utilisés correspondent aux catégories d'intérêt. Par exemple, un document sera labellisé *danais* s'il contient des demandes de dommages-intérêts pour abus de procédure, et *nodanais* sinon. Chaque document $d$ est représenté sous une forme vectorielle du type TF-IDF (*term frequency - inverse document frequency*) proposé par Salton & Buckley [1988] dont chaque dimension $k$ est identifiée par un terme $t_k$. Le poids $w(t_k, d)$ affecté à ce dernier est le produit normalisé d'un poids global $g(t_k)$ au corpus du mot et d'un poids local $l(t_k, d)$ de $t_k$ dans le document $d$ : $w(t_k, d) = l(t, d) \times g(t) \times nf(d)$, où $nf$ est un facteur de normalisation tel que la norme cosinus $\sqrt{\sum_k (w(t_k, d))^2}$ Pourquoi elle est appelée "norme cosinus" qui est généralement utilisée.

| Description | Formule |
|---|---|
| Décompte brute du terme [Salton & Buckley, 1988] | $tf(t,d) =$ nombre d'occurrences de $t$ dans $d$ |
| Présence du terme [Salton & Buckley, 1988] | $tp(t,d) = \begin{cases} 1 & \text{, si } tf(t,d) > 0 \\ 0 & \text{, sinon} \end{cases}$ |
| Normalisation logarithmique | $logtf(t,d) = 1 + \log(tf(t,d))$ |
| Fréquence augmentée et normalisée du terme [Salton & Buckley, 1988] | $atf(t,d) = k + (1-k)\frac{tf(t,d)}{\max_{t \in T} tf(t,d)}$ |
| Normalisation basée sur la fréquence moyenne du terme [Manning *et al.* , 2009b] ( avg représente la moyenne) | $logave(t,d) = \frac{1+\log tf(t,d)}{1+\log \operatorname{avg}_{t \in T} tf(t,d)}$ |

Tableau 2.4 – Métriques locales

Etant donné le grand nombre de métriques de pondération existantes, la métrique choisie est celle qui fournit la meilleure performance sur les données d'apprentissage.

### 2.3.2 Extraction basée sur la proximité entre sommes d'argent et termes-clés

Diverses approches d'extraction d'informations existent (§ 2.2.2). Il parait important de proposer dans un premier temps une approche basique explorant la solvabilité du problème du fait de ses multiples spécificités dont l'annotation d'une seule catégorie dans un document qui en contient plusieurs, l'annotation dans un tableau et donc à l'extérieur du document, la très faible quantité des données annotées, la multiplicité des demandes et des catégories dans un même document. Par conséquent, nous proposons ici une chaîne d'extraction à base de termes-clés, applicable pour chaque catégorie de demande. Il s'agit d'une approche qui tente de reproduire une lecture naïve du document en se basant sur des expressions couramment employées pour énoncer les demandes et résultats. La méthode consiste en deux phases dont une phase d'apprentissage des termes-clés de la catégorie, à proximité desquels seront identifiés les attributs durant la phase d'extraction des demandes comme l'illustre la Figure 2.2. On remarque en effet que, naïvement, le seul fait que 1500 euros soit aussi proche des termes-clés *amende civile* et *pour procédure abusive* signifie bien qu'il s'agit du quantum demandé comme amende civile pour procédure abusive.

" ...
- débouter M. S. de ...
**- le condamner à payer une amende civile de 1.500 euros pour procédure abusive** ...
- le condamner à payer la somme ..."

(a) Extrait original d'un énoncé de demande avant marquage

" ...
- débouter M. S. de ...
- le **<demande categorie="acpa">**condamner à payer une <terme-clef categorie="acpa">**amende civile**</terme-clef> de <argent> **1.500 euros** </argent> <terme-clef categorie="acpa"> **pour procédure abusive**</terme-clef> ...
- le**</demande>** condamner à payer la somme ..."

(b) Énoncé, sommes d'argent, et termes-clés marqués

Figure 2.2 – Illustration de la proximité des quantas et termes-clés

#### 2.3.2.1 Pré-traitement

Le pré-traitement est nécessaire pour :

1. sectionner le document comme décrit au chapitre 1 en sections Entête, Litige, Motifs, Dispositif ;
2. annoter les sommes d'argent (en chiffre) à l'aide de l'expression régulière « `[0-9]([0-9]|['',.]|\s)*\s*([Ee]uro[s]{0,1} |franc[s]{0,1} |€|F|XPF|CFP|EUR|EUROS|[i])( |$)` » ;
3. annoter les énoncés de demandes et de résultats respectivement dans les sections Litige et Dispositif. Pour cela, les mots introductifs du tableau 2.5 sont employés car ils indiquent le début d'un énoncé indépendamment de la catégorie.

| **Demande** | **Résultat** (organisé par polarité ou sens) | | |
|---|---|---|---|
| | **accepte** | **sursis à statuer** | **rejette** |
| *accorder, admettre, admission, allouer, condamnation, condamner, fixer, laisser, prononcer, ramener, surseoir* | *accorde, accordons, admet, admettons, alloue, allouons, condamne, condamnons, déclare, déclarons, fixe, fixons, laisse, laissons, prononce, prononçons* | *réserve, réservons, sursoit, sursoyons* | *déboute, déboutons, rejette, rejetons* |

Tableau 2.5 – Mots introduisant les énoncés de demandes et de résultats

La recherche de passages à l'aide de listes de termes est une technique souvent utilisée dans les décisions de justice, à l'exemple de Wyner [2010] qui utilise des termes similaires à ceux du Tableau 2.5 pour annoter les énoncés de résultats (toute phrase contenant un terme de jugement) : *affirm, grant, deny, reverse, overturn, remand, ...*

#### 2.3.2.2 Apprentissage des termes-clés d'une catégorie

Les termes-clés sont identifiés à l'aide de méthodes statistiques d'extraction ou sélection de terminologie. La base d'apprentissage comprend les corpus $D_c$ et $D_{\overline{c}}$ dont les documents ont été pré-traités. Le processus d'apprentissage des termes se déroule comme suit :

1. restreindre le contenu de chaque document de $D_c$ à la concaténation des énoncés de demande et résultats contenant des sommes d'argent de valeur égale à celle des quanta annotés.
2. restreindre chaque document de $D_{\overline{c}}$ à la concaténation des énoncés de demande et résultats contenant des sommes d'argent.

3. à l'aide d'une métrique global $g$, calculer le score des termes du corpus $D_c \cup D_{\bar{c}}$. Ce score est multiplié par le logarithme de la longueur du terme pour favoriser les termes longs : $g'(t,c) = \log_2(|t|) \times g(t,c)$.
4. normaliser les scores en appliquant à chaque score original ($g'(t,c)$) la formule $g'_{norm}(t,c) = \frac{g'(t,c) - \min_{t_k}(g'(t_k,c))}{\max_{t_k}(g'(t_k,c)) - \min_{t_k}(g'(t_k,c))}$.
5. trier par ordre décroissant des termes ;
6. sélectionner les premiers termes qui obtiennent les meilleurs performances sur la base d'apprentissage .

### 2.3.3 Application de l'extraction à de nouveaux documents

A l'aide des termes-clés appris, l'extraction des données de couples demandes-résultats se déroule comme suit :

1. reconnaître et marquer les occurrences des termes dans le document ;
2. extraire les quanta demandés ($q_d$) et résultats ($q_r$) à proximité des termes-clés respectivement dans les énoncés de demande et résultat qui contiennent des sommes d'argent et un terme-clé ;
3. le mot introductif de l'énoncé résultat indique le sens du résultat ($s_r$) tel que catégorisé dans le Tableau 2.5 ;
4. relier les attributs ($q_d, s_r, q_r$) de chaque paire demande-résultat :
   (a) former les paires (énoncé de demande, énoncé de résultat) similaire (nous utilisons la métrique de « la plus longue sous-séquence commune » [Bakkelund, 2009])
   (b) pour chaque paire d'énoncés formée, relier les quanta demandés et quanta résultats par ordre d'occurrence similaire.

## 2.4 Résultats expérimentaux

Nous analysons ici la capacité de l'approche proposée à reconnaître efficacement les catégories de demandes présentes dans les documents, et à extraire les valeurs des attributs des différentes paires demandes-résultats

qui y sont exprimées. Sont discutés les données et métriques d'évaluation employées, ainsi que des résultats expérimentaux observés avec des exemples annotés pour les six catégories du Tableau 2.1.

### 2.4.1 Données d'évaluation

L'annotation manuelle d'exemples s'effectue pour une catégorie à la fois afin que la tâche soit plus facile pour les experts. Le protocole d'annotation se déroule en 3 étapes :

1. définir une catégorie $c$ par son objet et sa norme juridique ;
2. former un corpus $D_c$ de documents contenant des demandes de $c$, et un autre $D_{\bar{c}}$ de documents n'en contenant pas ;
3. extraire toutes les demandes de catégories $c$ mentionnées dans $D_c$, pour annoter les données des paires demande-résultat dans un tableau comme celui illustré par le Tableau 2.6 ;

| | A | B | C | D | F | H | L | N |
|---|---|---|---|---|---|---|---|---|
| 1 | IDENTIFICATION DE LA DECISION | | | DESCRIPTION DE LA PRETENTION | | | DESCRIPTION DU RESULTAT | |
| 2 | Type | Ressort | RG | OBJET | NORME | QUANTUM | RESULTAT | QUANTUM RESULTAT (obtenu) |
| 441 | CA | Lyon | 14/06911 | dommages-intérêts | 700 Code de Procédure Civile | 3,500.00 € | rejette | 0.00 € |
| 442 | CA | Lyon | 14/06911 | dommages-intérêts | 700 Code de Procédure Civile | 2,000.00 € | accepte | 1,500.00 € |

Les noms des champs sont sur les 2 premières lignes et les demandes sont données en exemple pour la catégorie *dommages-intérêts sur le fondement de l'article 700 du code de procédure civile* (décision 14/06911 de la cour d'appel de Lyon).

Tableau 2.6 – Extrait du tableau d'annotations manuelles des demandes.

La répartition des données d'évaluation est donnée par la Figure 2.3.

Il faut aussi noter que bien que l'annotation manuelle des demandes et des résultats soit réalisée dans un tableau (annotation externe au contenu), elle reste une tâche très difficile. Le très faible nombre de documents annotés manuellement en témoigne. Le nombre maximum de documents annotés pour une catégorie est seulement de 198 (barres vertes de *danais*).

### 2.4.2 Métriques d'évaluation

**Reconnaissance de catégories par classification** La classification des documents est évaluée en utilisant les métriques précision (P), rappel (P),

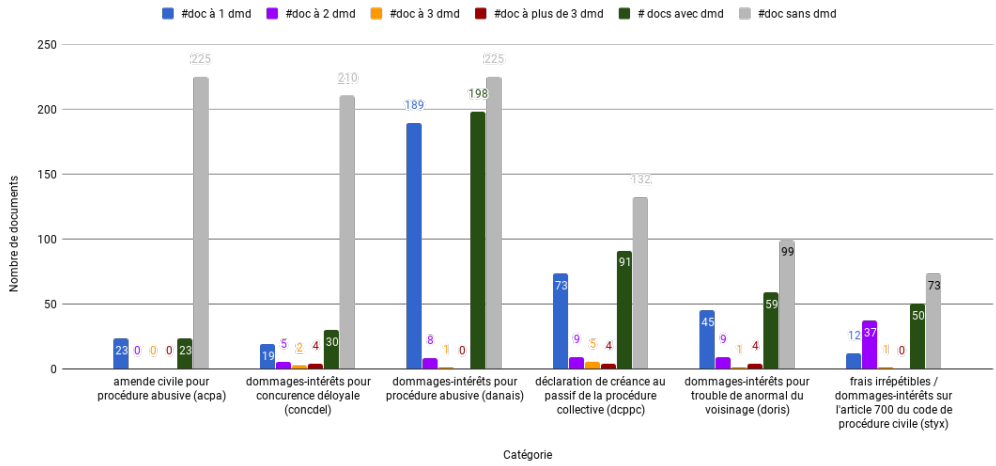



Figure 2.3 – Répartitions des demandes dans les documents annotées.

f1-mesure (F1) calculées à l'aide des nombres de vrais positifs (TP), faux positifs (FP), faux négatifs (FN) comme suit :

$$P = \frac{TP}{TP+FP} \qquad R = \frac{TP}{TP+FN} \qquad F1 = 2 \times \frac{P \times R}{P+R}$$

**Extraction des attributs des paires demande-résultat** Nous évaluons les approches proposées sur l'extraction de 3 données : le quantum demandé $q_d$, le sens du résultat $s_r$ et le quantum obtenu $q_r$. Une demande est donc un triplet $(q_d, s_r, q_r)$. Il est possible d'évaluer le système pour un sous-ensemble $x$ de $\{q_d, s_r, q_r\}$ sur les demandes extraites d'un corpus annotées $D$ de test. Nous utilisons les métriques traditionnellement employées en extraction d'information : la précision (Eq. 2.4.1), le rappel (Eq. 2.4.2), et la F1-mesure (Eq. 2.4.3).

$$Precision_{c,x,D} = \frac{TP_{c,x,D}}{TP_{c,x,D} + FP_{c,x,D}} \tag{2.4.1}$$

$$Rappel_{c,x,D} = \frac{TP_{c,x,D}}{TP_{c,x,D} + FN_{c,x,D}} \tag{2.4.2}$$

$$F1_{c,x,D} = 2 \times \frac{Precision_{c,x,D} \times Rappel_{c,x,D}}{Precision_{c,x,D} + Rappel_{c,x,D}} \tag{2.4.3}$$

Ces mesures sont définies à partir des nombres de vrais positifs ($TP$), faux positifs ($FP$) et faux négatifs ($FN$). Au niveau d'un document $d$ :

- $TP_{c,x,d}$ est le nombre de demandes extraites de $d$ par le système, qui sont effectivement de la catégorie $c$ ;
- $FP_{c,x,d}$ est le nombre de demandes extraites de $d$ par le système, mais qui ne sont pas des demandes de $c$ (demandes en trop) ;
- $FP_{c,x,d}$ est le nombre de demandes annotées comme étant de $c$ mais qui n'ont pas pu être extraites par le système (demandes manquées).

Au niveau d'un corpus d'évaluation $D$, ces métriques sont sommées : $TP_{c,x,D} = \sum_{d \in D} TP_{c,x,d}$ $FP_{c,x,D} = \sum_{d \in D} FP_{c,x,d}$ $FN_{c,x,D} = \sum_{d \in D} FN_{c,x,d}$.

Une donnée observée (par exemple « 3 000 € ») est bien extraite automatiquement si sa valeur (le nombre 3000) correspond à celle du quantum annoté dans le tableau. Nous considérons que les unités monétaires, entre les quanta extraits et ceux manuellement annotés, sont identiques.

### 2.4.3 Détection des catégories par classification

Les implémentations de la bibliothèque Weka [Frank *et al.* , 2016] ont permis d'utiliser plusieurs modèles de classification : le modèle Bayésien naïf (NB), l'arbre de décision C4.5 (implémenté sous l'appelation J48), les k-plus-proches-voisins (KNN), et le SVM. A chaque entraînement, s'exécute une sélection de modèles par validation croisée sur les données d'entraînement. Elle a pour but de sélectionner la métrique locale et la métrique globale appropriée. Les résultats obtenus par 5-*folds* validation croisée sont présentés sur le Tableau 2.7.

| | NB | | | J48 | | | KNN | | | SVM | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 |
| acpa | 1.0 | 1.0 | 1.0 | 0.996 | 0.955 | 0.972 | 1.0 | 1.0 | 1.0 | 0.996 | 0.955 | 0.972 |
| concdel | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.995 | 0.967 | 0.979 |
| danais | 0.988 | 0.989 | 0.988 | 0.996 | 0.995 | 0.995 | 0.995 | 0.995 | 0.995 | 0.993 | 0.993 | 0.993 |
| dcppc | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| doris | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| styx | 1.0 | 1.0 | 1.0 | 0.984 | 0.983 | 0.983 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |

P= Précision, R=Rappel, F1 = F1-mesure

Tableau 2.7 – Evaluation de la détection de catégories.

D'après les résultats, la tâche 1 est relativement aisée pour les algorithmes traditionnels qui détectent parfaitement la présence ou non d'une catégorie dans les documents. Par conséquent, pour toute catégorie $c$, les

résultats de l'extraction, dans la suite, ne sont discutés que pour les documents de $c$, car, grâce à l'efficacité de la phase de classification, aucun document de $\overline{c}$ ne sera traité par la phase d'extraction.

### 2.4.4 Extraction de données des paires demandes-résultats

Les scores des termes-clés candidats étant normalisés, si on sélectionne les termes dont les scores sont supérieurs à un seuil fixé, on remarque que chaque métrique d'extraction a un niveau d'efficacité différent entre les catégories de demande (Tableau 2.8 avec 0.5 comme seuil fixé).

| | *acpa* | *concdel* | *danais* | *dcppc* | *doris* | *styx* | **Moyenne** |
|---|---|---|---|---|---|---|---|
| $bidf$ | 37.33 | 32.73 | 23.96 | 20.46 | 8.08 | 28.43 | 25.17 |
| $\chi^2$ | **54.55** | 25.88 | 43.97 | 28.35 | 13.11 | 52.73 | 36.43 |
| $dbidf$ | 37.58 | 24.63 | 56.25 | **29.06** | 11.58 | 52.73 | 35.31 |
| $\Delta_{DF}$ | **54.55** | 25.55 | 48.16 | 28.1 | **19.64** | 52.73 | **38.12** |
| $dsidf$ | 37.58 | 25.25 | **56.42** | 26.05 | 8.72 | **53.46** | 34.58 |
| $gss$ | **54.55** | 25.11 | 48.16 | 28.1 | **19.64** | 52.73 | **38.05** |
| $idf$ | 38.78 | 32.73 | 22.31 | 20.53 | 8.27 | 25.22 | 24.64 |
| $ig$ | 4 | 12.4 | 45.21 | 14.99 | 16.74 | 51.13 | 24.08 |
| $marascuilo$ | **54.55** | 23.65 | 43.97 | 26.67 | 17.91 | 52.73 | 36.58 |
| $ngl$ | 42.02 | 23.97 | 52.31 | 27.21 | 13.29 | 53.2 | 35.33 |
| $pidf$ | 26.19 | **33.71** | 21.83 | 20.46 | 8.76 | 27.68 | 23.11 |
| $rf$ | 41.11 | 33.09 | 55.72 | 28.56 | 14.93 | 51.23 | 37.44 |

Tableau 2.8 – Comparaison des pondérations globales suivant la F1-mesure.

Par conséquent, la métrique et le seuil doivent être bien sélectionnés en fonction de la catégorie de demandes traitée. En choisissant, pour ces méta-paramètres, les valeurs les plus efficaces pour l'extraction sur la base d'apprentissage, les résultats du Tableau 2.9 sont observés.

Ces résultats détaillés font remarquer que les attributs, pris individuellement, présentent d'assez bonnes performances. Cependant, la mise en correspondance des attributs (triplet $(q_d, s_r, q_r)$) peine toujours à montrer des performances du même rang. On remarque néanmoins que les mesures-F1 $(q_d, s_r, q_r)$ sont proches de celles des attributs qui présentent le plus de difficulté. L'échec de l'extraction de ces attributs est une des principales causes des performances observées pour la liaison des attributs de paires similaires demande-résultat. Par ailleurs, les données sur le résultat, $s_r$ et $q_r$, sont en générale plus faciles à extraire que le quantum demandé $q_d$. Il est aussi bien de noter qu'une plus grande quantité d'exemples annotés de documents ne semble pas être la garantie d'une meilleure extraction. On remarque en effet que les meilleures performances sont obtenues pour la

| | | | Données d'entraînement | | | | Données de test | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $c$ | Données | $\|V_c\|$ | P | R | F1 | %Docs | P | R | F1 | %Docs |
| *acpa* | $q_d$ | 1 | 86.4 | 56.37 | 68.13 | 56.37 | 68.33 | 54 | 58.99 | 46 |
| | $q_r$ | 1 | 100 | 65.09 | 78.74 | 65.09 | 93.33 | 63 | 71.43 | 55 |
| | $s_r$ | 1 | 100 | 65.09 | 78.74 | 65.09 | 93.33 | 63 | 71.43 | 55 |
| | $(s_r, q_r)$ | 1 | 100 | 65.09 | 78.74 | 65.09 | 93.33 | 63 | 71.43 | 55 |
| | $(q_d, s_r, q_r)$ | 1 | 86.4 | 56.37 | 68.13 | 56.37 | 68.33 | 54 | 58.99 | 46 |
| *concdel* | $q_d$ | 26 | 49.33 | 44.02 | 45.31 | 24.17 | 73.2 | 29.72 | 33.29 | 26.67 |
| | $q_r$ | 26 | 48.3 | 42.66 | 44.1 | 22.5 | 75.73 | 28.89 | 34.3 | 26.67 |
| | $s_r$ | 26 | 46.52 | 40.89 | 42.36 | 22.5 | 74.93 | 26.39 | 33.09 | 26.67 |
| | $(s_r, q_r)$ | 26 | 46.52 | 40.89 | 42.36 | 22.5 | 74.93 | 26.39 | 33.09 | 26.67 |
| | $(q_d, s_r, q_r)$ | 26 | 42.43 | 37.41 | 38.68 | 20.83 | 68.27 | 23.06 | 28.65 | 23.33 |
| *danais* | $q_d$ | 37 | 77.71 | 48.71 | 59.68 | 37.3 | 79.25 | 47.5 | 59 | 37.3 |
| | $q_r$ | 37 | 77.68 | 48.71 | 59.67 | 37.03 | 77.78 | 46.46 | 57.79 | 36.22 |
| | $s_r$ | 37 | 77.05 | 48.33 | 59.19 | 37.03 | 77.78 | 46.46 | 57.79 | 36.22 |
| | $(s_r, q_r)$ | 37 | 77.05 | 48.33 | 59.19 | 37.03 | 77.78 | 46.46 | 57.79 | 36.22 |
| | $(q_d, s_r, q_r)$ | 37 | 74.45 | 46.65 | 57.16 | 35.81 | 74.41 | 44.38 | 55.23 | 34.59 |
| *dcppc* | $q_d$ | 35 | 45.71 | 36.64 | 40.66 | 34.05 | 44.64 | 40.73 | 41.75 | 31.4 |
| | $q_r$ | 35 | 78.99 | 63.21 | 70.2 | 59.33 | 75.48 | 64.51 | 68.41 | 53.82 |
| | $s_r$ | 35 | 84.73 | 67.85 | 75.33 | 63.24 | 81.21 | 69.14 | 73.51 | 57.43 |
| | $(s_r, q_r)$ | 35 | 78.99 | 63.21 | 70.2 | 59.33 | 75.48 | 64.51 | 68.41 | 53.82 |
| | $(q_d, s_r, q_r)$ | 35 | 34.2 | 27.39 | 30.41 | 28.03 | 31.66 | 28.55 | 29.41 | 25.37 |
| *doris* | $q_d$ | 8 | 31.98 | 35.76 | 32.94 | 7.75 | 37.48 | 35.9 | 36.63 | 7.12 |
| | $q_r$ | 8 | 35.73 | 39.72 | 36.69 | 8.63 | 39.43 | 38.47 | 38.89 | 7.12 |
| | $s_r$ | 8 | 35.06 | 39.56 | 36.24 | 9.06 | 42.91 | 41.44 | 42.12 | 8.94 |
| | $(s_r, q_r)$ | 8 | 32.61 | 36.16 | 33.45 | 8.2 | 38.14 | 37.04 | 37.54 | 7.12 |
| | $(q_d, s_r, q_r)$ | 8 | 24.48 | 27.16 | 25.13 | 5.61 | 29.7 | 28.53 | 29.08 | 7.12 |
| *styx* | $q_d$ | 4 | 69.34 | 59.55 | 64.04 | 33.5 | 69.3 | 59.49 | 63.61 | 32 |
| | $q_r$ | 4 | 75.87 | 65.17 | 70.08 | 31.5 | 74.86 | 64.08 | 68.63 | 28 |
| | $s_r$ | 4 | 75.87 | 65.17 | 70.08 | 31.5 | 74.86 | 64.08 | 68.63 | 28 |
| | $(s_r, q_r)$ | 4 | 75.87 | 65.17 | 70.08 | 31.5 | 74.86 | 64.08 | 68.63 | 28 |
| | $(q_d, s_r, q_r)$ | 4 | 57.61 | 49.44 | 53.19 | 25.5 | 57.24 | 48.36 | 52.08 | 24 |

P= Précision, R=Rappel, F1 = F1-mesure
%Docs : proportion de documents dont l'ensemble des données extraites est égale à l'attendu (documents parfaitement traités)
$|V_c|$ : nombre moyen de termes-clés identifiés pour la catégorie $c$

Tableau 2.9 – Résultats détaillés pour l'extraction des données avec sélection automatique de la méthode d'extraction des termes-clés

catégorie disposant du plus faible nombre d'exemples annotés (*acpa*) avec en moyenne un seul terme-clé appris.

### 2.4.5 Analyse des erreurs

En extraction d'éléments structurés, on retrouve trois types d'erreurs [Yang & Mitchell, 2016] : les données manquées (faux négatifs), les données en plus des attendues (faux positifs), et les mauvaises classifications (confusions). La confusion n'est pas discutée ici car les annotations ne sont faites que pour une seule classe.

Etant donné que la précision est en général supérieure au rappel, il est certain que les erreurs sont majoritairement dues aux données manquées comme le confirme le Tableau 2.10.

| | Données d'entraînement | | Données de test | |
|---|---|---|---|---|
| | **%erreurs FP** | **%erreurs FN** | **%erreurs FP** | **%erreurs FN** |
| $q_d$ | 36.90 | 63.10 | 36.52 | 63.48 |
| $q_r$ | 32.30 | 67.70 | 34.32 | 65.68 |
| $s_r$ | 31.72 | 68.28 | 34.11 | 65.89 |
| $(s_r, q_r)$ | 32.32 | 67.68 | 34.39 | 65.61 |
| $(q_d, s_r, q_r)$ | 37.77 | 62.23 | 37.72 | 62.28 |

Tableau 2.10 – Types et taux d'erreurs (pourcentage en moyenne sur les 6 catégories de demandes)

Trois raisons peuvent expliquer le fait que peu de données attendues soient extraites. Premièrement, certaines valeurs d'attributs ne sont pas mentionnées dans les sections Litige et Dispositif utilisées (pourcentages inférieurs à 100 dans les Tableaux 2.11 et 2.12). Par exemple, les quanta résultat de *doris* plus présents dans les Motifs que dans le Dispositif.

| | $\#q_d$ | $\#q_d \neq NUL$ | **# dans doc.** | **# dans Litige** | **# dans Motifs** | **# dans Dispositif** |
|---|---|---|---|---|---|---|
| acpa | 23 | 16 | 16 (100%) | 16 (100%) | 9 (56.25%) | 5 (31.25%) |
| concdel | 58 | 56 | 55 (98.21%) | 55 (98.21%) | 7 (12.5%) | 2 (3.57%) |
| danais | 208 | 182 | 182 (100%) | 179(100%) | 39 (21.43%) | 23 (12.64%) |
| dcppc | 126 | 126 | 122 (96.83%) | 109 (86.51%) | 71 (56.35%) | 65 (51.59%) |
| doris | 94 | 83 | 83 (100%) | 82 (98.80%) | 21 (25.30)% | 6 (7.23%) |
| styx | 89 | 86 | 86 (100%) | 86 (100%) | 12 (13.95%) | 9 (10.47%) |

*Les pourcentages ne sont calculés que pour les valeurs non nulles*

Tableau 2.11 – Taux de quanta demandés ($q_d$) mentionnés dans les documents annotés

| | $\#q_r$ | $\#q_r \neq NUL$ | **# dans doc.** | **# dans Litige** | **# dans Motifs** | **# dans Dispositif** |
|---|---|---|---|---|---|---|
| acpa | 23 | 6 | 6 (100%) | 3 (50%) | 6 (100%) | 5 (83.33%) |
| concdel | 58 | 8 | 8 (100%) | 2 (25%) | 8 (100%) | 6 (75%) |
| danais | 208 | 23 | 23 (100%) | 15 (65.22%) | 22 (95.65%) | 20 (86.96%) |
| dcppc | 126 | 76 | 75 (98.68%) | 55 (72.37%) | 56 (73.68%) | 64 (84.21%) |
| doris | 94 | 44 | 44 (100%) | 28 (63.64%) | 40 (90.91)% | 24 (54.55%) |
| styx | 89 | 30 | 29 (96.67%) | 16 (53.33%) | 22 (73.33%) | 29 (96.67%) |

*Les pourcentages ne sont calculés que pour les valeurs non nulles*

Tableau 2.12 – Taux de quanta accordés ($q_r$) mentionnés dans les documents annotés

Deuxièmement, la sélection des termes-clés n'est pas parfaite (Tableau 2.13). D'une part, l'ensemble sélectionné ne couvre pas toutes les situations d'expression de la catégorie (par exemple, pour la catégorie *styx*, le

terme « frais irrépétibles » est souvent utilisés à la place de « article 700 du code de procédure civile », mais dans très peu d'exemples annotés). D'autre part, certains termes sont trop spécifiques à la base d'apprentissage (par exemple, pour la catégorie *concdel*, des sommes d'argent et autres termes comme « condamner in solidum les sociétés » apparaissent dans la liste).

| Catégorie | Termes-clés appris |
|---|---|
| *acpa* | amende civile |
| *concdel* | titre de la concurrence déloyale, somme de 15000euros à titre, réparation de son préjudice financier, payer la somme de 15000euros, condamner in solidum les sociétés, agissements constitutifs de concurrence déloyale |
| *danais* | dommages et intérêts pour procédure, 32-1 du code de procédure, intérêts pour procédure abusive, titre de dommages-intérêts pour procédure, intérêts pour procédure, article 32-1 du code, dommages-intérêts pour procédure abusive |
| *dcppc* | admet la créance déclarée, admet la créance, passif de la procédure collective, passif de la procédure, hauteur de la somme, créance déclarée, titre chirographaire, admission de la créance, rejette la créance, |
| *doris* | préjudices, abusive, condamner solidairement, solidairement, réparation du préjudice, réparation, titre de dommages et intérêts, dommages, titre de dommages, dommages et intérêts, titre de dommages-intérêts, payer aux époux, jouissance |
| *styx* | 700 du code de procédure, article 700 du code, 700 du code, article 700, 700 |

Les termes candidats sont des n-grammes de taille variant d'1 à 5 mots consécutifs

Tableau 2.13 – Premiers termes sélectionnés lors de la première itération de la validation croisée

Troisièmement, les expérimentations ont été réalisées sur des décisions d'appel mais les énoncés, de demande et résultat renvoyant aux décisions de jugements antérieurs, ne sont pas encore traités. Ces références aux décisions antérieures représentent une part importante des demandes des décisions d'appel. Il est donc nécessaire de les intégrer explicitement dans le processus d'extraction, pour compléter les données extraites.

## 2.5 Conclusion

Ce chapitre décrit le problème d'extraction de données pertinentes relatives aux paires demande-résultat mentionnées dans les décisions de justice. Les divers défis relatifs à la tâche y sont discutés en remarquant des analogies avec d'autres tâches classiques de la fouille de données textuelles. Il a été démontré la solvabilité du problème par la proposition et l'expérimentation d'une approche d'extraction basée sur la terminologie de la catégorie des demandes à extraire et autres connaissances du do-

maine judiciaire telles que les motifs d'énoncés de demandes et de résultat, ainsi que leur position conventionnelle dans les documents. Les expérimentations démontrent que l'approche permet d'extraire plus ou moins bien des demandes selon la catégorie traitée. Même si nos résultats ont été obtenus à partir de terminologies apprises, une liste de termes fournis par les experts pourrait être plus précise et mettrait à l'abris des biais liés aux échantillons d'apprentissage. A cause de la forte dépendance aux subtilités de rédaction des décisions judiciaires, la méthode rencontre des limites qui ne peuvent être surmontées qu'en rendant la méthode beaucoup plus complexe qu'elle ne l'est déjà. Des approches d'apprentissage automatique sont recommandées comme perspectives. Elles devront être capables d'apprendre l'emplacement des données à extraire de manière semi-supervisée à l'aide de faibles quantités de documents annotés de grande taille.

# Chapitre 3

# Identification du sens du résultat

## 3.1 Introduction

quelle est l'entrée ? la demande ? comment elle est modélisée ? préciser l'importance de la tâche pour le métier : analyse descriptive du sens du résultat.

Comme le précédent, ce chapitre est relatif à l'extraction de données sur les demandes et résultats correspondants. Cependant, il est question ici d'extraire uniquement le sens du résultat d'une demande connaissant sa catégorie. Cette étude est intéressante parce que le problème devient plus simple. En se passant de la localisation précise de l'énoncé du résultat, l'extraction du sens du résultat peut être formulée comme une tâche de classification de documents. Nous modélisons la tâche comme un problème de classification binaire consistant à entrainer un algorithme à reconnaitre si la demande a été rejetée (sens = rejette) ou acceptée (sens = accepte). Cette modélisation est proposée sur une restriction du problème définie par les postulats (observations) 3.1.1 et 3.1.2 suivants. commencer par le postulat : qui sont des observations ! en précisant que le choix de motivation est motivé par l'expertise du métier.

**Postulat 3.1.1** *Pour toute catégorie de demande C, les documents ne contenant qu'une demande de catégorie C sont majoritaires.*

Ce postulat est légitime car les statistiques sur les données labellisées de la Figure 2.3 montrent bien que dans chaque catégorie, les décisions contiennent en majorité une demande. On remarque néanmoins l'exception de la catégorie STYX (dommage-intérêt sur l'article 700 CPC), où dans la majorité des documents, on a plutôt 2 demandes. Cette exception peut

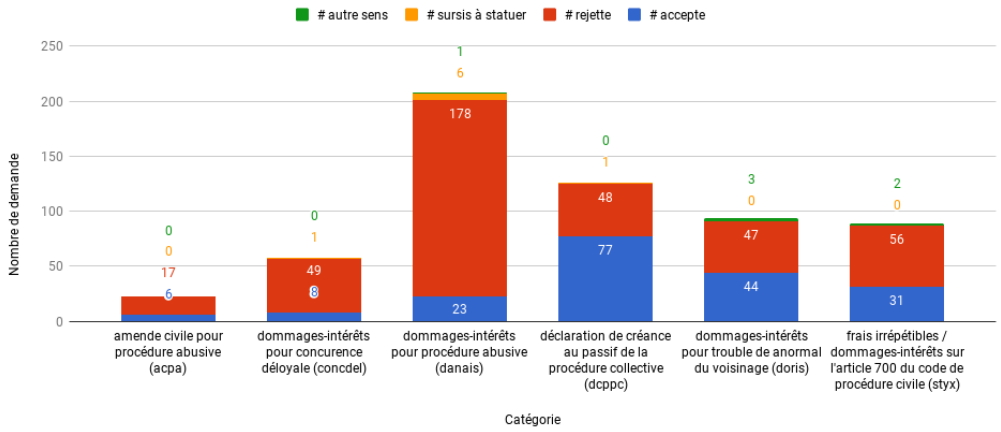



Figure 3.1 – Répartition des sens de résultat dans les données annotées.

se justifier par le fait que chaque partie fait généralement ce type de demande car elle porte sur le remboursement des frais de justice. Ce postulat présente cependant un inconvénient dû au fait que la majorité des demandes se trouvent dans des décisions à plus d'une demande. Il est donc possible de manquer un grand nombre de demandes.

**Postulat 3.1.2** *Le sens du résultat est généralement binaire : accepte ou rejette.*

Ce postulat est justifié car les sens de résultat ont majoritairement l'une de ces deux valeurs (Figure 3.1). Les autres sens sont très rares.

Cette étude porte sur l'analyse de l'impact de différents aspects techniques en général impliqués dans la classification de textes qui consistent en général en une combinaison de représentations des documents et d'algorithmes de classification. Cette analyse permettra de savoir s'il existe une certaine configuration permettant de déterminer le sens du résultat à une demande sans identifier précisément cette dernière dans le document.

## 3.2 Classification de documents

La classification de textes permet d'organiser des documents dans des groupes prédéfinis. Elle reçoit depuis longtemps beaucoup d'attentions. Deux choix techniques influencent principalement les performances : la

représentation des textes et l'algorithme de classification. Dans la suite, la variable à prédire est notée $y$, et la base d'apprentissage comprend les observations de l'échantillon $S = \{(x_i, y_i)_{i=1..n}\}$.

## 3.2.1 Algorithmes traditionnels de classification de données

Bien que la classification de documents voit se développer récemment des algorithmes propres aux textes, un grand nombre de méthodes ont été développées. Ces méthodes sont généralement basée sur une représentation des textes dans un espace vectoriel $\mathcal{X}$ d'entrée et délimitent une frontière entre les classes dans un espace multidimensionnel. Les notations du Tableau 2.3 sont utilisées dans cette section.

### 3.2.1.1 Le Bayésien naïf (NB)

Le classifieur naïf bayésien [Duda *et al.* , 1973] est un modèle à densité qui estime la probabilité qu'un texte appartienne à une classe à l'aide du théorème de Bayes [Raschka, 2014] :

$$\text{probabilité a posteriori} = \frac{\text{probabilité conditionnelle} \cdot \text{probabilité a priori}}{\text{évidence}} \tag{3.2.1}$$

La probabilité a posteriori peut être interprétée pour la classification de documents par la question "Quelle est la probabilité que le document $d$ soit de la classe $y = c \in C$?". La réponse à cette question se formalise comme suit :

$$\mathbb{P}(y = c|d) = \frac{\mathbb{P}(d|c)\mathbb{P}(c)}{\mathbb{P}(d)}, \forall c \in C$$

ou plus simplement $\mathbb{P}(y = c|d) = \mathbb{P}(c)\mathbb{P}(d|c)$ car $\mathbb{P}(d)$ ne change pas en fonction de la classe et peut donc être ignorée [Rish, 2001]. $d$ est catégorisé dans la classe $c$ pour laquelle $\mathbb{P}(c|d)$ est maximale :

$$y = \underset{c \in C}{\operatorname{argmax}}\, \mathbb{P}(c|d).$$

La phase d'entrainement, appliquée à des exemples déjà labellisés, permet d'estimer les paramètres $\mathbb{P}(c)$ et $\mathbb{P}(d|c)$ qui servent à calculer $\mathbb{P}(c|d)$.

$\mathbb{P}(c)$ est estimé par la proportion de documents classés dans $c$ parmi les exemples d'apprentissage : $\mathbb{P}(c_j) = \frac{N_c}{N}, \forall c \in C$.

$\mathbb{P}(c|d)$ est estimé grâce l'hypothèse 3.2.1 d'indépendance conditionnelle des descripteurs (termes). Une hypothèse naïve dont la violation, par les données réelles, n'empêche pas le NB de bien fonctionner [Rish, 2001].

**Hypothèse 3.2.1 (indépendance conditionnelle des descripteurs)** *[ Un modèle naïf bayésien étant de type génératif], la position de chaque mot dans le texte est générée indépendamment de tout autre mot étant connue la catégorie du texte.*

Si l'ensemble des termes de $d$ est $\{t_1, \ldots, t_K\}$, alors grâce à l'hypothèse 3.2.1, $\mathbb{P}(d|c) = \mathbb{P}(t_1, \ldots, t_K|c) = \prod_{k=1}^{K} \mathbb{P}(t_k|c)$, et pour un terme $t_k$, la probabilité conditionnelle $\mathbb{P}(t_k|c)$ est la proportion d'exemples de $c$ qui contiennent $t_k : \mathbb{P}(t_k|c) = \frac{N_{t_k,c}}{|D_c|}, \forall k \in \{1, \ldots, K\}$.

#### 3.2.1.2 Machine à vecteurs de support (SVM)

La classification binaire par une machine à vecteurs de support (SVM) [Vapnik, 1995] affecte à tout objet en entrée $x$ la classe $y$ qui correspond au coté d'un hyperplan, séparant les exemples d'entraînement des classes candidates, où $x$ se trouve. La phase d'apprentissage consiste à déterminer l'hyperplan optimal $w^T x + b = 0$ i.e. dont la marge [1] est maximale (Figure 3.2 [2]).

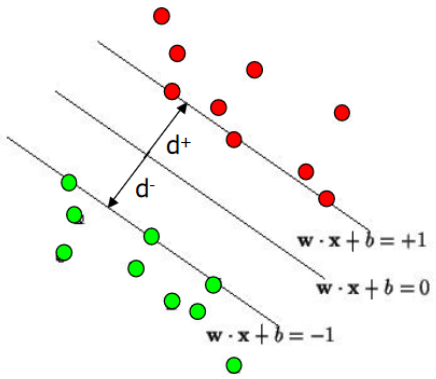



Figure 3.2 – Hyperplan optimal et marge maximale d'un SVM.

1. La plus petite distance entre les échantillons d'apprentissage et l'hyperplan séparateur
2. `http://web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf`

Le vecteur $w$ des poids des caractéristiques et le biais $b$ sont déterminés par le problème d'optimisation du « SVM à marges molles » de Cortes & *Vap*nik [1995] :

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.c.} \quad & y_i(w^T + b) \geq 1 - \xi_i, \xi_i \geq 0. \end{aligned}$$

où $C$ est la constante de régularisation pour éviter un sur-apprentissage ou le sous-apprentissage, les $\xi_i$ sont des variables ressort (*slack variables*) qui permettent à des points de se retrouver dans la marge.

La classification par SVM ne s'applique que lorsque les exemples d'apprentissage des classes sont linéairement séparables. Cependant, ils ne le sont pas toujours dans l'espace $\mathcal{X}$. Ainsi, une fonction « noyau » (*kernel*) $K : \mathcal{X} \longrightarrow \mathcal{F}$ doit être choisie pour transformer chaque donnée entrée $x$ de l'espace original $\mathcal{X}$ vers un nouvel espace $\mathcal{F}$ dit de caractéristiques dans lequel les classes sont linéairement séparables. Par conséquent, la fonction de classification s'écrit

$$f(x) = \sum_{i=1}^{n}\alpha_i K(x, x_i) + b$$

où les $\alpha_i$ sont les coefficients de la combinaison linéaire des exemples d'apprentissage égale à $w$ ($w = \sum_{i=1}^{n}\alpha_i x_i$) [Ben-Hur & Weston, 2010]. Parmi les multiples formes qu'il peut prendre, le noyau peut être, par exemple, soit linéaire($K(x, x_i) = x^T x_i + c$), soit polynomial ($K(x, x_i) = (\gamma x^T x_i + c)^d$), soit Gaussien ou RBF[3] ($K(x, x_i) = \exp -\gamma\|x - x_i\|$), soit une sigmoïde ($K(x, x_i) = \tanh(\gamma x^T x_i + c)$) [Amami *et al.* , 2015].

#### 3.2.1.3 k-plus-proches-voisins (kNN)

L'algorithme des $k$-plus-proches-voisins est un algorithme très simple qui consiste à affecter à un nouvel objet la classe majoritaire $y'$ parmi ceux des $k$ points d'exemples d'entrainement $\{(x_i, y_i)\}_{1:k}$, les plus proches du point $x'$ de cet objet selon la métrique $d$ choisie. Ainsi, trois éléments clés influencent l'efficacité de la classification :

3. *radial base function*

1. les données d'entrainement dont le nombre s'il est très grand peut rendre cher le processus de classification, car la distance du nouvel objet à chaque point annoté, est calculée ;
2. le nombre de voisins (c'est-à-dire la valeur de $k$) qui ne doit être ni très petit (sensibilité aux bruits / *outliers*), ni très grand (risque d'avoir dans le voisinage beaucoup de points d'une autre classe). La sensibilité au nombre de voisins peut être atténuée en pondérant les points par leur distance à l'objet à classer. Il a été proposé plusieurs variantes de cette stratégie de « vote pondéré par la distance », comme par exemple :
   - $y' = \underset{c}{\text{argmax}} \sum_{(x_i,y_i)} \frac{1}{Dis(x,x_i)^2} \times I(c = y_i)$ [Dudani, 1976]

     $$\text{où } I(c = y_i) = \begin{cases} 1 & \text{si } c = y_i \\ 0 & \text{sinon} \end{cases}$$

   - $y' = \underset{c}{\text{argmax}} \sum_{(x_i,y_i)} w_i \times I(c = y_i)$ [Gou *et al.* , 2011]

     $$\text{avec } w_i = \begin{cases} \frac{Dis(x,d_k) - Dis(x,d_i)}{Dis(x,d_k) - Dis(x,d_1} & \text{si } Dis(x,d_k) \neq Dis(x,d_1) \\ 1 & \text{sinon} \end{cases}$$

3. la métrique de calcul de distance qui doit être adéquate pour le type de donnée et la tâche, comme par exemple, la distance cosinus qui est préférable à la distance euclidienne pour la classification de documents, la deuxième métrique se dégradant lorsque le nombre d'attributs augmente.

#### 3.2.1.4 Arbre de décision

Un arbre de décision est une structure arborescente utilisée en fouille de données pour associer un label prédéfini à des objets (classification), ou prédire la valeur d'une variable continue (régression). Il comprend des nœuds internes qui correspondent chacun à un test sur la valeur d'un attribut (test uni-varié), des arêtes correspondant à une sortie du test, et enfin des feuilles ou nœuds terminaux qui correspondent chacun à une prédiction. La classification d'un objet $x$ consiste à faire passer successivement les tests en fonction des valeurs des attributs de $x$, de la racine jusqu'à une feuille dont le label est retourné comme classe de $x$ (Algorithme 3).

**Algorithme 3 :** Classification par arbre de décision

**Données :** Objet $x$, Arbre $A$
**Résultat :** label
$n := racine(A)$ ;
**tant que** *n n'est pas une feuille* **faire**
    Effectuer sur $x$ le test associé à $n$;
    $n :=$ noeud fils de $n$ correspondant au résultat du test ;
**retourner** le label associé à la feuille $n$;

La construction de l'arbre (phase d'apprentissage) consiste à générer une hiérarchie de tests, aussi courte que possible, qui divise successivement l'ensemble $S$ d'exemples d'apprentissage en sous-ensembles disjoints de plus en plus pures [4]. L'arbre est construit de la racine aux feuilles en divisant les données d'entraînement $S_t$ à chaque nœud $(t)$ de sorte à minimiser le degré d'impureté des sous-ensembles d'exemples $S_{t_i}$ dans les nœuds fils $(t_i)$. Le critère de coupe est généralement défini à partir d'une métrique d'impureté comme par exemple :

- l'entropie de la distribution des classes dans $S_t$ :
  $h_C(S_t) = -\sum_{c \in C} [p(c|S_t) \log_2 p(c|S_t)]$ ;
- l'indice de Gini mesurant la divergence entre les distributions de probabilité des valeurs de la variable prédite : $g_C(S_t) = 1 - \sum_{c \in C} [p(c|S_t)]^2$ ;
- l'erreur de classification définie par : $e_C(S_t) = 1 - \max_{c \in C} [p(c|S_t)]$.

Pour ces métriques, $p(c|S_t)$ représente la proportion d'exemples du nœud $t$ appartenant à $c$. Parmi les critères de séparation les plus populaires associés à ces métriques d'impureté, on retrouve :

- le gain d'information apporté par le test $t$ portant sur l'attribut $a$ (qui divise $S_t$ en des sous-ensembles $S_{t_i}$) utilisant l'entropie comme métrique d'impureté, et est définie par la différence entre l'entropie de $t$ et l'entropie moyenne des fils de $t$ :
  $ig(S_t, a) = h_C(S_t) - i(S_t, t, a) = h_C(S_t) - \sum_{S_{t_i}} \frac{|S_{t_i}|}{|S_t|} \cdot h_C(S_{t_i})$;

4. homogénéité des labels

- le rapport des gains, qui corrige le gain d'information, biaisé en faveur des tests ayant un grand nombre d'alternatives (sorties du nœud), en prenant en compte l'information intrinsèque $h_t(S_t)$ de la séparation de $S_t$ suivant le test $t$ en sous-ensembles $S_{t_i}$ :
  $gr(S_t, t, a) = \frac{ig(S_t,t,a)}{h_t(S_t)}$ avec $h_t(S_t) = \sum_i \frac{|S_{t_i}|}{|S_t|} \log_2 \left( \frac{|S_{t_i}|}{|S_t|} \right)$
- le critère binaire de "doublage" (*twoing criteria*) qui ne s'emploie que pour les arbres binaires :
  $tc(t) = \frac{P(S_{t_R}|S_t)P(S_{t_L}|S_t)}{4} \left[ \sum_{c \in C} |p(c|t_L) - p(c|t_R)| \right]^2$ où $P(S_{t_R}|S_t)$ et $P(S_{t_L}|S_t)$ sont les proportions de $S_t$ qui vont respectivement dans les fils $t_R$ et $t_L$ après séparation suivant le test $t$.

Les variables nominales peuvent être divisées soit en utilisant autant de partitions que de valeurs distinctes, soit uniquement en des partitions binaires suivant des tests booléens nécessitant de rechercher la division optimale. Les variables numériques sont divisées quant à elles soit par discrétisation de leur domaine en les transformant en variables catégoriques ordinales, soit en recherchant la meilleure division binaire parmi toutes les séparations possibles.

La construction de l'arbre est une division récursive qui peut continuer tant qu'il est possible d'améliorer la pureté des nœuds, ce qui peut engendrer un arbre très grand résultant en un sur-apprentissage [5], et une forte complexité temporelle et spatiale lors de la prédiction. Pour s'arrêter plus tôt ("pré-élagage"), plusieurs conditions sont possibles comme par exemple, l'atteinte par la taille des données ($|S_t|$) d'un seuil minimum, ou l'atteinte par l'arbre d'une profondeur maximale, ou l'amélioration du critère de division est très faible, etc. Le post-élagage [6] est appliqué après construction de l'arbre toujours dans le but de minimiser le sur-apprentissage et la complexité. Le post-élagage peut consister par exemple, soit à simplement et rapidement éliminer successivement les feuilles si cela ne fait pas croître le taux d'erreur sur $S$ (« élagage basé sur la réduction du taux d'erreur »), et remplacer chaque nœud par sa classe majoritaire, soit à éliminer successivement les sous-arbres qui minimisent $\frac{erreur(elagage(A,A'),S) - erreur(A,S)}{\|feuille(A)\| - \|feuille(elagage(A,A'))\|}$ (« stratégie coût-complexité »).

5. Un modèle trop précis a un très faible taux d'erreur sur les données d'entraînement (erreur d'apprentissage) mais un fort taux d'erreur sur les données de test (erreur de test).

6. Suppression de sous-arbres superflus ou en trop après génération de l'arbre.

Les algorithmes de construction d'arbres diffèrent ainsi par leur critère de séparation, leur stratégie d'élagage, et leur capacité à gérer les types d'attributs, les valeurs manquantes et extrêmes. Singh & Gupta [2014] comparent les deux algorithmes CART [Breiman *et al.* , 1984] (critère de « doublabe », élagage coût-complexité) et C4.5 [Quinlan, 1993] (rapport des gains, élagage à réduction d'erreur).

#### 3.2.1.5 Analyses discriminantes linéaires et quadratiques

Pas de citation ? L'analyse discriminante comprend l'ensemble des méthodes déterminant les combinaisons linéaires de variables qui permettent de séparer le mieux possible $K$ catégories ou variables qualitatives. Les analyses linéaires et quadratiques sont des méthodes probabilistes basées sur la probabilité conditionnelle d'appartenance d'un objet $X$ à une classe $y_k$ :

$$P(Y = y_k|X) = \frac{P(Y = y_k)P(X|Y = y_k)}{P(X)} = \frac{P(Y = y_k)P(X|Y = y_k)}{\sum_{j=1}^{K} P(Y = y_j)P(X|Y = y_j)}.$$

La classe de $X$ est donc $y_{k*} = \underset{k}{\operatorname{argmax}}\, P(Y = y_k|X) = P(Y = y_k)P(X|Y = y_k)$ car le dénominateur est le même pour toutes les classes. Dans cette expression, $P(Y = y_k)$ est la proportion d'exemples de classes $y_k$ dans l'ensemble des données d'apprentissage. Il ne reste donc qu'à déterminer $P(X|Y = y_k)$, pour trouver $y$. Deux hypothèses simplifient les calculs :

1. l'hypothèse de normalité statuant que la probabilité conditionnelle $P(X|Y)$ suit une loi normale multivariée :

$$P(X|Y = y_k) = \frac{1}{\sqrt{2\pi det(V_k)}} e^{-\frac{1}{2}(X-\mu_k)' V_k^{-1}(X-\mu_k)}$$

   $\mu_k$ étant le centre de gravité conditionnel, et $V_k$ la matrice de variance covariance conditionnelle ;

2. l'hypothèse d'homoscédasticité statuant que les matrices de variance co-variance conditionnelles sont identiques i.e. :

$$\forall j, k \in \{1, ..., K\}, V_j = V_k = V.$$

L'analyse discriminante linéaire (LDA) est définie par une simplification de $P(X|y_k)$ sous ces deux hypothèses. En effet, grâce à la proportionnalité de la probabilité conditionnelle à :$\ln[P(X|y_k)] \propto -\frac{1}{2}(X-\mu_k)'V^{-1}(X-\mu_k)$(Vérifier la position de la transposée), on déduit une fonction discriminante (ou de classement) linéaire proportionnelle à $P(y_k|X)$ :

$$d(y_k, X) = \ln[P(Y = y_k)] + \mu_k V^{-1} X' - \frac{1}{2}\mu_k V^{-1}\mu_k'.$$

Ainsi $y_{k*} = \underset{k\in\{1,\dots,K\}}{\operatorname{argmax}}\, d(y_k, X)$.

L'analyse discriminante quadratique (QDA) considère l'hétéroscédasticité (i.e. $\exists k \neq j, V_k \neq V_j$), et donc ne s'appuie que sur la 1e hypothèse (multinormalité). Dans ce cas, on obtient une règle quadratique de classification $k* = \underset{k\in\{1,\dots,K\}}{\operatorname{argmax}}\, Q_k(X)$ où :

$$Q_k(X) = (X-\mu_k)'V_k^{-1}(X-\mu_k) - 2\ln(\pi_k) + \ln(det(V_k))$$

est la fonction quadratique de classement de la classe $k$.

## 3.2.2 Algorithmes dédiés aux textes

Les algorithmes dédiés aux textes intègrent leur propre représentation de document, contrairement aux algorithmes opérant dans des espaces vectoriels aux axes et poids paramétrables à volonté comme le SVM. Actuellement, les algorithmes NBSVM [Wang & Manning, 2012] et FastText [Grave *et al.* , 2017] sont les plus populaires pour la classification de documents avec une très bonne précision pour l'analyse de sentiments [citation].

### 3.2.2.1 NBSVM

Le NBSVM [Wang & Manning, 2012] est un classifieur binaire (deux labels $\{-1;1\}$) dont le principe consiste à transformer les poids $f^{(k)}$ caractéristiques $V$ des textes $x^{(k)}$, réduits à leur simple présence $\widehat{f}^{(k)}$ en réalisant leur produit élément à élément ($\overset{\sim}{f}^{(k)} = r \circ \widehat{f}^{(k)}$) avec le vecteur de poids $r$ du classifieurs bayésien multinomial (calculé avec le vecteur

présence de caractéristique) : $r = \log\left(\frac{p/||p||_1}{q/||q||_1}\right)$ avec $p = \alpha + \sum_{k:y^{(k)}=1} f^{(k)}$, $q = \alpha + \sum_{k:y^{(k)}=-1} f^{(k)}$. L'ensemble des caractéristiques $V$ est constitué de n-grammes de mots. Le nouveau vecteur issu de ce produit représente le texte ($x^{(k)} = \overset{\sim}{f}^{(k)}$) en entrée d'un SVM classique. La classe de $x^{(k)}$ est prédite par : $y^{(k)} = sign(\mathbf{w}^T x^{(k)} + b)$, $\mathbf{w}$ et $b$ étant appris lors de l'entraînement du SVM. Une interpolation entre le bayésien multinomial et le SVM est nécessaire pour assurer la robustesse du NBSVM et des performances excellentes pour toute tâche de classification de documents ; les poids $\mathbf{w}$ sont réajustés par le modèle $\mathbf{w}' = (1-\beta)\overline{w} + \beta\mathbf{w}$, où $\overline{w} = ||\mathbf{w}||_1/|V|$ et $\beta \in [0;1]$.

#### 3.2.2.2 FastText

FastText [Grave *et al.* , 2017], quant à lui, est un modèle de réseau de neurones dont l'architecture est semblable à celle de la variante CBOW de la méthode de plongement sémantique Word2Vec [Mikolov *et al.* , 2013] dans laquelle le mot du milieu a été remplacé par le label de la classe du texte et au-dessus de laquelle la fonction softmax $f(z) = \left[\frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}\right]_{\forall j \in \{1,\ldots,K\}}$ est rajoutée pour réaliser la classification à partir de la représentation distribuée du texte. L'entraînement, sur un corpus manuellement annoté de $n$ documents, consiste à minimiser $-\frac{1}{n} y_i \cdot \sum_{i=1}^{n} y_i \cdot \log f(B \cdot A \cdot x_i)$ qui estime la distribution de probabilité des classes ; $A$ et $B$ étant les matrices des poids à apprendre, et $x_i$ la représentation vectorielle sous forme de « sac de $N$-grammes »[7] du $i^{\text{eme}}$ document. Comment se déroule l'entrainement et l'apprentissage ?

### 3.2.3 Techniques d'amélioration de l'efficacité

La faible quantité [Ruparel *et al.* , 2013] et le déséquilibre des données sont susceptibles d'être des obstacles à l'entraînement des modèles de clas-

7. Modèle sac-de-mot calculé sur des occurrences de $N$-grammes (segments $N$ mots consécutifs dans un texte.)

sification. De nombreuses techniques permettent néanmoins d'optimiser l'apprentissage en fonction des données. La sélection de modèle consiste à choisir les meilleures valeurs des hyper-paramètres (par exemple $C$ et $\gamma$ chez le SVM) en testant différentes combinaisons de valeurs candidates sur une fraction de la base d'entraînement (base de développement). La combinaison de classifieurs est aussi une méthode très étudiée [Kittler *et al.* , 1996; Kuncheva, 2004; Tulyakov *et al.* , 2008] notamment par l'exemple des forêts aléatoires [Breiman, 2001], ou de SVM ensembliste (*Ensemble SVM*) [Dong & Han, 2005]. Par ailleurs, la représentation vectorielle des textes résulte généralement en des vecteurs de haute dimension dont les coordonnées sont en majorité nulles. Par conséquent, les techniques de réduction ou transformation des dimensions, comme les analyses discriminantes, permettent de d'obtenir des vecteurs plus pertinents pour la classification.

## 3.3 Application du PLS à la classification des textes

Justification : Pourquoi le PLS ? à quel problème cette méthode répond spécifiquement ? Qu'est ce que moi j'ai fait/ajouté ? : le modèle a été déjà publié en temps que méthode de régression. nous l'adaptons pour de la classification de textes pourquoi ? : ça n'a jamais été appliqué en classif. de texte, orienté par l'expertise de l'encadrement sur cette technique déjà publiée.

La méthode ou l'analyse des moindres carrés partiels PLS [Wold, 1966] explique la dépendance entre une ou plusieurs variables $y$ (dite dépendantes) et des $K$ variables $x_1, x_2, ..., x_K$ (dites explicatives ou indépendantes). Elle consiste principalement à transformer les variables explicatives en un nombre réduit de $h$ composantes principales orthogonales $t_1, ..., t_h$. Il s'agit donc d'une méthode de réduction de dimension au même titre que l'analyse en composantes principales, l'analyse discriminante linéaire (LDA), et l'analyse discriminante quadratique (QDA). Les composantes $t_h$ sont construites par étapes en appliquant l'algorithme du PLS de façon récurrente sur les données mal prédites (résidus). Plus précisément, à chaque itération $h$, la composante $t_h$ est calculée par la formule $t_h = w_{h1}x_1 + \cdots + w_{hj}x_j + \cdots + w_{hp}x_K$ dont les coefficients $w_{hj}$ sont à estimer. L'analyse PLS présente plusieurs avantages [Lacroux, 2011] dont la robustesse

au problème de haute-dimension [8] comme on peut l'observer dans nos données (faible quantité de données annotées), et aussi prend en compte la multicolinéarité qui peut exister entre les variables explicatives, notamment quand celles-ci sont des mots/termes co-occurrents dans les documents. Cette méthode est étendue et appliquée avec succès pour divers problèmes de régression [Lacroux, 2011] ou de classification de données vectorielles en général [Liu & Rayens, 2007; Durif *et al.* , 2017; Bazzoli & Lambert-Lacroix, 2018], et de textes en particulier [Zeng *et al.* , 2007]. Nous nous sommes intéressés particulièrement à deux extensions : la régression Gini-PLS [Mussard & Souissi-Benrejab, 2018] dont l'intérêt est de réduire la sensibilité aux valeurs aberrantes des variables, et la méthode Logit-PLS [Tenenhaus, 2005] combinant la régression logistique et la PLS. Une combinaison de ces deux approches (Logit-Gini-PLS) est décrite dans la suite de cette section (Section § 3.3.3.2).

### 3.3.1 L'opérateur Gini covariance

Soit $\bar{x}_k$ la moyenne arithmétique de la variable $x_k, \forall k \in [1 \,..\, m]$ sur les $n$ observations d'apprentissage. L'opérateur de Gini covariance proposé par Schechtman & Yitzhaki [2003], encore appelé opérateur co-Gini est donné par :

$$\text{cog}(x_\ell, x_k) := \text{cov}(x_\ell, F(x_k)) = \frac{1}{n}\sum_{i=1}^{n}(x_{i\ell} - \bar{x_\ell})(F(x_{ik}) - \bar{F}_{x_k}), \qquad (3.3.1)$$

où $F(x_k)$ est la fonction de répartition de $x_k$, $\bar{F}_{x_k}$ sa moyenne, avec $\ell \neq k = 1, \ldots, K$. Lorsque $k = \ell$ le co-Gini mesure la variabilité entre une variable et elle-même (l'équivalent de la variance mesurée sur la norme $\ell_2$). Le co-Gini est une mesure basée sur la distance de Manhattan (distance de métrique $\ell_1$), en effet :

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}|x_{ik} - x_{jk}| = 4\text{cog}(x_k, x_k).$$

8. Lorsque le nombre de variables explicatives est très grand devant le nombre d'observations ($n << K$).

D'autre part, lorsque $k \neq \ell$, le co-Gini produit une mesure de la variabilité jointe entre deux variables. Puisque le co-Gini n'est pas symétrique :

$$\mathrm{cog}(x_k, x_\ell) := \mathrm{cov}(x_k, F(x_\ell)) = \frac{1}{n}\sum_{i=1}^{n}(x_{ik} - \bar{x_k})(F(x_{i\ell}) - \bar{F}_{x_\ell}).$$

Définissons les rangs croissants d'une variable alétoire afin de fournir un estimateur de $F$,

$$R_\uparrow(x_{i\ell}) := nF(x_{i\ell}) = \begin{cases} \#\{x \leq x_{i\ell}\} & \text{si aucune observation similaire} \\ \frac{\sum_{i=1}^{p}\#\{x \leq x_{i\ell}\}}{p} & \text{s'il existe } p \text{ valeurs similaires } x_{i\ell}. \end{cases}$$

Alors, un estimateur du co-Gini est donné par,

$$\widehat{\mathrm{cog}}(x_\ell, x_k) := \frac{1}{n}\sum_{i=1}^{n}(x_{i\ell} - \bar{x_\ell})(R_\uparrow(x_{ik}) - \bar{R}_{\uparrow_{x_k}}),\ \forall k, \ell = 1, \ldots, K, \quad (3.3.2)$$

avec $\bar{R}_{\uparrow_{x_k}}$ la moyenne arithmétique du vecteur rang de la variable $x_k$.

### 3.3.2 Gini-PLS

Le premier algorithme Gini-PLS a été proposé par Mussard & Souissi-Benrejab [2018]. Nous le décrivons dans les lignes qui suivent. Il s'agit d'une méthode de compression avec débruitage qui consiste à réduire les dimensions de l'espace généré par $X$ afin de trouver des composantes principales débruitées, dans le même esprit qu'une ACP débruitée, néanmoins l'approche est supervisée dans la mesure où une variable cible $y$ est prise en compte dans le changement d'espace. Le sous-espace formé par les composantes principales $\{t_1, t_2, \cdots\}$ est construit de telle sorte que le lien entre les variables explicatives $X = [x_1, x_2, \cdots]$ et la cible $y$ est maximisé.

• **Étape 1 :** La régression Gini permet de concevoir un nouveau type de lien entre la variable expliquée et les variables explicatives tout en évitant l'influence des valeurs aberrantes. Ceci est permis grâce notamment à l'opérateur co-Gini dans lequel le rôle de la variable explicative est remplacé par celui de son vecteur rang dans un espace muni d'une métrique $\ell_1$. Ainsi, il est possible de créer un nouveau vecteur de poids $w_1$ qui renforce le lien (co-Gini) entre la variable expliquée $y$ et les régresseurs $X$ dans

le cadre d'une régression (linéaire ou non linéaire).
La solution du programme,

$$\max \operatorname{cog}(y, Xw_1) \text{ , s.c. } \|w_1\| = 1 \text{ , est}$$

$$w_{1j} = \frac{\operatorname{cog}(y, x_j)}{\sqrt{\sum_{k=1}^{K} cog^2(y, x_k)}} \text{ , } \forall j = 1 \ldots, p \text{ .}$$

La pondération est équivalente à :

$$w_{1k} = \frac{\operatorname{cov}(y, R(x_k))}{\sqrt{\sum_{k=1}^{K} \operatorname{cov}^2(y, R(x_k))}} \text{ , } \forall k = 1 \ldots, K \text{ .}$$

Comme dans la régression PLS, on régresse $y$ sur la composante $t_1$ qui est construite de la manière suivante :

$$t_1 = \sum_{k=1}^{K} w_{1k} x_k \implies y = \hat{c}_1 t_1 + \hat{\varepsilon}_1 \text{ .}$$

• **Étape 2 :** On régresse le vecteur rang de chaque régreseur $R(x_k)$ sur la composante $t_1$ par moindres carrés ordinaires afin de récupérer les résidus $\hat{U}_{(1)k}$ :

$$R(x_k) = \hat{\beta} t_1 + \hat{U}_{(1)k} \text{ , } \forall k = 1, \cdots, K \text{ .}$$

On construit le nouveau vecteur de pondération en utilisant les rangs des résidus des régressions partielles :

$$\max \operatorname{cog}(\hat{\varepsilon}_1, \hat{U}_{(1)} w_2) \text{ , s.c. } \|w_2\| = 1 \implies w_{2k} = \frac{\operatorname{cog}(\hat{\varepsilon}_1, \hat{U}_{(1)k})}{\sqrt{\sum_{k=1}^{K} \operatorname{cog}^2(\hat{\varepsilon}_1, \hat{U}_{(1)k})}}.$$

On utilise à présent les composantes $t_1$ et $t_2$ pour établir un lien entre $y$ et les régresseurs $x_k$ :

$$t_2 = \sum_{k=1}^{K} w_{2k} \hat{U}_{(1)k} \implies y = \hat{c}_1 t_1 + \hat{c}_2 t_2 + \hat{\varepsilon}_2 \text{ .}$$

La validation croisée permet de savoir si $t_2$ est significative.

• **Étape $h$ :** Les régressions partielles sont réitérées en ajoutant l'influence de $t_{h-1}$ :

$$R(x_k) = \beta t_1 + \cdots + \gamma t_{h-1} + \hat{U}_{(h-1)k}\ ,\ \forall k = 1, \cdots, K.$$

D'où, après maximisation :

$$w_{hk} = \frac{\cos(\hat{\varepsilon}_{h-1}, \hat{U}_{(h-1)k})}{\sqrt{\sum_{k=1}^{K} \cos^2(\hat{\varepsilon}_{h-1}, \hat{U}_{(h-1)k})}}\ ,$$

$$t_h = \sum_{k=1}^{K} w_{hk} \cdot \hat{U}_{(h-1)k} \implies y = \alpha_2 + c_1 t_1 + \cdots + c_h t_h + \varepsilon_h\ .$$

La procédure s'arrête lorsque la validation croisée indique que la composante $t_h$ n'est pas significative. L'algorithme Gini-PLS1 est valable si toutes les composantes $t_h$ et $t_l$ sont orthogonales, $\forall h \neq l$.

La validation croisée permet de trouver le nombre optimal $h > 1$ de composantes à retenir. Pour tester une composante $t_h$, on calcule la prédiction du modèle avec $h$ composantes comprenant l'observation $i$, $\hat{y}_{h_i}$, puis sans l'observation $i$, $\hat{y}_{h_{(-i)}}$. L'opération est répétée pour tout $i$ variant de 1 à $n$ : on enlève à chaque fois l'observation $i$ et on ré-estime le modèle.[9] Pour mesurer la robustesse du modèle, on mesure l'écart entre la variable prédite et la variable observée :

$$PRESS_h = \sum_{i=1}^{n} \left(y_i - \hat{y}_{h_{(-i)}}\right)^2\ .$$

La somme des carrés résiduels obtenue avec le modèle à $(h-1)$ composantes est :

$$RSS_{h-1} = \sum_{i=1}^{n} \left(y_i - \hat{y}_{(h-1)_i}\right)^2\ .$$

Le critère $RSS_h$ (Residual Sum of Squares) du modèle à $h$ composantes et $PRESS_h$ (PRedicted Error Sum of Squares) sont comparés. Leur rapport

9. Les observations peuvent être éliminées par blocs *Cf.* Tenenhaus (1998), p. 77.

permet de savoir si le modèle avec la composante $t_h$ améliore la prédictibilité du modèle :

$$Q_h^2 = 1 - \frac{PRESS_h}{RSS_{h-1}} \ .$$

La composante $t_h$ est retenue si : $\sqrt{PRESS_h} \leq 0,95\sqrt{RSS_h}$. Autrement dit, lorsque $Q_h^2 \geq 0,0975 = (1 - 0,95^2)$, la nouvelle composante $t_h$ est significative, elle améliore la prévision de la variable $y$. Pour la significativité de la composante $t_1$, on utilise :

$$RSS_0 = \sum_{i=1}^{n} (y_i - \bar{y})^2 \ .$$

### 3.3.3 Régressions Gini-PLS généralisée

Schechtman & Yitzhaki [2003] ont récemment généralisé l'opérateur co-Gini afin d'imposer plus ou moins de poids en queue de distribution. Notons $r_k = (R_\downarrow(x_{1k}), \ldots, R_\downarrow(x_{nk}))$ le vecteur rang décroissant de la variable $x_k$, autrement dit, le vecteur qui assigne le rang le plus petit (1) à l'observation dont la valeur est la plus importante $x_{ik}$ :

$$R_\downarrow(x_{ik}) := \begin{cases} n+1-\#\{x \leq x_{ik}\} & \text{pas d'observation similaire} \\ n+1-\frac{\sum_{i=1}^{p}\#\{x \leq x_{ik}\}}{p} & \text{si } p \text{ observations similaires } x_{ik}. \end{cases}$$

L'opérateur co-Gini est généralisé grâce au paramètre $\nu$ :

$$\text{cog}_\nu(x_\ell, x_k) := -\nu\text{cov}(x_\ell, r_k^{\nu-1});\ \nu > 1. \tag{3.3.3}$$

Afin de bien comprendre le rôle de l'opérateur co-Gini, revenons sur la mesure du coefficient de corrélation linéaire généralisé au sens de Gini :

$$GC_\nu(x_\ell, x_k) := \frac{-\nu\text{cov}(x_\ell, r_k^{\nu-1})}{-\nu\text{cov}(x_\ell, r_\ell^{\nu-1})}\ ;\ GC_\nu(x_k, x_\ell) := \frac{-\nu\text{cov}(x_k, r_\ell^{\nu-1})}{-\nu\text{cov}(x_k, r_k^{\nu-1})}.$$

**Property 1 – Schechtman & Yitzhaki [2003] :**
(i) $GC_\nu(x_\ell, x_k) \leq 1$.
(ii) *Si les variables $x_\ell$ et $x_k$ sont indépendantes, pour tout $k \neq \ell$, alors* $GC_\nu(x_\ell, x_k) = GC_\nu(x_k, x_\ell) = 0$.

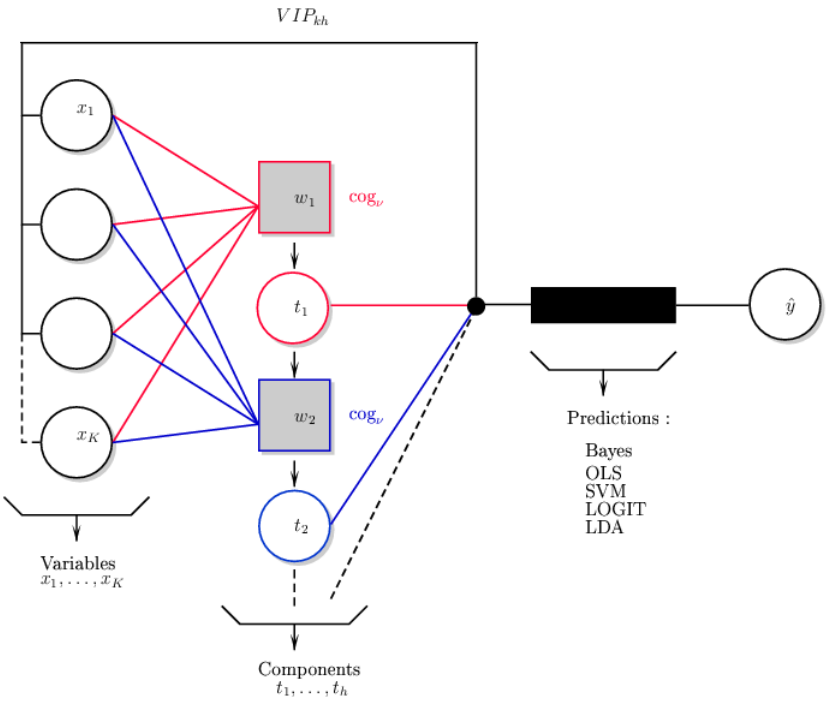
(iii) *Une transformation monotone des données $\varphi$ n'affecte pas le coefficient de corrélation, $GC_\nu(x_\ell, \varphi(x_k)) = GC_\nu(x_\ell, x_k)$.*
(iv) *Pour une transformation linéaire $\varphi$, $GC_\nu(\varphi(x_\ell), x_k) = GC_\nu(x_\ell, x_k)$ [comme le coefficient de corrélation de Pearson].*
(v) *Si $x_k$ et $x_\ell$ sont deux variables échangeables à une transformation linéaire près, alors $GC_\nu(x_\ell, x_k) = GC_\nu(x_k, x_\ell)$.*

Le rôle de l'opérateur co-Gini peut être expliqué de la manière suivante. Lorsque $\nu \to 1$, la variabilité des variables est atténuée de telle sorte que $\mathrm{cog}_\nu(x_k, x_\ell)$ tend vers zéro (même si les variables $x_k$ et $x_\ell$ sont fortement corrélées). Au contraire, si $\nu \to \infty$ alors $\mathrm{cog}_\nu(x_k, x_\ell)$ permet de se focaliser sur les queues de distribution $x_\ell$. Comme le montrent Olkin & Yitzhaki [1992], l'emploi de l'opérateur co-Gini atténue la présence d'outliers, du fait que le vecteur rang agit comme un instrument dans la régression de $y$ sur $X$ (régression par variables instrumentales).

Ainsi, en proposant une régression Gini-PLS basée sur le paramètre $\nu$, nous pouvons calibrer la puissance du débruitage grâce à l'opérateur co-Gini qui va localiser le bruit dans la distribution. Cette régression Gini-PLS généralisée devient une régression Gini-PLS régularisée où le paramètre $\nu$ joue le rôle de paramètre de régularisation.

#### 3.3.3.1 L'algorithme Gini-PLS généralisé

Dans ce qui suit nous généralisons la régression Gini-PLS de Mussard & Souissi-Benrejab [2018] avec renforcement du pouvoir de débruitage par l'intermédiaire du paramètre $\nu$.




La première étape consiste à trouver des poids de débruitage associés à chaque variable $x_k$ afin d'en déduire la première composante $t_1$ (ou première variable latente). Cette opération est bouclée jusqu'à la composante $t_{h^*}$, où $h^*$ est le nombre optimal de variable latentes. Ainsi, le modèle est estimé :

$$y = \sum_{h=1}^{h^*} c_h t_h + \varepsilon_h. \tag{3.3.4}$$

La statistique $VIP_{hj}$ est mesurée afin de sélectionner la variable $x_j$ qui a l'impact significatif le plus important sur $y$ estimé. Les variables les plus significatives sont celles dont $VIP_{hj} > 1$ avec :

$$VIP_{hj} := \sqrt{\frac{K \sum_{\ell=1}^{h} Rd(y; t_\ell) w_{\ell j}^2}{Rd(y; t_1, \ldots, t_h)}}$$

et

$$Rd(y; t_1, \ldots, t_h) := \frac{1}{K} \sum_{\ell=1}^{h} \text{cor}^2(y, t_\ell) =: \sum_{\ell=1}^{h} Rd(y; t_\ell).$$

où $\text{cor}^2(y, t_\ell)$ est le coefficient de corrélation de Pearson entre $y$ et la composante $t_\ell$. Cette information est rétro-propagée dans le modèle (une seule fois) afin d'obtenir les variables latentes $t_{h^*}$ et leurs coefficients estimés $\hat{c}_{h^*}$ sur les données d'entraînement. La variable cible $y$ est ensuite prédite grâce à (3.3.4). Cette prévision est comparée aux modèles standards SVM, LOGIT, Bayes et LDA lorsque les données tests sont projetées dans le sous-espace $\{t_1, \ldots, t_{h^*}\}$.

**Algorithme 4 :** Gini-PLS Généralisé

**Résultat :** Prédiction du juge $y = 0; 1$

**répéter**
  **répéter**
    $\max \text{cog}_\nu(y, w_h X)$ s.t. $\|w_h\| = 1 \Longrightarrow$ poids $w_h$ de $X$ ;
    MCO équation : $y = \sum_h c_h t_h + \varepsilon_h$ ;
    MCO équation : $R(x_j) = \sum_h \beta_h t_h + \epsilon_k \; \forall k = 1, \ldots, K$ ;
    $X := (\hat{\epsilon}_1, \ldots, \hat{\epsilon}_K)$ $y := \hat{\varepsilon}_h$ ;
  **jusqu'à** $h = 10$ $[h = h + 1]$;
  Mesurer $VIP_{kh}$, $Q_h^2$ ;
  Sélectionner le nombre optimal de composantes $h^*$ ;
**jusqu'à** $\nu = 14$ $[\nu = \nu + 2]$;
Déduire le paramètre optimal $\nu^*$ qui minimise l'erreur ;
**retourner** Prédiction $\hat{y}$ avec Gini-PLS $(h^*, \nu^*)$ ;
**retourner** Prédiction $\hat{y}$ avec SVM, LOGIT, Bayes, LDA sur les composantes $(t_1, \ldots, t_{h^*})$;

#### 3.3.3.2 L'algorithme LOGIT-Gini-PLS généralisé

Comme nous le constatons dans l'algorithme Gini-PLS généralisé que nous avons proposé dans le section précédente, les poids $w_j$ proviennent de l'opérateur co-Gini appliqué à une variable booléenne $y \in \{0; 1\}$. Afin de trouver les poids $w_j$ qui maximisent le lien entre les variables $x_j$ et la variable cible $y$, nous proposons d'utiliser la régression LOGIT, autrement dit, une sigmoïde qui est mieux adaptée aux variables booléennes. Ainsi, dans chaque étape de la régression Gini-PLS nous remplaçons la maximisation du co-Gini par la mesure de la probabilité conditionnelle suivante :

$$\mathbb{P}(y_i = 1/X = X_i) = \frac{\exp\{X_i\beta\}}{1 + \exp\{X_i\beta\}} \qquad \text{(LOGIT)}$$

où $X_i$ est la $i$-ème ligne de la matrice $X$ (observation des caractéristiques/dimensions de la décision juridique $i$). L'estimation du vecteur $\beta$ se fait par maximum de vraisemblance. On en déduit alors les pondérations $w_j$ :

$$w_j = \frac{\beta_j}{\|\beta\|}$$

L'algorithme LOGIT-Gini-PLS généralisé est donc le suivant :

**Algorithme 5 :** LOGIT-Gini-PLS Généralisé

**Résultat :** Prédiction du juge $y = 0; 1$
**répéter**
  **répéter**
    LOGIT équation : $\Longrightarrow$ poids $w_j$ de $X$ ;
    MCO équation : $y = \sum_h c_h t_h + \varepsilon_h$ ;
    $X := (\hat{\varepsilon}_1, \ldots, \hat{\varepsilon}_K)$ $y := \hat{\varepsilon}_h$ ;
  **jusqu'à** $h = 10$ $[h = h + 1]$;
  Mesurer $VIP_{kh}$, $Q_h^2$ ;
  Sélectionner le nombre optimal de composantes $h^*$ ;
**jusqu'à** $\nu = 14$ $[\nu = \nu + 2]$;
Déduire le paramètre optimal $\nu^*$ qui minimise l'erreur ;
**retourner** Prédiction $\hat{y}$ avec Gini-PLS $(h^*, \nu^*)$ ;
**retourner** Prédiction $\hat{y}$ avec SVM, LOGIT, Bayes, LDA sur les composantes $(t_1, \ldots, t_{h^*})$;

## 3.4 Expérimentations et résultats

Nous discutons ici les performances de divers algorithmes populaires et l'impact de la quantité et du déséquilibre des données, de l'heuristique, et de la restriction explicite des documents aux passages relatifs à la catégorie de demandes, ainsi que leur capacité à faire abstraction des autres demandes du document. Ces expériences visent aussi à comparer l'efficacité du Gini-Logit-PLS par rapport à d'autres analyses discriminantes. Comme Im *et al.* [2017], différentes combinaisons d'algorithmes de classification et méthodes de pondération sont comparées ; ce qui représente un total de 8 algorithmes * 11 métriques globales * 5 métriques locales = 440

configurations (+ 2 algorithmes i.e. FastText et NBSVM). La méthode Gini-Logit-PLS réalisant une projection dans un espace de petite dimension, elle a été comparée aussi à d'autres méthode de réduction de dimension.

### 3.4.1 Protocole d'évaluation

Deux métriques d'évaluation sont utilisées : la précision et la $F_1$-mesure. Pour tenir compte du déséquilibre entre les classes, la moyenne macro est préférée. Il s'agit de l'agrégation de la contribution individuelle de chaque classe : $F_{1macro} = \frac{2 \times P_{macro} \times R_{macro}}{P_{macro} + R_{macro}}$, où les macro-moyennes de la précision ($P_{macro}$) et du rappel ($R_{macro}$) sont calculées en fonction des nombres moyens de vrais positifs ($\overline{TP}$), faux positifs ($\overline{FP}$), et faux négatifs ($\overline{FN}$) comme suit [Van Asch, 2013] : $P_{macro} = \frac{\overline{TP}}{\overline{TP}+\overline{FP}}$, $R_{macro} = \frac{\overline{TP}}{\overline{TP}+\overline{FN}}$.

Les données utilisées sont une restriction, des données du chapitre précédent, aux documents n'ayant qu'une seule demande annotée pour chacune des catégories de demande. Le déséquilibre entre les classes est illustrée par la figure 3.3. En effet, les demandes sont en majorité rejetées pour les catégories ACPA, CONCDEL, DANAIS et STYX. Le contraire est observé pour DCPPC, et le rapport est légèrement équilibré pour DORIS.

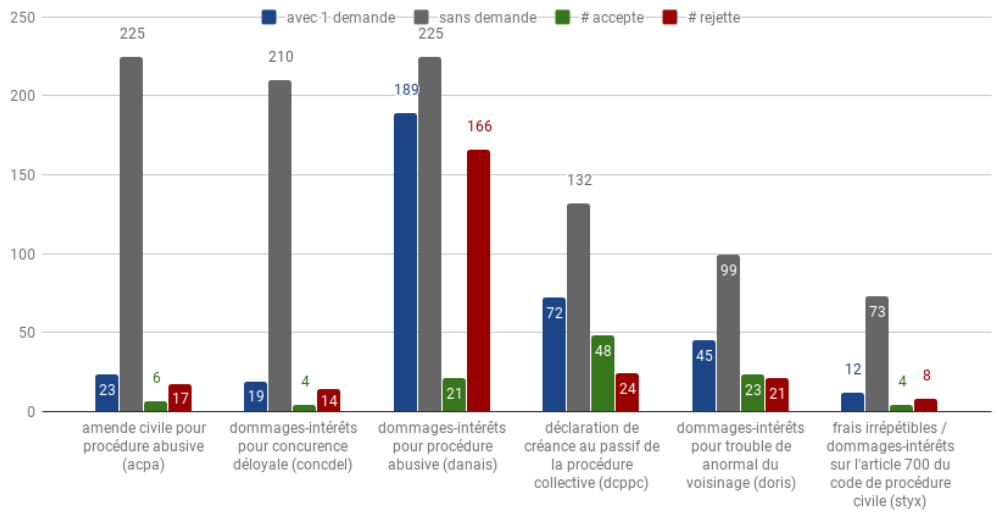



Figure 3.3 – Répartition des documents à une demande de la catégorie considérée.

L'efficacité des algorithmes dépend souvent des valeurs de méta-paramètres

dont il faut déterminer des valeurs optimales. Scikit-learn implémente deux stratégies de recherche des ces valeurs : RandomSearch et GridSearch. Malgré la rapidité de la méthode RandomSearch, elle est non déterministe et les valeurs qu'elle trouve donnent une prédiction moins précise que les valeurs par défaut. Idem pour la méthode GridSearch, qui est très lente, et donc peu pratique face au grand nombre de configurations à évaluer. Par conséquent, les valeurs utilisées pour les expérimentations sont les valeurs par défaut définies par Scikit-learn (Tableau 3.1).

| algorithmes | hyper-paramètres |
|---|---|
| SVM | $C = 1.0; \gamma = \frac{1}{n\_features * var(X)}; noyau = RBF$ (fonction de base radiale) |
| KNN | $k = 5, weights =' uniform', algorithm =' auto'$ |
| LDA | $solver =' svd', n\_components = 10^{10}$ |
| QDA | |
| Arbre | critère de séparation='gini' |
| NBSVM | $--ngram = 123, linearSVM,$ |
| FastText | -minCount=5, -wordNgrams=1, -lr=0.05, |
| Gini-PLS | $h = n\_components = 10$ |
| Logit-PLS | $h = n\_components = 10$ |
| Gini-Logit-PLS | $h = n\_components = 10; \nu = 14$ |

Tableau 3.1 – Valeurs utilisées pour les méta-paramètres des algorithmes de classification.

### 3.4.2 Classification de l'ensemble du document

En représentant l'ensemble du document à l'aide de diverses représentations vectorielles, les algorithmes sont comparés avec les représentations qui leurs sont optimales. On remarque d'après les résultats du Tableau 3.2 que les arbres sont en moyenne meilleurs sur l'ensemble des catégories même si en moyenne la $F_1$-mesure moyenne est limité à 0.668. Les résultats des extensions du PLS ne sont pas très éloignées de ceux des arbres avec des différences de $F_1$ à moins de 0.1 (si on choisit le bon schéma de "vectorisation").

Les scores $F_1$ moyens des algorithmes NBSVM et FastText n'excédent en général pas 0.5 malgré qu'ils soient spécialement conçus pour les textes. On peut estimer qu'ils sont très sensibles au déséquilibre des données entre les catégories (plus de rejets que d'acceptations), soit il est plus difficile de détecter l'acceptation des demandes. En effet, ces algorithmes classent toutes les données test avec le label (sens) majoritaire i.e. le rejet, et par conséquent, ils ne détectent quasiment pas d'acceptation de demande.

ajouter les $F_1$ ou erreur de rejette et de accepte
mettre aussi en avant les analyses discriminantes comme réducteur de dimension et comme classifieurs

| **Vecteur** | **algorithme** | $F_1$ | **min** | **Cat. min** | **max** | **Cat. max** | $F_1$ **- 1er**$F_1$ | **max - min** | **rang** |
|---|---|---|---|---|---|---|---|---|---|
| GSS*TF | Arbre | 0.668 | 0.5 | doris | 0.92 | dcppc | 0 | 0.42 | 1 |
| AVG-G*TF | LogitPLS | 0.648 | 0.518 | danais | 0.781 | dcppc | 0.02 | 0.263 | 13 |
| AVG-G*TF | StandardPLS | 0.636 | 0.49 | danais | 0.836 | dcppc | 0.032 | 0.346 | 24 |
| DELTADF*TF | GiniPLS | 0.586 | 0.411 | danais | 0.837 | dcppc | 0.082 | 0.426 | 169 |
| DELTADF*TF | GiniLogitPLS | 0.578 | 0.225 | styx | 0.772 | dcppc | 0.09 | 0.547 | 220 |
| - | NBSVM | 0.494 | 0.4 | styx | 0.834 | dcppc | 0.174 | 0.434 | |
| - | FastText | 0.412 | 0.343 | doris | 0.47 | danais | 0.256 | 0.127 | |

Tableau 3.2 – Comparaison des algorithmes sur une représentation globale des documents pour la détection du sens du résultat.

Le cas des catégories DORIS et DCPPC pour le NBSVM ($F_{1macro} = 0.834$) tend à démontrer la forte sensibilité aux cas négatifs de ces algorithmes puisque même avec presque autant de labels "accepte" que "rejette", la $F_1$-mesure de "rejette" est toujours supérieure à celle de "accepte" (Tableau 3.3).

| **Cat. Dmd.** | **Algo.** | **Préc.** | **Préc. équi.** | **err-0** | **err-1** | $F_1(0)$ | $F_1(1)$ | $F_{1macro}$ |
|---|---|---|---|---|---|---|---|---|
| **dcppc** | nbsvm | 0.875 | 0.812 | 0.375 | 0 | 0.752 | 0.916 | **0.834** |
| danais | fasttext | 0.888 | 0.5 | 0 | 1 | 0.941 | 0 | 0.47 |
| danais | nbsvm | 0.888 | 0.5 | 0 | 1 | 0.941 | 0 | 0.47 |
| concdel | fasttext | 0.775 | 0.5 | 0 | 1 | 0.853 | 0 | 0.437 |
| concdel | nbsvm | 0.775 | 0.5 | 0 | 1 | 0.873 | 0 | 0.437 |
| acpa | fasttext | 0.745 | 0.5 | 0 | 1 | 0.853 | 0 | 0.426 |
| acpa | nbsvm | 0.745 | 0.5 | 0 | 1 | 0.853 | 0 | 0.426 |
| doris | nbsvm | 0.5 | 0.492 | 0.85 | 0.167 | 0.174 | 0.63 | 0.402 |
| dcppc | fasttext | 0.667 | 0.5 | 1 | 0 | 0.8 | 0 | 0.4 |
| styx | fasttext | 0.667 | 0.5 | 1 | 0 | 0.8 | 0 | 0.4 |
| styx | nbsvm | 0.667 | 0.5 | 0 | 1 | 0.8 | 0 | 0.4 |
| doris | fasttext | 0.523 | 0.5 | 1 | 0 | 0 | 0.686 | 0.343 |

$0 ==' accepte'; 1 ==' rejette'$

Tableau 3.3 – Détails des résultats de FastText et NBSVM.

### 3.4.3 Réduction du document aux régions comprenant le vocabulaire de la catégorie

Etant donné que les décisions portent sur plusieurs catégories de demande, nous avons expérimenté la restriction du document aux passages comprenant du vocabulaire de la catégorie d'intérêt : demande, résultat,

résultat antérieur (resultat_a), paragraphes dans les motifs (motifs). Les combinaisons passages-représentation vectorielle-algorithme sont comparées dans le Tableau 3.4. Les résultats s'améliorent énormément avec les réductions, sauf pour la catégorie DORIS. La meilleure restriction combine les passages comprenant le vocabulaire de la catégorie dans la section Litige (demande et résultat antérieur), dans la section Motifs (contexte), et dans la section Dispositif (résultat).

| Catégorie | zone | Vecteur (pondération) | classifieur | $F_1$ |
|---|---|---|---|---|
| acpa | **demande_resultat_a_resultat_context** | **DBIDF*TF** | **Tree** | **0.846** |
| | litige_motifs_dispositif | DELTADF*TF | StandardPLS | 0.697 |
| | litige_motifs_dispositif | AVERAGEGlobals*TF | LogitPLS | 0.683 |
| concdel | **litige_motifs_dispositif** | **GSS*TF** | **Tree** | **0.798** |
| | motifs | IDF*TF | GiniLogitPLS | 0.703 |
| | context | DBIDF*LOGAVE | StandardPLS | 0.657 |
| danais | **demande_resultat_a_resultat_context** | **CHI2*AVERAGELocals** | **Tree** | **0.813** |
| | demande_resultat_a_resultat_context | AVERAGEGlobals*ATF | LogitPLS | 0.721 |
| | demande_resultat_a_resultat_context | AVERAGEGlobals*ATF | StandardPLS | 0.695 |
| dcppc | **demande_resultat_a_resultat_context** | **CHI2*TF** | **Tree** | **0.985** |
| | demande_resultat_a_resultat_context | CHI2*TF | LogitPLS | 0.94 |
| | litige_motifs_dispositif | MARASCUILO*TP | StandardPLS | 0.934 |
| doris | **litige_motifs_dispositif** | **DSIDF*TP** | **GiniPLS** | **0.806** |
| | litige_motifs_dispositif | DSIDF*TP | GiniLogitPLS | 0.806 |
| | litige_motifs_dispositif | IG*ATF | StandardPLS | 0.772 |
| styx | **motifs** | **DSIDF*TF** | **Tree** | **1** |
| | demande_resultat_a_resultat_context | DSIDF*LOGAVE | GiniLogitPLS | 0.917 |
| | litige_motifs_dispositif | RF*TF | GiniPLS | 0.833 |

Tableau 3.4 – Détection du sens du résultat : Comparaison des réductions du document.

## 3.5 Conclusion

L'étude de ce chapitre tente de simplifier l'extraction du sens du résultat rendu par les juges sur une demande de catégorie donnée. Elle a consisté à formuler le problème comme une tâche de classification de documents. On évite ainsi de passer par la détection ad-hoc [11] des passages et données à l'aide de termes-clés qui est un inconvénient de la méthode à règles du chapitre précédent car elle n'est peut-être pas généralisable à tous types de décisions (i.e. il pourrait être nécessaire d'établir de nouvelles listes de mots-clés pour d'autres domaines). Au total dix algorithmes de classification on été expérimentés sur 55 méthodes de représentations

11. i.e. spécialement conçu pour nos données.

vectorielles de texte. Nous avons remarqué que les résultats de classification sont principalement influencés par 3 caractéristiques de nos données. Tout d'abord, le très faible nombre d'exemples d'entraînement défavorise certains algorithmes (sensibilité aux valeurs aberrantes ou *outliers*), comme par exemple FastText qui nécessite plusieurs milliers d'exemples pour mettre à jour le pas du gradient (*learning rate*). Ensuite, le fort déséquilibre entre les classes ("accepte" vs. "rejette") rend difficile la reconnaissance de la classe minoritaire qui est généralement la classe "accepte". Le fort gap entre les erreurs sur "rejette" et celles sur "accepte", ainsi que les bons résultats obtenus sur DCPPC en sont la preuve. Enfin, la présence d'autres catégories de demande dans le document dégrade l'efficacité de la classification parce que les algorithmes ne parviennent pas seuls à retrouver les éléments en rapport direct avec la catégorie choisie. Ceci est démontré par l'impact positif de la restriction du contenu à classer à certains passages particuliers, même si la restriction adéquate est fonction de la catégorie.

Au final, les arbres de décision sont adaptés pour la tâche, mais l'usage du Gini-PLS et du Gini-Logit-PLS permet d'obtenir des performances assez proches de celles des arbres. Il serait intéressant de combiner ces variantes de l'analyse PLS, à d'autres comme le Sparse-PLS qui pourrait peut-être aider à résoudre le problème de vecteurs/matrices creuses dont sont victimes les représentations vectorielles de texte. Il existe aussi un grand nombre d'architectures neuronales pour la classification de document et de très grands nombres de métriques de pondération de termes pour la représentation des textes, mais aucune ne semble s'adapter à toutes les catégories. Par conséquent, une étude sur l'usage des représentations par plongement sémantique comme Word2Vec [Mikolov *et al.* , 2013], Sent2Vec [Pagliardini *et al.* , 2018] ou Doc2Vec [Le & Mikolov, 2014b] serait intéressante.

# Chapitre 4

# Découverte des circonstances factuelles

## 4.1 Introduction

Les circonstances factuelles définissent les contextes possibles dans lesquels une catégorie de demande peut être formulée (voir § 4.4.1 pour des exemples). Les analyses descriptives ou prédictives ne prennent sens que lorsqu'elles sont appliquées à un ensemble de décisions aux circonstances similaires. Par exemple, il serait imprudent de considérer toutes les décisions pour analyser les chances d'acceptation d'une demande de dommages et intérêts fondée sur l'« article 700 du code de procédure civile ». Les taux d'acceptation ou de rejet peuvent être différents entre des affaires de licenciement et celles portant sur les troubles anormaux du voisinage, et même plus spécifiquement entre des troubles de voisinage entre particuliers et entreprises. Il est indispensable de travailler uniquement avec des décisions similaires à la situation d'intérêt. L'identification des circonstances factuelles devient donc une étape préalable indispensable à l'analyse du résultat. Malheureusement, les circonstances sont très diverses et quasi infinies pour être identifiées par classification supervisée à l'aide d'annotation manuelle d'exemples comme dans les chapitres précédents. Il est donc plus adéquat d'adopter une approche non-supervisée capable de découvrir les circonstances factuelles à partir d'un corpus de documents d'une même catégorie de demandes. Plus précisément, la méthode doit construire des sous-ensembles de décisions partageant des situations similaires. L'objectif de ce chapitre est d'expérimenter des algorithmes de regroupement (*clustering*) et des métriques de similarité généralement utilisées sur les textes. Ce chapitre propose aussi une méthode d'apprentissage d'une distance qui est basée sur la transformation de documents, et montre qu'une telle métrique permet de bien mesurer la (dis-) similarité

sémantique définie par les circonstances factuelles.

## 4.2 Catégorisation non-supervisée de documents

Cette section fait une synthèse bibliographique de différents aspects qui rentrent dans la conception d'un système de regroupement de documents. Elle aborde principalement le choix de l'algorithme, la définition d'une mesure de similarité, la représentation des documents, la détermination du nombre de groupes (appelés *clusters*), et l'évaluation de la catégorisation générée. Le corpus à catégoriser est noté $\mathcal{D}$ et comprend $N$ documents. Tout document $d \in \mathcal{D}$ est une séquence de mots $d = (d[1], \ldots, d[|d|])$, où $d_i$ est le mot à la position $i$ dans $d$. Sa représentation vectorielle est notée $\vec{d} = (\vec{d}[1], \vec{d}[2], \ldots, \vec{d}[m])$. Pour un modèle vectoriel de type TF-IDF de vocabulaire $T = \{t_1, t_2, \ldots, t_m\}$, $\vec{d}[i] = w(t_i, d)$ le poids du terme $t_i \in T$ dans le texte $d$ (cf. § 2.3.1). La catégorisation obtenue est un ensemble de clusters $C = \{C_1, C_2, \cdots, C_K\}$, $K$ étant le nombre de clusters formés.

### 4.2.1 Algorithmes de catégorisation non-supervisé

La catégorisation de documents a pour objectif d'identifier, sans supervision [1], une organisation pertinente (pour le domaine expert) de l'ensemble $\mathcal{D}$ en construisant des groupes représentants des catégories inconnues au départ. Ces groupes, appelés *clusters*, peuvent être disjoints ou se chevaucher, organisés de manière plate ou hiérarchique suivant les contraintes du domaine expert. L'algorithme à utiliser dépend généralement de la forme qu'on souhaite donner à l'organisation.

#### 4.2.1.1 Partitionnement disjoint

Pour réaliser des partitions distinctes [2] (*hard clustering*), des algorithmes tels que celui des K-moyennes (*K-means*) [Forgey, 1965] et celui des K-medoïdes (*K-medoids*) [Kaufman & Rousseeuw, 1987] sont les plus simples [Balabantaray *et al.* , 2015]. Ces deux algorithmes fonctionnent de manière

1. Sans utiliser des exemples annotés.
2. Chaque document n'appartient qu'à un seul cluster.

similaire, et nécessitent que le nombre $K$ de clusters soit prédéfini. Ils commencent par une définition aléatoire de $K$ centres initiaux de clusters (*centroïdes*) et l'affectation des différents documents au cluster dont le centre est le plus proche. S'en suit une boucle dans laquelle le centroïde est recalculé (le point dont la somme des distances aux membres du cluster est minimale) et les documents sont réaffectés chacun au cluster dont le centroïde est le plus proche. L'algorithme s'arrête si aucune amélioration n'est plus observée, ce qui se traduit soit par l'atteinte d'une valeur minimale prédéfinie de l'erreur de catégorisation [3] ou d'une mesure d'évaluation non supervisée (§ 4.2.5.2). La différence entre l'algorithmes des K-moyennes et celui des K-medoïdes tient principalement au fait que les centroïdes du premier ne sont pas nécessairement des points (documents) de l'ensemble d'origine, mais des points moyennes des représentations vectorielles des membres du cluster, contrairement à l'algorithme des K-medoïdes qui ne considère comme centres que des documents de $\mathcal{D}$. Cette différence donne l'avantage au K-medoïdes de ne pas dépendre d'une représentation vectorielle nécessaire au calcul de la moyenne, mais elle a aussi l'inconvénient d'augmenter sa complexité en temps et en espace car il faut calculer et stocker la distance entre toutes les paires de documents. Il existe plusieurs autres algorithmes de partitionnement dont le principe est différent de celui des K-moyennes. Par exemple, l'algorithme DBSCAN (*Density-based spatial clustering of applications with noise*) [Ester *et al.* , 1996] ne prend pas en paramètre le nombre de clusters à construire. Il est défini sur le concept de régions de densité caractérisées par la distance minimale $\epsilon$ autorisée entre deux points d'une même région, et le nombre maximal de points qui doivent être dans le voisinage de rayon $\epsilon$ d'un point pour que ce voisinage soit une région de densité (le point central est appelé "point noyau" (*core point*). Le principe du DBSCAN est de construire les clusters successivement en reliant les régions (voisinages) dont les noyaux sont à distance plus ou moins inférieure à $\epsilon$. Les points qui sont seuls dans leur cluster sont qualifiés de points aberrants (*outliers*).

La catégorisation spectrale est une autre méthode efficace de partitionnement qui effectue préalablement une réduction de dimensions à l'aide du spectre [4] de la matrice de similarité $M \in \mathbb{R}^{N \times N}$ [5] des données avant

3. Somme des distances au carré entre les points et leur centre respectif.
4. Le spectre d'une matrice est l'ensemble de ses valeurs propres
5. $M_{ij}$ est la mesure de la similarité entre les élément $d_i$ et $d_j$ de $\mathcal{D}$.

d'appliquer un algorithme traditionnel comme celui des K-moyennes. Les dimensions du nouvel espace sont définies par les vecteurs propres de la matrice Laplacienne $L$ de $M$ [Shi & Malik, 2000; Von Luxburg, 2007] qui peut être normalisée ($L = T^{-1/2}(T - S)T{-}1/2$) ou pas ($L = T - M$), $T$ étant la matrice diagonale déduite de $M$ i.e. $T_{ii} = \sum_j M_{ij}$.

Il est aussi possible d'utiliser les arbres de décision pour améliorer les résultats des K-moyennes. En effet, les forêts aléatoires [Breiman, 2001] permettent d'estimer la similarité entre deux points. Le principe consiste à générer un ensemble de $n$ points synthétiques, et d'entraîner une forêt aléatoire à une classification binaire supervisée avec les points originaux considérés dans la classe des "originaux" et les données synthétiques dans la seconde classe des "synthétiques" [Afanador *et al.* , 2016]. Une forêt aléatoire construit des arbres de décision sur des parties de l'ensemble d'apprentissage, auxquelles on a retiré une ou plusieurs variables prédictives. La similarité entre 2 points est la proportion d'arbres dans lesquels ces points se trouvent dans le même nœud feuille. Cette métrique "apprise" peut-être par la suite utilisée dans un algorithme comme les K-moyennes.

#### 4.2.1.2 Catégorisation avec chevauchements

Les regroupements avec chevauchement sont intéressants parce qu'il est possible qu'une décision traite de plusieurs circonstances factuelles. Lorsque des chevauchements sont observables entre clusters [6], un degré d'appartenance (*membership degree*) d'un document à chaque groupe est estimé par une fonction $u_{ij}, \forall d_i \in \mathcal{D}, \forall j \in [1..K]$ [Baraldi & Blonda, 1999]. Ce degré d'appartenance est employé dans des algorithmes de partitionnement "flou" comme l'algorithme des c-moyennes flou (FCM) [Bezdek *et al.* , 1984; Hathaway *et al.* , 1989], ou le fuzzy c-Medoids (FDMdd) [Krishnapuram *et al.* , 2001], ou la version améliorée IFKM (*improved fuzzy K-medoids*)[Sabzi *et al.* , 2011]. Nefti & Oussalah [2004] proposent par ailleurs les C-moyennes floues probabilistes qui sont une variante du FCM pour laquelle la somme des degrés d'appartenance d'un document aux clusters est de 1.

Ces algorithmes consistent en deux étapes principales [Sabzi *et al.* , 2011] :

6. Un chevauchement est l'appartenance d'un document à plusieurs groupes.

1. l'estimation des degrés d'appartenance de chaque instance $d_i \in \mathcal{D}$ à chaque cluster $j \in [1..K]$ de centroïde $z_j$ est réalisée par la minimisation de la fonction objectif $P(\mathcal{D}, Z) = \sum_{i=1}^{N} \sum_{j=1}^{K} \left[u_{ij} r(d_i, z_j)\right]$ [Krishnapuram *et al.* , 2001] améliorée par Sabzi *et al.* [2011] en :

$$P(\mathcal{D}, Z) = \sum_{i=1}^{N} \sum_{j=1}^{K} \left[u_{ij} r(d_i, z_j)\right] + \lambda \sum_{i=1}^{N} \sum_{j=1}^{K} \left[u_{ij} \log_2(u_{ij})\right]$$

$$s.c. \sum_{j=1}^{K} u_{ij} = 1, 0 \leq u_{ij} < 1$$

dont la valeur approximative de la solution est

$$u_{ij} = \frac{\exp\left(\frac{-r(d_i, z_j)}{\lambda}\right)}{\sum_{l=1}^{k} \exp\left(\frac{-r(d_l, z_j)}{\lambda}\right)},$$

$r(d_i, z_j)$ étant la distance entre $d_i$ et $z_j$

2. Le nouveau centre de chaque cluster $C_j$ est redéfini comme étant la moyenne des membres de chaque cluster chez le FCM. Mais pour les K-medoïdes flous, il s'agit du membre $z_j$ dont la somme des distances pondérées [7] aux autres membres est minimale :

$$\forall j \in [1 \,..\, K], z_j = \underset{d_q \in C_j}{\operatorname{argmin}} \sum_{d_i \in C_j} \left[u_{ij} r(d_i, d_q)\right].$$

Ainsi l'objectif de l'entraînement des algorithmes de la catégorisation floue est double : déterminer les degrés optimaux d'appartenance $u_{ij}$ et l'ensemble $Z$ des centroïdes.

#### 4.2.1.3 Catégorisation hiérarchique

La catégorisation hiérarchique consiste à construire une hiérarchie de clusters. Le regroupement hiérarchique ascendant ou regroupement *agglomératif* (*Agglomerative Clustering*) est une technique de catégorisation hiérarchique qui commence par autant de clusters que de documents (chacun des groupes comprenant un document). Ensuite, l'algorithme détecte

7. La distance est pondérée par le degré d'appartenance de l'autre membre.

et fusionne successivement les paires de groupes dont la fusion vérifie un critère (par exemple la paire est celle dont la distance est la plus petite et/ou proche d'une valeur seuil donnée, ou bien la fusion forme un groupe à inertie [8]minimale), jusqu'à ce que tous les documents soient dans un unique groupe (racine). Pour déterminer le partitionnement optimal, le nombre de clusters doit être déterminé par l'une des diverses techniques existantes [Thorndike, 1953; Salvador & Chan, 2004].

### 4.2.2 Métriques de dis-similarité (distances)

Les algorithmes de catégorisation dépendent de la distance utilisée qui doit être bien choisie pour que le résultat révèle au mieux la sémantique visée. Une distance $Dis$ est une fonction réelle d'une paire de documents $(d, d')$ qui mesure le degré de différence ou dis-similarité entre $d$ et $d'$ en satisfaisant aux propriétés suivantes $\forall d, d', d'' \in \mathcal{D}$ [Harispe *et al.* , 2015; Wang & Sun, 2015] :

1. $Dis(d, d') \geq 0$ ("non-négativité")
2. $Dis(d, d') = 0 \Leftrightarrow d = d'$ (identité des indiscernables)
3. $Dis(d, d') = Dis(d', d)$ (symétrie)
4. $Dis(d, d'') \leq Dis(d, d') + Dis(d', d'')$ (inégalité triangulaire)

La métrique peut être normée ($\forall (d, d') \in \mathcal{D} \times \mathcal{D}; 0 \leq Dis(d, d') \leq 1$). Dans ce cas, la relation entre la similarité $Sim$ et la dis-similarité $Dis$ est définie par $Sim(d, d') = 1 - Dis(d, d')$.

Parmi les nombreuses métriques généralement utilisées sur les textes [Huang, 2008; Vijaymeena & Kavitha, 2016; Afzali & Kumar, 2018], on retrouve par exemple :

- Les distances de Minkowski $Dis(d, d') = \left\| \vec{d} - \vec{d'} \right\|_{Lp} = \sqrt[p]{\sum_{i=1}^{m} |\vec{d}[i] - \vec{d}[i]|^p}$, dont font partie la distance euclidienne $Dis_{euclidienne}$ ($p = 2$) et la distance de Manhattan $Dis_{manhattan}$ ($p = 1$).
- La distance de Bray & Curtis [1957] : $Dis_{braycurtis}(d, d') = \frac{\sum_{i=1}^{m} |\vec{d}[i] - \vec{d'}[i]|}{\sum_{i=1}^{m} |\vec{d}[i] + \vec{d'}[i]|}$ [Huang, 2008].

8. L'inertie d'un cluster est la variance de ses points c'est-à-dire la somme des erreurs (distance d'un membre au centre) au carré.

- La similarité cosinus basée sur le cosinus de l'angle entre $\vec{d}$ et $\vec{d'}$ par la formule : $Sim_{cos}(d,d') = \frac{\vec{d}^t\vec{d'}}{\|\vec{d}\|\|\vec{d'}\|}$. Pour un modèle vectoriel du type TF-IDF, cette formulation considère que tous les termes du vocabulaire $T$ sont différents et ne partagent aucune relation. Sidorov *et al.* [2014] la corrigent en proposant la fonction *soft-cosine* utilisant la matrice de similarité entre termes $S = \{s_{ij}\}_{1\leq i,j\leq m}$ :

  $$Sim_{soft-cos}(d,d') = \frac{\vec{d}^T\cdot S\cdot\vec{d'}}{\sqrt{\vec{d}^T\cdot S\cdot\vec{d'}}\cdot\sqrt{\vec{d'}^T\cdot S\cdot\vec{d'}}} = \frac{\sum\limits_{1\leq i,j\leq m} s_{ij}\vec{d}[i]\vec{d'}[j]}{\sqrt{\sum\limits_{1\leq i,j\leq m} s_{ij}\vec{d}[i]\vec{d}[j]}\sqrt{\sum\limits_{1\leq i,j\leq m} s_{ij}\vec{d'}[i]\vec{d'}[j]}}.$$

  $S$ peut être calculée à partir de n'importe quelle métrique comme la distance d'édition de Levenshtein [Sidorov *et al.* , 2014], la similarité cosinus entre plongements lexicaux [Charlet & Damnati, 2017, 2018], ou la similarité WordNet. La fonction cosinus étant comprise entre -1 et +1, la distance déduite $Dis_{cos}(d,d') = 1 - Sim_{cos}(d,d')$ est comprise entre 0 et 2.

- Le coefficient similarité de Jaccard [1901] : $Sim_{Jaccard}(d,d') = \frac{\vec{d}^T\vec{d'}}{\|\vec{d}\|^2+\|\vec{d'}\|^2-\vec{d}^T\vec{d'}}$ [Huang, 2008] qui donne la distance $Dis_{jaccard}(d,d') = 1 - Sim(d,d')$.

- La similarité basée sur le coefficient de corrélation de Pearson est calculée comme suit [Huang, 2008] :

  $Sim_{pearson}(d,d') = \frac{\sum\limits_{i=1}^{m}\vec{d}[i]\cdot\vec{d'}[i] - TF_d\cdot TF_{d'}}{\sqrt{[m\sum\limits_{i=1}^{m}\vec{d}[i]^2 - TF_d^2][m\sum\limits_{i=1}^{m}\vec{d'}[i]^2 - TF_{d'}^2]}}$, avec $TF_d = \sum\limits_{i=1}^{m}\vec{d}[i]$. Sa distance est déduite par la formule :

  $$Dis_{pearson}(d,d') = \begin{cases} 1 - Sim_{pearson}(d,d') & \text{si } Sim_{pearson}(d,d') \geq 0 \\ |Sim_{pearson}(d,d')| & \text{si } Sim_{pearson}(d,d') < 0. \end{cases}$$

- « La distance du déménageur de mot » (*word mover's distance - WMD*) [Kusner *et al.* , 2015] est incluse la similarité sémantique entre les mots dans l'estimation de la distance entre documents. En effet, elle est la solution optimale du problème de transport suivant [9] :

9. Valeur minimale du coût cumulatif pondéré nécessaire pour déplacer tous les mots de $d$ à $d'$ i.e. transformer $d$ en $d'$.

$$Dis_{wmd}(d,d') = \min_{T>0} \sum_{i,j=1}^{m} T_{ij}c(i,j)$$
$$\text{s.c.} \quad \sum_{j=1}^{m} T_{ij} = \vec{d}[i], \forall i \in [1\,..\,m]; \sum_{i=1}^{m} T_{ij} = \vec{d'}[j], \forall j \in [1\,..\,m]$$

$m$ est le nombre de mots considérés ; $T$ est une matrice dont $T_{ij}$ est interprété comme étant la quantité du mot $i$ de $d$ qui est va au mot $j$ dans $d'$ ("voyage"). $c(i,j)$ est la distance euclidienne entre les vecteurs des mots $i$ et $j$ ; $\vec{d}[i] = \frac{compte(i,d)}{\sum_{k=1}^{m} compte(k,d)}$, $compte(i,d)$ étant le nombre d'occurrences du mot $i$ dans $d$.

## 4.2.3 Représentation des textes

### 4.2.3.1 Modèle vectoriel

La formulation des distances (cf. § 4.2.2) s'applique généralement à une représentation vectorielle des textes. Le chapitre 2 décrit différentes métriques de pondération des termes qui permettent de définir des variantes du modèle TF-IDF. Cependant, il s'agit de modèles basés uniquement sur le lexique des textes, et par conséquent, les distances appliquées sur ces représentations correspondent à une similarité plus lexicale que sémantique. Il existe quelques techniques permettant de définir des dimensions par des thématiques qui transparaissent dans le corpus, et apportant par conséquent plus de sémantique à la représentation vectorielle.

### 4.2.3.2 Réduction de dimension

La réduction de dimension a généralement pour but de réduire la taille de la représentation de documents i.e. transformer leur vecteur de dimension $m$ en un nouveau de dimension $q << m$. Les méthodes expérimentées ici sont non supervisées [10] et génèrent de nouvelles dimensions [11].

10. Elles ne prennent en compte aucune classification prédéfinie des documents.
11. Par opposition aux algorithmes de sélection de caractéristiques comme le BDS et le SFFS expérimentées au Chapitre 1.

**L'analyse en composantes principales (ACP)** [Burrows, 1992] est une technique de transformation de l'espace originel en un espace de dimension réduite préservant au mieux l'inertie du nuage originel [12]. L'intérêt de l'ACP se traduit par une meilleur interprétation des données dans le nouvel espace. Son principe consiste à construire successivement les composantes principales qui sont des combinaisons linéaires des observations ; chaque composante étant orthogonale à la suivante. Plus précisément, les colonnes de la matrice document-terme $A$ de dimensions $N \times m$ sont préalablement transformée en des variables centrées réduites [13] résultant en une matrice $\hat{A}$. Ensuite, l'ACP décompose $\hat{A}$ en un produit de trois matrices : $U$ la matrice $N \times N$ des vecteurs propres de $\hat{A}\hat{A}^T$, $S$ la matrice diagonale $m \times m$ des valeurs propres de la plus grande à la plus petite, et $V$ la matrice $m \times N$ des vecteurs propres de $\hat{A}^T\hat{A}$. Un nombre $q$ de composantes (vecteurs propres $\hat{U}_q = U[1, N; 1, q]$) peut être sélectionné pour la réduction de dimension suivant la variance totale qu'on souhaite conserver. Ainsi, tout vecteur $\vec{d}$ est réduit à $q$ dimensions en le multipliant par $\hat{U}_q : \hat{\vec{d}}_q = \hat{U}_q\vec{d}$. La méthode la plus simple et populaire pour déterminer $q$ est la règle dite de "Kaiser" ou de "Kaiser-Guttman" [Guttman, 1954; Kaiser, 1960] même si elle tend à extraire beaucoup plus de composantes que nécessaire [Bandalos & Boehm-Kaufman, 2010]. Le critère de Kaiser juge que seules les valeurs propres supérieures ou égales à la moyenne des valeurs propres sont plus informatives que les variables initiales.

**L'allocation latente de Dirichlet** (ALD) [Blei *et al.* , 2003] est une technique qui extrait un nombre $q$ de thématiques à partir du corpus $\mathcal{D}$. Elle introduit les "variables latentes" comme pont pour modéliser les relations entre les textes et les termes. Chaque document est caractérisé comme une distribution de Dirichlet sur les variables latentes (thématiques), et chaque thématique est caractérisée par une autre distribution de Dirichlet sur tous les termes. La réduction de dimension est déduite de la première distribution en obtenant pour chaque document, un vecteur de taille $q$ représentant la distribution de probabilité des thématiques dans le document. Un des défis de la modélisation thématique est la détermination d'une va-

12. Distance entre les individus pris 2 à 2, ou dispersion autour du barycentre.

13. Centrer-réduire une variable $X$ d'espérance $\mu$ et d'écart-type $\sigma$ revient à obtenir une variable $\hat{X}$ d'espérance nulle et variance à 1 en calculant pour chaque valeur $X_i$ de $X$, $\hat{X}_i = \frac{X_i - \mu}{\sigma}$.

leur optimale du nombre de thèmes $q$. Parmi plusieurs valeurs candidates, celle qui maximise la cohérence du modèle peut être choisie. La cohérence du modèle est la moyenne des cohérences des thèmes. La cohérence de chaque thème peut être considérée comme étant la moyenne de la similarité entre les paires de termes de la description du thème [14]. Fang *et al.* [2016] démontrent que la cohérence est bien estimée avec la similarité cosinus des plongements lexicaux des termes.

**L'analyse latente sémantique** (ALS) [Dumais *et al.* , 1988; Deerwester *et al.* , 1990] réalise une décomposition en valeurs singulières (SVD) de la matrice $A$ des scores de co-occurrence document-terme. Plus précisément, l'ALS applique la décomposition SVD directement sur $A$, contrairement à l'ACP qui centre-réduit cette dernière au préalable. L'ALS permet ainsi de réduire la grande dimension lexicale des documents à un espace de thématiques de dimensions définies par le nombre de valeurs propres choisies. Comme pour l'ALD, le nombre $q$ de thématiques peut être sélectionné comme étant celui qui maximise la cohérence.

**La factorisation de matrice non-négative** (FMN) [Paatero & Tapper, 1994] factorise la matrice $A$ des scores non-négatifs de co-occurrence document-terme, en deux matrices $W$ et $H$ sans élément négatif, dont le produit est une approximation de $A$. Il s'agit en effet d'une méthode de modélisation de thématiques dans laquelle $W$ est décrite comme la matrice $N \times q$ de relation entre les termes et les thèmes découverts dans les documents, et $H$ est la matrice $q \times m$ des poids d'appartenance des documents aux thèmes. Le calcul de $W$ et $H$ consiste à minimiser la fonction objectif : $\frac{1}{2}\|A - WH\|_F^2 = \sum_{i=1}^{N}\sum_{i=1}^{m}(A_{ij} - (WH)_{ij})^2$, où $\|\cdot\|_F$ désigne la norme matricielle de Frobenius [15], et $A_{ij} = w(t_j, d_i)$, $d_i \in \mathcal{D}$. Comme pour l'ALD, le nombre $q$ de composantes (ou thématiques) peut être sélectionné comme étant celui qui maximise la cohérence.

**Barycentre des termes** A partir de vecteurs de mots appris à l'aide de méthodes comme Word2Vec [Mikolov *et al.* , 2013] ou GloVe [Pennington

14. Les $n$ premiers termes les plus fréquents du thème.

15. $\|X\|_F = \sqrt{\sum_{i=1}^{N}\sum_{i=1}^{m} X_{ij}^2}$.

*et al.* , 2014], il est souvent proposé de considérer le document comme le barycentre des mots qu'il contient, et de le représenter comme la moyenne des vecteurs de termes pondérés par leur poids dans le document (par exemple TF-IDF) [Le & Mikolov, 2014a; Charlet & Damnati, 2018; Arora *et al.* , 2017]. Tout vecteur $\vec{d}$ est réduit en un vecteur $\hat{\vec{d}}_q$ de $q$ dimensions correspondant à la taille des vecteurs de termes : $\hat{\vec{d}}_q[k] = \frac{1}{\sum_{i=1}^{m} \vec{d}[i]} \sum_{i=1}^{m} \vec{d}[i] * \vec{t}_i[k]$.

### 4.2.4 Sélection du nombre optimal de groupes

Au delà de l'algorithme à utiliser, le nombre $K$ approprié de clusters ne doit pas être prédéfini mais déterminé automatiquement, puisqu'il est difficile de savoir à l'avance le nombre de groupes. Une méthode très connue est celle du « coude » (ou « genou ») [Halkidi *et al.* , 2001], qui est basée sur le principe de base des algorithmes de partitionnement (e.g. K-moyennes) i.e. minimiser le critère d'inertie : $J(K) = \sum_{j=1}^{K} \sum_{d_i \in C_j} \left\| \vec{d}_i - \overrightarrow{\overline{d}_j} \right\|^2$, $C_j$ étant ensemble des objets du cluster $j$, et $\overrightarrow{\overline{d}_j}$ le centre du cluster $j$. La méthode du coude consiste à essayer différentes valeurs consécutives de $K$, puis à choisir celle qui correspond au coude de la courbe du critère d'inertie $J(K)$ c'est-à-dire le point à partir duquel la décroissance de la courbe commence à être très faible. Le choix de ce coude est visuel et peut être ambigu (plusieurs valeurs de $K$ sur le coude par exemple).

La méthode de la silhouette moyenne [Rousseeuw, 1987] est une alternative moins ambigüe qui consiste à choisir comme valeur optimale de $K$, celle qui maximise le critère de la largeur moyenne de la silhouette : $\overline{s_K}(C) = \frac{1}{K} \sum_{i=1}^{N} s(d)$. La largeur $s(d)$ de la silhouette est un indice qui compare la ressemblance d'un document $d$ aux autres membres de son cluster $C_t$ par rapport à sa ressemblance aux autres clusters $C_l, l \neq t : s(d) = \frac{b(d)-a(d)}{\max\{a(d),b(d)\}}$ où $a(d) = \frac{1}{|C_t|} \sum_{d' \in C_t} Dis(d,d')$, et $b(d) = \min_{l \neq t} \frac{1}{|C_l|} \sum_{d' \in C_t} Dis(d,d')$, pour $d \in C_l$. $K$ est optimal lorsque $\overline{s_K}(C)$ est maximale. Les valeurs de ce dernier varient entre -1 (pire valeur) et +1 (meilleure valeur). Les valeurs proches de zéro indiquent que les clusters se chevauchent en $x$, et il est

difficile de savoir à quel cluster $x$ doit être affecté. Une valeur négative indique que $x$ a été affecté à cluster inapproprié.

### 4.2.5 Validation de la catégorisation

La validation peut être supervisée ou non suivant l'emploie ou pas des affectations manuelles de documents à des groupes attendus.

#### 4.2.5.1 Métriques supervisées ou indices externes

Les métriques couramment utilisées mesurent la ressemblance entre deux catégorisations $X = \{X_1, X_2, ..., X_r\}$ et $Y = \{Y_1, Y_2, ..., Y_s\}$ :

- l'indice ajusté par chance de Rand (*ajusted Rand index* - ARI) [Hubert & Arabie, 1985] corrigent l'indice de Rand (RI) [Rand, 1971] pour obtenir une valeur très proche de 0 pour les catégorisations aléatoires et exactement 1 lorsque les clusters sont identiques aux classes attendues. En effet, l'indice de Rand prend ses valeurs en pratique dans l'intervalle $[0.5; 1]$, et par conséquent, considère une valeur de base très élevée. ARI est calculé à l'aide du tableau de contingence résumant les chevauchements que partagent $X$ et $Y$ (Tableau 4.1) par la formule :

$$ARI(X,Y) = \frac{\sum_{i,j}\binom{n_{ij}}{2} - \frac{\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}}{\binom{N}{2}}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \frac{\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}}{\binom{N}{2}}}$$

avec $n_{i,j} = |X_i \cap Y_j|$ et $\binom{n}{2} = \frac{n(n-1)}{2}$.

| | $Y_1$ | $Y_2$ | $\cdots$ | $Y_s$ | $\sum$ |
|---|---|---|---|---|---|
| $X_1$ | $n_{11}$ | $n_{12}$ | $\cdots$ | $n_{1s}$ | $a_1$ |
| $X_2$ | $n_{21}$ | $n_{22}$ | $\cdots$ | $n_{2s}$ | $a_2$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\ddots$ | $\cdots$ | $\cdots$ |
| $X_r$ | $n_{r1}$ | $n_{r2}$ | $\cdots$ | $n_{rs}$ | $a_r$ |
| $\sum$ | $b_1$ | $b_2$ | $\cdots$ | $b_s$ | |

Tableau 4.1 – Tableau de contingence des chevauchement entre les catégorisations $X = \{X_1, X_2, ..., X_r\}$ et $Y = \{Y_1, Y_2, ..., Y_s\}$

ARI a des valeurs dans $[-1; 1]$. Une valeur négative indique que la catégorisation obtenue s'accorde moins bien avec l'attendu qu'une catégorisation aléatoire.

- L'information mutuelle normalisée (NMI) [Kvalseth, 1987; Strehl *et al.* , 2000; Vinh *et al.* , 2010] normalise l'information mutuelle entre $X$ et $Y$ par une agrégation de leur entropie respective. Par exemple, l'incertitude symétrique [Kvalseth, 1987] est une variante qui utilise la moyenne comme fonction d'agrégation : $NMI(X,Y) = \frac{2 \cdot I(X,Y)}{H(Y)+H(Y)}$, avec $I(X,Y) = H(X) - H(X|Y) = \sum_{i=1}^{r} \sum_{j=1}^{s} \frac{n_{ij}}{N} \log_2 \frac{n_{ij}/N}{a_i b_j / N}$ et $H(X) = \sum_{X_i \in X} (-p(X_i) \log_2 p(X_i))$, $p(X_i) = \frac{a_i}{N}$ et $p(Y_j) = \frac{b_j}{N}$ ; $n_{ij}, a_i$ et $b_j$ provenant du tableau de contingence (Tableau 4.1). La meilleure catégorisation est celle qui a la plus grande valeur.
- Au regard de leur formulation, les métriques ARI et NMI sont plus focalisées sur la différence de volume entre les clusters de deux catégorisations. D'autres méthodes appelées « mesures de comptage de paires » (*pair counting measures*) mesurent la capacité du modèle à mettre deux documents similaires (de labels identiques dans les données annotées) dans le même groupe, et des documents dis-similaires (de labels différents dans les données annotées) dans des clusters différents. Parmi ces mesures, on retrouve par exemple, la précision, le rappel, et la $F_1$-mesure qui sont définies par les formules suivantes [Manning *et al.* , 2009a] :

$$P = \frac{TP}{TP + FP}, R = \frac{TP}{TP + FN}, F_1 = \frac{2 \times P \times R}{P + R}.$$

Ne pas mentionner plusieurs fois les formules de rappel, précision, f1. Mais uniquement les définitions de TP, FP, TN, FN.

Ces scores prennent leurs valeurs dans $[0; 1]$. Les métriques de base, qui servent à les calculer, sont :

- un vrai positif (TP) survient si le modèle place deux documents similaires dans le même cluster (groupe généré par le modèle) ;
- un faux négatif (FN) survient si deux documents similaires sont dans des clusters différents ;
- un vrai négatif (TN) survient si deux documents dissemblables se retrouve dans deux clusters différents ;
- un faux positif (FP) survient si deux documents dissemblables sont dans le même cluster.

#### 4.2.5.2 Métriques non-supervisées ou indices internes

La cohésion et la séparation des clusters sont les principaux indices internes. La cohésion mesure le degré de proximité entre objets d'un cluster à partir du carré de la somme des erreurs [16] dans les clusters : $WCSS(C) = \sum_{j=1}^{K} \sum_{d \in C_j} (Dis(d, z_j))^2$, où $C = \{C_1, C_1, \cdots, C_K\}$ est l'ensemble des clusters de la catégorisation, $z_j$ le centre de $C_j$, et $Dis(d, z_j)$ la distance (généralement euclidienne) entre un point $d$ et $z_j$. En général, une valeur faible de la cohésion indique que les clusters sont plus compacts, et donc de meilleur qualité. Tandis qu'une valeur élevée révèle une grande variabilité entre les objets à l'intérieur les clusters. La séparation quant à elle mesure l'éloignement de chaque cluster des autres à partir du carré de la somme des distances entre clusters : $BCSS(C) = \sum_{j=1}^{K} |C_j|(\overline{z} - z_j)$, $\overline{z}$ étant le centre de tous les documents. Une grande valeur de séparation indique que les clusters sont isolés les uns des autres. Par conséquent, elle doit être maximisée. Le coefficient de silhouette de Rousseeuw [1987] (cf. § 4.2.4) combine les idées de cohésion et séparation mais individuellement pour chaque document.

## 4.3 Apprentissage d'une distance basée sur la transformation de document

Nous définissons une métrique $Dis_{\mathcal{M}}$ qui est fonction des transformations permettant de passer d'un document $d$ à un autre $d'$ :

$$\begin{array}{rccl} Dis_{\mathcal{M}} & : & \mathcal{D} \times \mathcal{D} & \rightarrow & \mathbb{R} \\ & & d, d' & \mapsto & Dis_{\mathcal{M}}(d, d') = f(\mathcal{M}_{d,d'}). \end{array} \tag{4.3.1}$$

$\mathcal{D}$ est le corpus. $\mathcal{M}_{d,d'}$ est l'ensemble des modifications de $d$ permettant d'obtenir $d'$ i.e. les paires de mots différents $(d[k], d'[k])$ telles que le mot $d[k]$ a été remplacé par $d'[k]$. $f$ est une fonction qui croît avec le nombre de modifications. Après une légère modification, le sens d'un texte reste assez similaire à celui de l'original. Tandis qu'après un grand nombre de modifications, le sens du texte est très différent de l'original.

16. Erreur : distance entre un point et le centre du cluster dont il est membre.

Pour des documents de même taille, cette distance peut, par exemple, se formuler comme étant la proportion de mots modifiés :

$$Dis_{\mathcal{M}}(d, d') = f(\mathcal{M}_{(d,d')}) = \frac{|\mathcal{M}_{(d,d')}|}{|d|} \tag{4.3.2}$$

Par contre, pour des textes de tailles différentes, il est impossible de savoir les positions où des mots ont été supprimés ou ajoutés, et par conséquent, il devient impossible de calculer leur distance. La distance étant une valeur continue, en entraînant un modèle de régression sur un ensemble de paires de documents pour lesquelles la distance est connue, il est possible de la prédire pour des paires de documents de taille quelconque. Nous proposons de générer une base synthétique de paires de documents dont l'un est un document du corpus original mais l'autre est le résultat de substitutions et suppressions de mots du premier. En contrôlant ces modifications, il est facile de calculer une valeur de $Dis_{\mathcal{M}}$ pour chaque paire générée de documents, même s'ils sont de tailles différentes (en considérant la suppression d'un mot comme son remplacement par le « mot vide »).

### 4.3.1 Génération d'une base d'apprentissage

La génération de la base synthétique nécessite de définir une formulation de la fonction $f(\mathcal{M}_{d,d'})$ pour les documents de taille égale, comme par exemple celle de l'Equation 4.3.2. Cette formulation considère que toutes les modifications ont la même importance c'est-à-dire qu'une substitution de mots contraires est équivalente à une substitution de mots similaires. Pour corriger cette limite, chaque modification peut être pondérée par la distance entre les mots substitués (le vecteur du « mot vide » étant nul) :

$$Dis_{\mathcal{M}}(d, d') = f(\mathcal{M}_{(d,d')}) = \frac{\sum_{(d[k],d'[k])\in\mathcal{M}_{(d,d')}} Dis_{cos}(\overrightarrow{d[k]}, \overrightarrow{d'[k]})}{|d|} \tag{4.3.3}$$

$d$ est un document du corpus original $\mathcal{D}$, et $d'$ est le résultat d'une transformation contrôlée de $d$. $\overrightarrow{d[k]}$ désigne le plongement lexical du mot $d[k]$. Pour garantir la symétrie et la réflexivité de la métrique, nous imposons respectivement $Dis_{\mathcal{M}}(d, d') = Dis_{\mathcal{M}}(d', d)$ et $Dis_{\mathcal{M}}(d, d) = Dis_{\mathcal{M}}(d', d') = 0, \forall d \in \mathcal{D}$ sur le jeu d'entraînement généré. L'algorithme 6 de génération

de documents synthétiques contrôle le taux de modifications à effectuer sur le document original grâce à un seuil donnée $0 \leq p \leq 1$. En variant $p$, plusieurs documents $d'$ sont générés pour chaque $d \in \mathcal{D}$ pour former un jeu d'apprentissage $B_{\mathcal{M}} = \{((d_1, d_2)*, Dis(d_1, d_2))_i\}_{1 \leq i \leq |B_{\mathcal{M}}|}$.

**Algorithme 6 :** Transformation de document

**Données :** document $d \in \mathcal{D}$, valeur seuil $p$, ensemble $W$ des mots
**Résultat :** $d'$, $\mathcal{M}_{(d,d')}$

$d' = []$;
$\mathcal{M}_{(d,d')} = \emptyset$;
**pour** $k \in [1 \,..\, |d|]$ **faire**
  $v = valeur\_alatoire\_entre(0, 1)$;
  **si** $v < p$ **alors**
    $d'[k] = modifie\_mot(d[k], W)$ ; // mot aléatoire de $W$ différent de $d[k]$;
    $\mathcal{M}_{d[k],d'[k]} = \mathcal{M}_{(d,d')} \cup \{(d[k], d'[k])\}$;
  **sinon**
    $d'[k] = d[k]$;
**retourner** $d', \mathcal{M}_{(d,d')}$;

### 4.3.2 Entraînement de la métrique

Sur $B_{\mathcal{M}}$, un modèle de régression peut être entrainé pour prédire la distance entre deux documents quelconques $d_i$ et $d_j$ en fonction de leur représentation vectorielle. Ce modèle de régression $Reg_{\mathcal{M}}$ peut être utilisé comme distance dans un algorithme de catégorisation comme celui des K-moyennes. Cependant, les modèles de régression ne supportent généralement qu'un seul vecteur en entrée, et pas deux comme en dispose la base $B_{\mathcal{M}}$. Les vecteurs $\vec{d_i}$ et $\vec{d_j}$ doivent donc être agrégés en un seul. Une bonne agrégation est la soustraction car l'agrégation de documents similaires est un vecteur proche du nul. La fonction d'estimation automatique de la distance entre $x$ et $y$ s'écrit : $Dis_{\mathcal{M}}(d_i, d_j) = Reg_{\mathcal{M}}(\vec{d_i} - \vec{d_j})$.

### 4.3.3 Utilisation pour le regroupement des documents

Nous proposons dans un premier temps de sélectionner la représentation vectorielle pour laquelle la distance sépare au mieux les groupes manuellement annotées et rapproche au mieux les éléments à l'intérieur de

ces groupes. La représentation [17] optimale maximise la largeur moyenne de la silhouette de la catégorisation manuelle. L'idée étant d'avoir une représentation qui optimise à la fois les métriques supervisées et non supervisées de validation. La représentation ainsi déterminée est utilisée ensuite pour la catégorisation sur les corpus non annotés.

## 4.4 Expérimentations et résultats

Cette section discute de la validité, l'adéquation, et l'efficacité de la métrique apprise en comparaison avec d'autres distances. La validité de la métrique est établie si cette dernière respecte les propriétés d'une distance. L'adéquation de la métrique avec le problème à résoudre mesure la capacité de la métrique à estimer une distance très faible entre documents de mêmes circonstances factuelles, et une similarité très faible entre documents de circonstances différentes, indépendamment de tout algorithme de catégorisation. Enfin, l'efficacité de la métrique est liée à la qualité de la catégorisation résultant de l'application d'un algorithme de regroupement utilisant cette distance.

### 4.4.1 Données

Pour l'évaluation supervisée, nous disposons d'une base annotée sur la catégorie de demande "dommage-intérêts / action en responsabilité civile professionnelle contre les avocats" (*arcpa*) qui concerne les contentieux impliquant des avocats. L'annotateur (juriste) a organisé 81 documents [18] en 4 circonstances factuelles, avec 6 documents appartenant simultanément chacun à 2 groupes (chevauchements) :

- cas *a* (46 documents) : il s'agit d'un avocat qui est négligent et envoie son assignation de manière tardive ;
- cas *b* (20 documents) : il s'agit d'un avocat qui n'a pas donné un conseil opportun, qui n'a pas soulevé le bon argument ;
- cas *c* (18 documents) : un avocat qui n'a pas rédigé un acte valide ou réussi à obtenir un avantage fiscal ;

17. Combinaison modèle vectoriel et méthode de réduction.
18. Sur 85 documents disponibles de catégorie *arcpa*.

- cas *d* (3 documents) : il s'agit d'un avocat attaqué par son adversaire et non par son propre client.

Les terminologies des cas se distinguent bien (Tableau 4.2). Par conséquent, une représentation des documents qui mettrait en évidence de tels termes permettrait de retrouver les circonstances factuelles.

| Corpus | Terminologie |
|---|---|
| *arcpa* | chance, perte chance, avocat, perte, diligence, chance obtenir, perdre, client, devoir conseil, manquement |
| *cas a* | chance, perte chance, chance succès, perte, client, préjudice indemnisable, article code commerce, indemnisable, condamnation emporter, emporter nécessairement rejet |
| *cas b* | défense intérêt, intérêt client, avocat, contractuel égard, responsabilité contractuel droit, responsabilité professionnel avocat, contractuel droit commun, assurer défense intérêt, civil avocat, grief articuler |
| *cas c* | rédacteur acte, rédacteur, avocat rédacteur acte, avocat rédacteur, qualité rédacteur acte, rédaction acte, qualité rédacteur, projet acte, prendre initiative conseiller, initiative conseiller |
| *cas d* | revêtir aucun, revêtir aucun caractère, article code, article code procédure, faire référence aucun, fautif madame, civil profit autre, civil depuis, mention expresse, moyen dont |

*10 premiers termes lémmatisés de 1 à 3 mots sélectionnés à l'aide du coefficient de corrélation ngl (cf. § 2.2.3.2)*

Tableau 4.2 – Terminologies de la catégorie *arcpa* et de ses circonstances factuelles manuellement annotées.

Pour l'évaluation non supervisée, les corpus des chapitres 2 et 3 sont aussi employés. Ils sont notés $\mathcal{D}_{acpa}$, $\mathcal{D}_{concdel}$, $\mathcal{D}_{danais}$,$\mathcal{D}_{dcppc}$, $\mathcal{D}_{doris}$, $\mathcal{D}_{styx}$.

### 4.4.2 Protocole et outils logiciels

La métrique apprise est entraînée sur une base générée, puis nous l'évaluons sur le corpus annoté $\mathcal{D}_{arcpa}$ restreint aux 74 documents n'appartenant qu'à l'un des cas $L = \{a, b, c\}$ (les chevauchements ne sont pas traités). Les documents sont pré-traités avant leur représentation vectorielle. Ce pré-traitement consiste à les sectionner (chapitre 1), à les restreindre à la section Motifs, à mettre en minuscule et lemmatiser leur contenu, puis à y éliminer la ponctuation et des mots inutiles (*stop words*) car ils sont généralement indépendants de toute catégorie. Après pré-traitement, les documents ont une taille allant de 208 à 3812 mots dont une moyenne de 1381 mots. Pour générer les données d'entrainement de $Dis_M$, le vocabulaire $W$ utilisé est restreint aux mots du corpus original $D$ sur lequel il faut appliquer les catégorisations. La représentation vectorielle emploie des modèles de type TF-IDF (poids local $\times$ poids global $\times$ facteur de normalisation, cf. § 2.3.1) avec des n-grammes de 1 à 3 mots. Les poids globaux sont

appris sur la discrimination entre deux corpus $\mathcal{D}_c$ et $\mathcal{D}_{\overline{c}}$ ($|\mathcal{D}_{\overline{arcpa}}| = 427$ documents) c'est-à-dire la terminologie d'une catégorie $c$ de demandes.

La librairie Python Scikit-Learn [Pedregosa *et al.* , 2011] a été utilisée pour les implémentations des algorithmes de réduction de dimension, et ceux du DBSCAN, de la catégorisation spectrale (*Spectral Clustering*), et du regroupement hiérarchique (*Agglomerative Clustering*). Les implémentations des C-moyennes floues probabilistes (*Probabilistic FCM*) et des arbres aléatoires (*Random Forest*) proviennent respectivement des librairies *scikit-cmeans 0.1* [19] et RandomForestClustering [20]. La distance WMD est celle implémentée dans la librairie Gensim de Řehůřek & Sojka [2010]. Les expérimentations utilisent un modèle de plongement lexical de type GLoVE [Pennington *et al.* , 2014] entraîné sur un corpus de +800k décisions de justice lémmatisées avec des fenêtres de contexte de 15 mots. Ce modèle dispose de 32445 mots représentés par des vecteurs de 300 dimensions. Le vecteur $\vec{t}$ d'un terme $t$ de $n$ mots $(w_1, w_2, \cdots, w_n)$ est obtenu en concaténant leur vecteur respectif de la droite vers la gauche : $\vec{t} = [\vec{w_1}\vec{w_2}\cdots\vec{w_n}]$.

### 4.4.3 Validité de la distance apprise

Pour la catégorie *arcpa*, la base d'entraînement $B_{\mathcal{M}}$ comprend 935 documents dont 10 documents synthétiques générés pour chacun des 85 documents. Sur un modèle TF-IDF, la régression linéaire approxime bien la distance proposée $Dis_{\mathcal{M}}$ car elle a un faible taux d'erreur (Figure 4.1a).

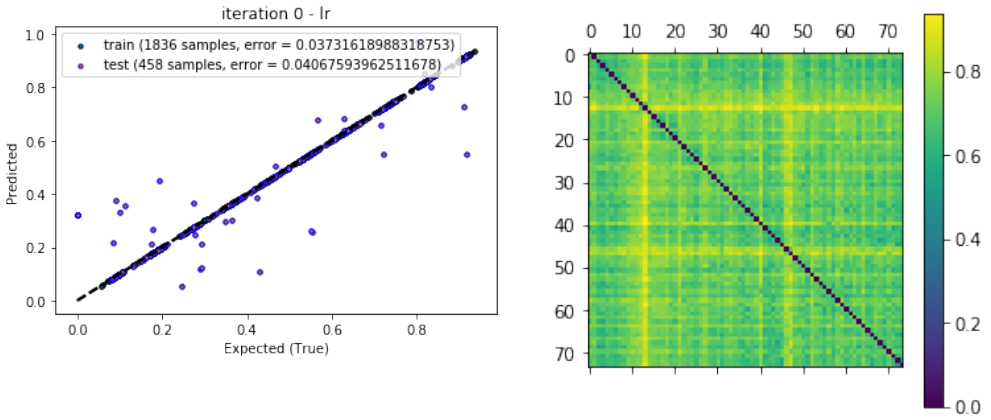
D'après la matrice des distances entre les 74 documents de $\mathcal{D}_{\text{arcpa}}$ (Figure 4.1b), la "non-négativité", l'identité discernable, et la symétrie sont respectée car toutes les valeurs sont non-négatives, seule la diagonale est nulle, et la matrice est symétrique. De plus, toutes les distances sont comprises entre 0 et 1, et par conséquent la métrique est normalisée.

L'inégalité triangulaire est vérifiée car $Dis_{\mathcal{M}}(d, d'') - (Dis_{\mathcal{M}}(d, d') + Dis_{\mathcal{M}}(d', d'')) \leq 0, \forall (d, d', d'') \in \mathcal{D}_{arcpa} \times \mathcal{D}_{arcpa} \times \mathcal{D}_{arcpa}$ (Figure 4.2).

### 4.4.4 Sélection de la représentation optimale des textes

En considérant la catégorisation manuelle de $\mathcal{D}_{arcpa}$, différentes représentation vectorielles peuvent être comparées, à l'aide de la silhouette, sur

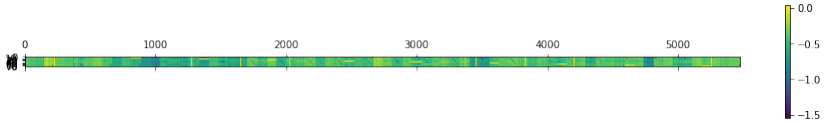
19. https://bm424.github.io/scikit-cmeans/index.html
20. https://github.com/joshloyal/RandomForestClustering/




(a) Différence entre valeurs prédites et valeurs attendues

(b) Matrice des distances entre documents

Figure 4.1 – Validité de la distance apprise.




$Dis_{\mathcal{M}}(d,d'') - (Dis_{\mathcal{M}}(d,d') + Dis_{\mathcal{M}}(d',d'')), \forall (d,d',d'') \in \mathcal{D}_{arcpa} \times \mathcal{D}_{arcpa} \times \mathcal{D}_{arcpa}$, avec les $d$ indicés en lignes et les paires $(d',d'')$ en colonnes.

Figure 4.2 – Matrice de vérification de l'inégalité triangulaire

leur habilité à séparer les clusters manuels de documents dans l'espace. Nous comparons ici les combinaisons de différents poids locaux (Tableau 2.4), poids globaux (cf. § 2.2.3) et méthodes de réduction de dimensions (cf. § 4.2.3.2). Le Tableau 4.3 présente la représentation de largeur moyenne optimale de silhouette sur les données annotées pour chaque distance. Pour les réductions par ALD, ALS, et FNM, le nombre de thèmes est déterminé entre 4 et 10.

Le modèle vectoriel TP-NGL réduit à 4 dimensions par la factorisation de matrice non négative (FNM) est préférée par la majorité des distances. Nous remarquons que la pondération globale supervisée (NGL et CHI2) met en évidence non seulement la terminologie de la catégorie de demande mais aussi celle des circonstances factuelles associées. La FNM marche mieux en moyenne pour toutes les classes avec des silhouettes maximales comprises entre 0.052 et 0.212, suivie de l'ALS (0.048 – 0.119) et de l'ACP (0.029 – 0.079). Les scores maximaux de silhouette observés par

| Distance | Base[a] | Silhouette optimale (pondération, réduction, dim.) |
|---|---|---|
| $Dis_{jaccard}$ | 0.001 | 0.212 (TP-NGL, FNM, 4) |
| $Dis_{cos}$ | 0.002 | 0.202 (TP-NGL, FNM, 4) |
| $Dis_{M}$ | -0.049 | 0.195 (TP-NGL, FNM, 4) |
| $Dis_{braycurtis}$ | 0.002 | 0.182 (TP-NGL, FNM, 4) |
| $Dis_{euclidienne}$ | 0.001 | 0.168 (TP-NGL, FNM, 4) |
| $Dis_{manhattan}$ | -0.019 | 0.17 (TP-NGL, FNM, 4) |
| $Dis_{pearson}$ | 0.014 | 0.057 (TP-CHI2, aucun, 19763) |
| $Dis_{wmd}$ | -0.096 | - |

[a] occurrence de mots pour $Dis_{wmd}$, et TF-IDF pour les autres

Tableau 4.3 – Meilleures représentations sur la catégorisation manuelle.

l'ALD (-0.032 – -0.001) sont moins bons que la représentation sans réduction (0.001 – 0.008). La réduction par la méthode du barycentre des termes quant à elle donne un score de silhouette compris entre -0.048 et 0.013 au maximum sur l'ensemble des distances, avec une moyenne de 0.002. Les représentations sélectionnées sont utilisées dans la suite.

Le Tableau 4.3 classe aussi les distances en fonction de leur adaptabilité à la tâche, et la $Dis_{\mathcal{M}}$ se replace bien grâce à la représentation optimale. $Dis_{wmd}$ n'a pas été adaptée avec un modèle vectoriel plus adéquat que le sac-de-mots. Par conséquent, elle présente un score légèrement moins bon que celui d'un regroupement aléatoire (négatif et proche de 0).

### 4.4.5 Catégorisation dans le corpus annoté manuellement

#### 4.4.5.1 Nombre prédéfini de *clusters*

Le Tableau 4.4 présente les mesures ARI, NMI et $F_1$-mesure de la catégorisation par K-moyennes et K-medoïdes sur $\mathcal{D}_{arcpa}$, avec $K = 3$.

Les valeurs de silhouette se reflètent plus sur les indices ARI et NMI, mais moins sur la $F_1$-mesure. Suivant le score $F_1$, $Dis_M$, semble le mieux adaptée pour les K-moyennes, et $Dis_{cos}$ pour les K-medoïdes. Mais les distances, $Dis_{jaccard}$, $Dis_{cos}$, $Dis_{euclidienne}$ semblent mieux faire le compromis entre les différents critères de validation.

#### 4.4.5.2 Nombre de *clusters* déterminé automatiquement

Nous analysons ici la détermination du nombre de clusters par la méthode de la silhouette pour chaque distance (Tableau 4.5). La sélection de $K$

| Distance | Algorithme | Silhouette | ARI | NMI | R | P | $F_1$ |
|---|---|---|---|---|---|---|---|
| $Dis_{\mathcal{M}}$ | K-moyennes | 0.403 | 0.411 | 0.427 | 0.574 | 0.648 | **0.607** |
| $Dis_{\mathcal{M}}$ | K-medoïdes | 0.398 | 0.321 | 0.340 | 0.483 | 0.591 | 0.532 |
| $Dis_{braycurtis}$ | K-moyennes | 0.370 | 0.364 | 0.382 | 0.545 | 0.603 | 0.570 |
| $Dis_{braycurtis}$ | K-medoïdes | 0.358 | 0.272 | 0.292 | 0.444 | 0.540 | 0.487 |
| $Dis_{cosine}$ | K-moyennes | 0.422 | 0.389 | 0.406 | 0.556 | 0.616 | 0.583 |
| $Dis_{cosine}$ | K-medoïdes | **0.448** | 0.437 | **0.455** | 0.656 | 0.598 | **0.626** |
| $Dis_{euclidean}$ | K-moyennes | 0.372 | **0.417** | **0.434** | 0.591 | 0.603 | 0.592 |
| $Dis_{euclidean}$ | K-medoïdes | 0.369 | 0.392 | 0.409 | 0.566 | 0.672 | 0.615 |
| $Dis_{jaccard}$ | K-moyennes | **0.442** | 0.371 | 0.389 | 0.554 | 0.600 | 0.574 |
| $Dis_{jaccard}$ | K-medoïdes | 0.431 | **0.440** | **0.455** | 0.529 | 0.645 | 0.581 |
| $Dis_{manhattan}$ | K-moyennes | 0.390 | 0.376 | 0.394 | 0.567 | 0.582 | 0.571 |
| $Dis_{manhattan}$ | K-medoïdes | -0.059 | 0.097 | 0.127 | 0.479 | 0.422 | 0.448 |
| $Dis_{pearson}$ | K-moyennes | 0.434 | 0.088 | 0.117 | 0.585 | 0.487 | 0.530 |
| $Dis_{pearson}$ | K-medoïdes | -0.019 | 0.111 | 0.136 | 0.421 | 0.476 | 0.447 |
| dis_wmd | K-medoïdes | 0.105 | -0.004 | 0.024 | 0.333 | 0.401 | 0.364 |

Tableau 4.4 – Evaluation de la catégorisation par K-moyennes et K-medoïdes sur $\mathcal{D}_{arcpa}$ avec le nombre de groupes prédéfini à $K = 3$.

est effectué pour les valeurs entre 2 et 30. $Dis_{\mathcal{M}}$ retrouve le nombre attendu avec les K-moyennes. 4 est la plus choisie par les distances, et 4 donne une $F_1$-mesure au maximum à 0.551 et un ARI de 0.398 avec $Dis_{cos}$. Les valeurs déterminées (entre 2 et 6) sont très proches de la valeur attendue 3 pour toutes les distances. Notons aussi que l'efficacité de $Dis_{\mathcal{M}}$ et $Dis_{cos}$ reste presque aussi élevée qu'avec un $K$ prédéfini (Tableau 4.4). $Dis_{\mathcal{M}}$ et $Dis_{cos}$ semblent ainsi former de bonnes combinaison avec respectivement les K-moyennes et les K-medoïdes. Par ailleurs, seules $Dis_{manhattan}$ et $Dis_{wmd}$ obtiennent de très faibles valeurs pour les indices ARI et NMI. Même si ces valeurs sont inférieures pour les autres distances, ces dernières, en particulier $Dis_{\mathcal{M}}$, $Dis_{jaccard}$ et $Dis_{cos}$, parviennent quand même à un bon compromis entre les 3 critères ARI, NMI et $F_1$. $Dis_{jaccard}$ et $Dis_{cos}$ sont efficaces à la fois avec les K-moyennes et les K-medoïdes. Elles parviennent à associer une bonne validation non-supervisée (silhouette) à une bonne validation non-supervisée.

La Figure 4.3 montre l'évolution de la valeur de la silhouette en fonction du nombre de clusters pour $Dis_{\mathcal{M}}$ utilisé dans les K-moyennes.

| Distance | Algorithme | K | Silhouette | ARI | NMI | R | P | $F_1$ |
|---|---|---|---|---|---|---|---|---|
| $Dis_{\mathcal{M}}$ | K-moyennes | **3** | 0.438 | **0.407** | **0.423** | 0.552 | 0.654 | **0.599** |
| $Dis_{\mathcal{M}}$ | K-medoïdes | 6 | 0.453 | 0.359 | 0.395 | 0.298 | 0.669 | 0.413 |
| $Dis_{braycurtis}$ | K-moyennes | 4 | 0.473 | 0.383 | 0.407 | 0.446 | 0.658 | 0.532 |
| $Dis_{braycurtis}$ | K-medoïdes | 5 | 0.448 | 0.344 | 0.375 | 0.331 | 0.645 | 0.437 |
| $Dis_{cosine}$ | K-moyennes | 4 | 0.528 | 0.383 | 0.407 | 0.446 | 0.658 | 0.532 |
| $Dis_{cosine}$ | K-medoïdes | 4 | 0.526 | **0.398** | **0.421** | 0.464 | 0.680 | **0.551** |
| $Dis_{euclidean}$ | K-moyennes | 5 | 0.478 | 0.365 | 0.395 | 0.341 | 0.670 | 0.452 |
| $Dis_{euclidean}$ | K-medoïdes | 5 | 0.456 | 0.313 | 0.346 | 0.335 | 0.619 | 0.434 |
| $Dis_{jaccard}$ | K-moyennes | 4 | 0.570 | 0.367 | 0.391 | 0.439 | 0.643 | 0.522 |
| $Dis_{jaccard}$ | K-medoïdes | 4 | **0.560** | 0.389 | 0.412 | 0.451 | 0.666 | 0.538 |
| $Dis_{manhattan}$ | K-moyennes | 4 | 0.482 | 0.376 | 0.400 | 0.452 | 0.657 | 0.535 |
| $Dis_{manhattan}$ | K-medoïdes | 5 | 0.452 | 0.368 | 0.397 | 0.345 | 0.675 | 0.456 |
| $Dis_{pearson}$ | K-moyennes | 2 | **0.611** | 0.054 | 0.072 | 0.746 | 0.453 | 0.564 |
| $Dis_{pearson}$ | K-medoïdes | 2 | 0.171 | 0.152 | 0.166 | 0.598 | 0.482 | 0.534 |
| $Dis_{wmd}$ | K-medoïdes | 2 | 0.332 | -0.016 | 0.002 | 0.545 | 0.397 | 0.459 |

Tableau 4.5 – Evaluation de la catégorisation par K-moyennes et K-medoïdes sur $\mathcal{D}_{arcpa}$ avec détermination du nombre de clusters basée sur la silhouette.

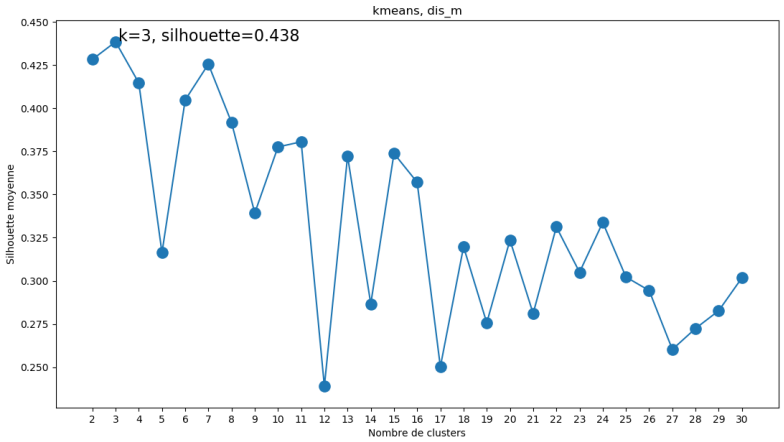



Figure 4.3 – Evolution de la silhouette pour les K-moyennes et la distance apprise.

#### 4.4.5.3 Autres algorithmes de catégorisation

Avec la représentation sélectionnée TP-NGL, nous appliquons d'autres algorithmes de catégorisation sur $\mathcal{D}_{arcpa}$ (Tableau 4.6). L'algorithme de regroupement à chevauchement des C-moyennes floues probabilistes (*Pro-*

*babilistic FCM*) est utilisé pour partitionner le corpus en choisissant un seul cluster pour chaque document (celui pour qui le degré d'appartenance est maximal). Seuls le regroupement hiérarchique et les C-moyennes floues parviennent à trouver un nombre de clusters (4) assez proche de celui attendu (3). Les forêts aléatoires ont une très basse $F_1$-mesure du fait du grand nombre de clusters déterminé. Par ailleurs, la tâche ne semble pas correspondre ni à la catégorisation par densité qui tend toujours à mettre tous les documents dans un même cluster, ni à la catégorisation spectrale qui sélectionne un très grand nombre de clusters.

| **Algorithme** | **K** | **Silhouette** | **ARI** | **NMI** | **R** | **P** | **$F_1$** |
|---|---|---|---|---|---|---|---|
| Spectral Clustering | 19 | 0.352 | 0.193 | 0.317 | 0.069 | 0.632 | 0.124 |
| DBSCAN | 2 | -1.000 | 0.000 | 0.000 | 1.000 | 0.398 | 0.570 |
| Agglomerative Clustering | 4 | 0.475 | 0.355 | 0.381 | 0.428 | 0.567 | 0.487 |
| Probabilistic FCM | 4 | 0.521 | 0.394 | 0.417 | 0.444 | 0.657 | 0.530 |
| Random Forest | 11 | 0.272 | 0.228 | 0.303 | 0.127 | 0.598 | 0.210 |

Tableau 4.6 – Evaluation de la catégorisation proposée par plusieurs algorithmes sur $\mathcal{D}_{arcpa}$ avec détermination du nombre de clusters avec la silhouette.

Le score de silhouette des C-moyennes floues probabilistes (Figure 4.4) a une décroissance plus rapide et plus monotone que notre implémentation des K-moyennes (Figure 4.3).

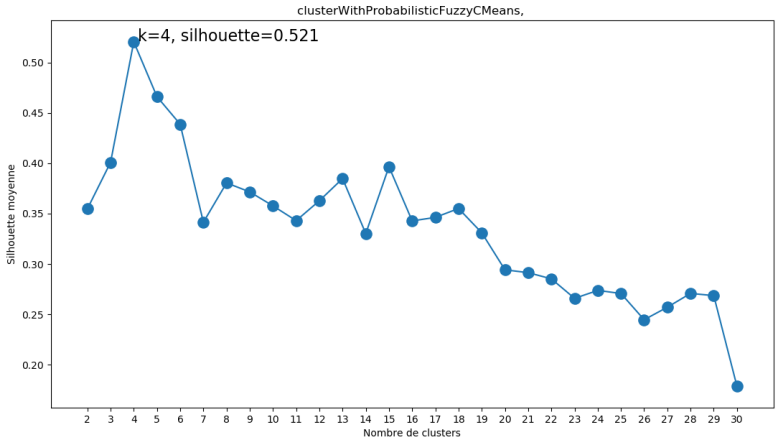



Figure 4.4 – Évolution de la silhouette pour les C-moyennes floues probabilistes

### 4.4.6 Catégorisation des corpus non annotés manuellement

Les matrices documents-termes des autres catégories sont construites à base la représentation (TP-NGL, FMN, 4 thématiques) sélectionnée sur la catégorisation manuelle (cf. § 4.4.4). Le tableau 4.7 présente les valeurs de silhouette ($\bar{s}_K$) obtenues par les K-moyennes et le K-medoïdes sur ces matrices. Les nombres déterminés de clusters restent bas (entre 2 et 6) bien qu'ils soient sélectionnés entre 2 et 30.

| $\mathcal{D}$ | **Distance** | **Algorithme** | **K** | $\bar{s}_K$ |
|---|---|---|---|---|
| $\mathcal{D}_{acpa}$ (21*) | $Dis_{\mathcal{M}}$ | K-medoïdes | 2 | 0.769 |
| | $Dis_{\mathcal{M}}$ | K-moyennes | 2 | 0.769 |
| | $Dis_{cosine}$ | K-medoïdes | 5 | 0.694 |
| | $Dis_{cosine}$ | K-moyennes | 3 | 0.762 |
| | $Dis_{jaccard}$ | K-medoïdes | 2 | 0.499 |
| | $Dis_{jaccard}$ | K-moyennes | 3 | 0.769 |
| $\mathcal{D}_{concdel}$ (30) | $Dis_{\mathcal{M}}$ | K-medoïdes | 4 | 0.639 |
| | $Dis_{\mathcal{M}}$ | K-moyennes | 4 | 0.624 |
| | $Dis_{cosine}$ | K-medoïdes | 4 | 0.675 |
| | $Dis_{cosine}$ | K-moyennes | 4 | 0.632 |
| | $Dis_{jaccard}$ | K-medoïdes | 5 | 0.719 |
| | $Dis_{jaccard}$ | K-moyennes | 5 | 0.702 |
| $\mathcal{D}_{danais}$ (198) | $Dis_{\mathcal{M}}$ | K-medoïdes | 4 | 0.403 |
| | $Dis_{\mathcal{M}}$ | K-moyennes | 2 | 0.442 |
| | $Dis_{cosine}$ | K-medoïdes | 4 | 0.475 |
| | $Dis_{cosine}$ | K-moyennes | 4 | 0.471 |
| | $Dis_{jaccard}$ | K-medoïdes | 3 | 0.483 |
| | $Dis_{jaccard}$ | K-moyennes | 3 | 0.482 |

| $\mathcal{D}$ | **Distance** | **Algorithme** | **K** | $\bar{s}_K$ |
|---|---|---|---|---|
| $\mathcal{D}_{dcppc}$ (91) | $Dis_{\mathcal{M}}$ | K-medoïdes | 2 | 0.451 |
| | $Dis_{\mathcal{M}}$ | K-moyennes | 2 | 0.781 |
| | $Dis_{cosine}$ | K-medoïdes | 2 | 0.549 |
| | $Dis_{cosine}$ | K-moyennes | 2 | 0.925 |
| | $Dis_{jaccard}$ | K-medoïdes | 2 | -0.016 |
| | $Dis_{jaccard}$ | K-moyennes | 3 | 0.820 |
| $\mathcal{D}_{doris}$ (59) | $Dis_{\mathcal{M}}$ | K-medoïdes | 2 | 0.509 |
| | $Dis_{\mathcal{M}}$ | K-moyennes | 3 | 0.527 |
| | $Dis_{cosine}$ | K-medoïdes | 5 | 0.549 |
| | $Dis_{cosine}$ | K-moyennes | 4 | 0.586 |
| | $Dis_{jaccard}$ | K-medoïdes | 3 | 0.600 |
| | $Dis_{jaccard}$ | K-moyennes | 4 | 0.645 |
| $\mathcal{D}_{styx}$ (50) | $Dis_{\mathcal{M}}$ | K-medoïdes | 2 | 0.669 |
| | $Dis_{\mathcal{M}}$ | K-moyennes | 2 | 0.669 |
| | $Dis_{cosine}$ | K-medoïdes | 5 | 0.695 |
| | $Dis_{cosine}$ | K-moyennes | 4 | 0.705 |
| | $Dis_{jaccard}$ | K-medoïdes | 6 | 0.635 |
| | $Dis_{jaccard}$ | K-moyennes | 4 | 0.690 |

* Nombre de documents
$\bar{s}_K$ : largeur moyenne de la silhouette

Tableau 4.7 – Evaluation non-supervisée des K-moyennes et K-medoïdes sur $\mathcal{D}_{acpa}, \mathcal{D}_{concdel}, \mathcal{D}_{danais}, \mathcal{D}_{dcppc}, \mathcal{D}_{doris}, \mathcal{D}_{styx}$.

La silhouette étant bonne en général, on s'attend à ce que les groupes soient en majorité formés effectivement de décisions partageant les mêmes circonstances factuelles. Il est possible de se faire une idée des circonstances factuelles découvertes en observant leur terminologie qu'on peut extraire à l'aide d'une pondération globale supervisée comme le coefficient NGL. Considérons par exemple le regroupement avec les K-medoïdes combinés à la distance cosinus pour laquelle des résultats intéressants ont été obtenus sur $\mathcal{D}_{arcpa}$. Pour la catégorie CONCDEL [21], on obtient 4 circonstances factuelles dont les champs lexicaux (Tableau 4.8) semblent définir les cas de contrefaçon pour le cluster 0, de publicité déloyale pour le

21. CONDEL : dommages-intérêts pour concurrence déloyale.

cluster 1, de recrutement d'un salarié par un concurrent pour le cluster 2, et de contrats avec des partenaires pour le dernier cluster.

| Cluster | Terminologie |
|---|---|
| *0* | distinctif, reproduire, code propriété intellectuel, contrefaçon, attaque, dépôt marque, circonstance intervenir, intervenir jugement, intervenir jugement déférer, caractère distinctif |
| *1* | publicitaire, action concurrence, défaut qualité agir, action concurrence déloyal, qualité agir, fichier client, force chose juger, fonder demande titre, date transfert, celui -ci fonder |
| *2* | acte concurrence, acte concurrence déloyal, clause non concurrence, non concurrence, clause non, entreprise concurrent, démarchage, démarcher, salarié, massif |
| *3* | non concurrencer, clause non concurrencer, tout droit, résilier contrat, marcher, préjudice invoquer, détournement, compte entre partie, contrat @card@ juillet, compte entre |

*10 premiers termes de 1 à 3 mots sélectionnés à l'aide du coefficient de corrélation ngl (cf. § 2.2.3.2)*

Tableau 4.8 – Terminologies des circonstances factuelles découvertes en combinant les K-medoïdes et la distance cosinus sur $\mathcal{D}_{concdel}$.

Pour la catégorie DORIS [22], on obtient 5 circonstances factuelles dont les champs lexicaux (Tableau 4.9) semblent définir les cas d'excès de trouble pour le cluster 0, de différents entre copropriétaires d'un immeuble pour le cluster 1, de toit dépassant la limite entre deux habitations pour le cluster 2, et d'ouvrage dépassant la hauteur limite autorisée pour le dernier cluster.

| Cluster | Terminologie |
|---|---|
| *0* | excéder inconvénient, inconvénient normal, excéder inconvénient normal, normal voisinage, inconvénient normal voisinage, inconvénient, trouble excéder inconvénient, trouble excéder, excéder, normal |
| *1* | copropriétaire, syndicat copropriétaire, syndicat, condamner in, anormal voisinage, trouble anormal voisinage, in, trouble anormal, syndic, jouissance subir |
| *2* | deux fond \| fonds, séparatif deux fond \| fonds, limite séparatif deux, ordonner démolition, séparatif deux, implanter, condamner démolir, devoir établir toit, devoir établir, toit manière |
| *3* | manière plus, chose manière plus, chose manière, usage prohiber loi, prohiber loi règlement, prohiber loi, absolu, usage prohiber, manière plus absolu, plus absolu, |
| *4* | situer zone, hauteur @card@ mètre, hauteur dépasser, appel contester, vitrer, dont hauteur dépasser, urbaniser, recevabilité <unknown> appel, cahier charge lotissement, charge lotissement, |

*10 premiers termes de 1 à 3 mots sélectionnés à l'aide du coefficient de corrélation ngl (cf. § 2.2.3.2)*

Tableau 4.9 – Terminologies des circonstances factuelles découvertes en combinant les K-medoïdes et la distance cosinus sur $\mathcal{D}_{doris}$.

22. DORIS : dommages-intérêts pour trouble anormal de voisinage

## 4.5 Conclusion

Les circonstances factuelles organisent les décisions d'une même catégorie de demande mais sont illimitées car elles correspondent aux faits courants de la vie. Leur découverte est indispensable afin de rapprocher les litiges non décidés des cas similaires de la jurisprudence. Ce chapitre aborde ce problème comme une tâche de catégorisation non supervisée de documents. La proposition faite ici est double : (i) l'apprentissage d'une métrique de dis-similarité en considérant qu'un document est obtenu par transformation de tout autre document, (ii) l'exploitation de la faible quantité de catégorisations manuelles pour sélectionner la représentation de texte qui correspond au mieux à la sémantique des circonstances factuelles. Le schéma sélectionné permet de transformer de nouveaux corpus non annotés afin d'y découvrir les circonstances factuelles par catégorisation non supervisée. Les expérimentations montrent une amélioration considérable par rapport au modèle de base TF-IDF. La silhouette reste néanmoins faible. Ce qui signifie que la réduction de dimension par FNM est efficace mais il faudrait la combiner avec de meilleurs modèles vectoriels ou mieux l'intégrer au processus de catégorisation. Une approche semblable à celle proposée par Xie & Xing [2013] basée sur l'allocation latente de Dirichlet, mériterait d'être étudiée. Néanmoins cette sélection de représentation permet d'obtenir une assez bonne efficacité de catégorisation sur le corpus annoté. En effet, nous obtenons, avec respectivement les K-moyennes et les K-medoïdes, 0.599 et 0.551 de $F_1$-mesure, correspondant à 0.407 et 0.398 de ARI, et 0.423 et 0.421 de NMI pour un nombre de clusters déterminé automatiquement. Ces résultats se traduisent aussi sur la largeur moyenne de la silhouette autant pour le corpus annoté (0.438 et 0.526 respectivement) que pour les six corpus non annotés utilisés (entre 0.403 et 0.770 au maximum pour toutes les catégories). La métrique apprise s'accorde mieux avec les K-moyennes que les autres distances selon différents indices de validation, et même pour la détermination du nombre de clusters.

Dans ce chapitre, la représentation vectorielle n'est sélectionnée que sur un seul corpus. Il serait intéressant à l'avenir d'annoter plusieurs corpus pour sélectionner la représentation qui correspond le mieux en moyenne aux circonstances factuelles sur l'ensemble de ces catégorisations manuelles. De plus, les expérimentations doivent être étendues au regroupement à

chevauchement pour les affaires concernant plus d'une circonstance factuelle.

# Bibliographie


ACE. 2005. *ACE (Automatic Content Extraction) English Annotation Guidelines for Events*. 5.4.3 edn. Linguistic Data Consortium.

ACE. 2008. *ACE (Automatic Content Extraction) English Annotation Guidelines for Relations*. 6.2 edn. Linguistic Data Consortium. https ://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/english-relations-guidelines-v6.2.pdf.

Afanador, Nelson Lee, Smolinska, Agnieszka, Tran, Thanh N, & Blanchet, Lionel. 2016. Unsupervised random forest : a tutorial with case studies. *Journal of chemometrics*, **30**(5), 232–241.

Afzali, Maedeh, & Kumar, Suresh. 2018. An extensive study of similarity and dissimilarity measures used for text document clustering using k-means algorithm. *I.J. Information Technology and Computer Science*, **9**, 64–73.

Ahn, David. 2006. The stages of event extraction. *Pages 1–8 of : Proceedings of the workshop on annotating and reasoning about time and events*. Association for Computational Linguistics.

Amami, Rimah, Ayed, Dorra Ben, & Ellouze, Noureddine. 2015. Practical selection of svm supervised parameters with different feature representations for vowel recognition. *Corr*, **abs/1507.06020**.

Arora, Sanjeev, Liang, Yingyu, & Ma, Tengyu. 2017. A simple but tough-to-beat baseline for sentence embeddings. *In : Proceedings of iclr 2017*.

Bakkelund, Daniel. 2009. An LCS-based string metric. *Olso, norway : University of oslo*.

Balabantaray, Rakesh Chandra, Sarma, Chandrali, & Jha, Monica. 2015. Document clustering using k-means and k-medoids. *arxiv preprint arxiv :1502.07938*.

Baldwin, Breck. 2009. *Coding chunkers as taggers : IO, BIO, BMEWO, and BMEWO+*.

Bandalos, Deborah L, & Boehm-Kaufman, Meggen R. 2010. Four common misconceptions in exploratory factor analysis. *Pages 81–108 of : Statistical and methodological myths and urban legends*. Routledge.

Baraldi, Andrea, & Blonda, Palma. 1999. A survey of fuzzy clustering algorithms for pattern recognition. i. *Ieee transactions on systems, man, and cybernetics, part b (cybernetics)*, **29**(6), 778–785.

Bazzoli, Caroline, & Lambert-Lacroix, Sophie. 2018. Classification based on extensions of ls-pls using logistic regression : application to clinical and multiple genomic data. *Bmc bioinformatics*, **19**(1), 314.

Ben-Hur, Asa, & Weston, Jason. 2010. A user's guide to support vector machines. *Chap. 13, pages 223–239 of : Data mining techniques for the life sciences*. Totowa, NJ : Humana Press.

Bezdek, James C, Ehrlich, Robert, & Full, William. 1984. Fcm : The fuzzy c-means clustering algorithm. *Computers & geosciences*, **10**(2-3), 191–203.

Blei, David M., Ng, Andrew Y., & Jordan, Michael I. 2003. Latent Dirichlet Allocation. *the Journal of Machine Learning Research*, **3**, 993–1022.

Bray, J Roger, & Curtis, John T. 1957. An ordination of the upland forest communities of southern wisconsin. *Ecological monographs*, **27**(4), 325–349.

Breiman, Leo. 2001. Random forests. *Machine learning*, **45**(1), 5–32.

Breiman, Leo, Friedman, J. H., Olshen, R. A., & Stone, C. J. 1984. *Classification and regression trees*. Statistics/Probability Series. Belmont, California, U.S.A. : Wadsworth Publishing Company.

Brown, Ralf D. 2013. Selecting and weighting n-grams to identify 1100 languages. *Pages 475–483 of : International conference on text, speech and dialogue*. Springer.

Burrows, John F. 1992. Not unles you ask nicely : The interpretative nexus between analysis and information. *Literary and linguistic computing*, **7**(2), 91–109.

Cardellino, Cristian, Teruel, Milagro, *et al.* . 2017. A low-cost, high-coverage legal named entity recognizer, classifier and linker. *Pages 9–18 of : Proceedings of the 16th edition of the international conference on articial intelligence and law*. ACM.

Charlet, Delphine, & Damnati, Geraldine. 2017. Simbow at semeval-2017 task 3 : Soft-cosine semantic similarity between questions for community question answering. *Pages 315–319 of : Proceedings of the 11th international workshop on semantic evaluation (semeval-2017)*.

Charlet, Delphine, & Damnati, Géraldine. 2018. Similarité textuelle pour l'associationde documents journalistiques. *In : 15e conférence en recherche d'information et applications (coria)*.

Chau, Michael, Xu, Jennifer J, & Chen, Hsinchun. 2002. Extracting meaningful entities from police narrative reports. *Pages 1–5 of : Proceedings of the 2002 annual national conference on digital government research*. Digital Government Society of North America.

Chiticariu, Laura, Krishnamurthy, Rajasekar, Li, Yunyao, Reiss, Frederick, & Vaithyanathan, Shivakumar. 2010. Domain adaptation of rule-based annotators for named-entity recognition tasks. *Pages 1002–1012 of : Proceedings of the 2010 conference on empirical methods in natural language processing*. Association for Computational Linguistics.

Cohen, Jacob. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, **20**(1), 37–46.

Cortes, Corinna, & *Vap*nik, Vladimir. 1995. Support-vector networks. *Machine learning*, **20**(3), 273–297.

Cover, Thomas, & Hart, Peter. 1967. Nearest neighbor pattern classification. *Ieee transactions on information theory*, **13**(1), 21–27.

Deerwester, Scott, Dumais, Susan T, Furnas, George W, Landauer, Thomas K, & Harshman, Richard. 1990. Indexing by latent semantic analysis. *Journal of the american society for information science*, **41**(6), 391–407.

Dong, Yan-Shi, & Han, Ke-Song. 2005. Boosting svm classifiers by ensemble. *Pages 1072–1073 of : Special interest tracks and posters of the 14th international conference on world wide web*. ACM.

Dozier, Christopher, Kondadadi, Ravikumar, Light, Marc, Vachher, Arun, Veeramachaneni, Sriharsha, & Wudali, Ramdev. 2010. Named entity recognition and resolution in legal text. *Pages 27–43 of : Semantic processing of legal texts*. Springer.

Duda, Richard O, Hart, Peter E, *et al.* . 1973. *Pattern classification and scene analysis*. Vol. 3. Wiley New York.

Dudani, Sahibsingh A. 1976. The distance-weighted k-nearest-neighbor rule. *Ieee transactions on systems, man, and cybernetics*, 325–327.

Dumais, Susan T., Furnas, George W., Landauer, Thomas K., Deerwester, Scott, & Harshman, Richard. 1988. Using latent semantic analysis to improve access to textual information. *Pages 281–285 of : Proceedings of the sigchi conference on human factors in computing systems*. Acm.

Durif, Ghislain, Modolo, Laurent, Michaelsson, Jakob, Mold, Jeff E, Lambert-Lacroix, Sophie, & Picard, Franck. 2017. High dimensional classification with combined adaptive sparse pls and logistic regression. *Bioinformatics*, **34**(3), 485–493.

Elman, Jeffrey L. 1990. Finding structure in time. *Cognitive science*, **14**(2), 179–211.

Ester, Martin, Kriegel, Hans-Peter, Sander, Jörg, Xu, Xiaowei, *et al.* . 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. *Pages 226–231 of : Kdd*, vol. 96.

Fang, Anjie, Macdonald, Craig, Ounis, Iadh, & Habel, Philip. 2016. Using word embedding to evaluate the coherence of topics from twitter data. *Pages 1057–1060 of : Proceedings of the 39th international acm sigir conference on research and development in information retrieval*. ACM.

Finkel, Jenny Rose, Grenager, Trond, & Manning, Christopher. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. *Pages 363–370 of : Proceedings of the 43rd annual meeting on association for computational linguistics*. Association for Computational Linguistics.

Forgey, Edward. 1965. Cluster analysis of multivariate data : Efficiency vs. interpretability of classification. *Biometrics*, **21**(3), 768–769.

Frank, Eibe, Hall, MA, & Witten, IH. 2016. The weka workbench. *Data mining : Practical machine learning tools and techniques. burlington : Morgan kaufmann*.

Frantzi, Katerina, Ananiadou, Sophia, & Mima, Hideki. 2000. Automatic recognition of multi-word terms :. the c-value/nc-value method. *International journal on digital libraries*, **3**(2), 115–130.

Galavotti, Luigi, Sebastiani, Fabrizio, & Simi, Maria. 2000. Experiments on the use of feature selection and negative evidence in automated text categorization. *Pages 59–68 of : International conference on theory and practice of digital libraries*. Springer.

Gou, Jianping, Xiong, Taisong, & Kuang, Yin. 2011. A novel weighted voting for k-nearest neighbor rule. *Jcp*, **6**(5), 833–840.

Grave, E, Mikolov, T, Joulin, A, & Bojanowski, P. 2017. Bag of tricks for efficient text classification. *Pages 427–431 of : Proceedings of the 15th conference of the european chapter of the association for computational linguistics*.

Guttman, Louis. 1954. Some necessary conditions for common-factor analysis. *Psychometrika*, **19**(2), 149–161.

Halkidi, Maria, Batistakis, Yannis, & Vazirgiannis, Michalis. 2001. On clustering validation techniques. *Journal of intelligent information systems*, **17**(2-3), 107–145.

Hanisch, Daniel, Fundel, Katrin, *et al.* . 2005. ProMiner : rule-based protein and gene entity recognition. *BMC bioinformatics*, **6**(1), 14.

Harispe, Sébastien, Ranwez, Sylvie, Janaqi, Stefan, & Montmain, Jacky. 2013. The semantic measures library and toolkit : fast computation of semantic similarity and relatedness using biomedical ontologies. *Bioinformatics*, **30**(5), 740–742.

Harispe, Sébastien, Ranwez, Sylvie, Janaqi, Stefan, & Montmain, Jacky. 2015. Semantic similarity from natural language and ontology analysis. *Synthesis lectures on human language technologies*, **8**(1), 1–254.

Hathaway, Richard J, Davenport, John W, & Bezdek, James C. 1989. Relational duals of the c-means clustering algorithms. *Pattern recognition*, **22**(2), 205–212.

Huang, Anna. 2008. Similarity measures for text document clustering. *Pages 9–56 of : Proceedings of the sixth new zealand computer science research student conference (nzcsrsc2008), christchurch, new zealand*, vol. 4.

Hubert, Lawrence, & Arabie, Phipps. 1985. Comparing partitions. *Journal of classification*, **2**(1), 193–218.

Im, Chan Jong, Mandl, Thomas, *et al.* . 2017. Text classification for patents : Experiments with unigrams, bigrams and different weighting methods. *International journal of contents*, **13**(2).

Jaccard, Paul. 1901. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin de la société vaudoise sciences naturelles*, **37**, 547–579.

Jones, K. Sparck, Walker, Steve, & Robertson, Stephen E. 2000. A probabilistic model of information retrieval : development and comparative experiments. *Information processing & management*, **36**(6), 809–840.

Jordan, MI. 1986. *Serial order : a parallel distributed processing approach. technical report, june 1985-march 1986*. Tech. rept. California Univ., San Diego, La Jolla (USA). Inst. for Cognitive Science.

Kaiser, Henry F. 1960. The application of electronic computers to factor analysis. *Educational and psychological measurement*, **20**(1), 141–151.

Kaufman, Leonard, & Rousseeuw, Peter J. 1987. Clustering by means of medoids. *Page 405–416 of :* Dodge, Yadolah (ed), *Statistical data analysis based on the l1-norm*. North Holland/Elsevier. Amsterdam.

Kitoogo, Fredrick Edward, & Baryamureeba, Venansius. 2007. A methodology for feature selection in named entity recognition. *Strengthening the role of ict in development*, 88.

Kittler, Josef, Hater, Mohamad, & Duin, Robert PW. 1996. Combining classifiers. *Pages 897–901 of : Proceedings of 13th international conference on pattern recognition*, vol. 2. IEEE.

Klinger, Roman, & Friedrich, Christoph M. 2009. Feature subset selection in conditional random fields for named entity recognition. *Pages 185–191 of : Proceedings of the international conference ranlp-2009*.

Konkol, Michal, & Konopík, Miloslav. 2015. Segment representations in named entity recognition. *Pages 61–70 of : International conference on text, speech, and dialogue*. Springer.

Krishnapuram, Raghu, Joshi, Anupam, Nasraoui, Olfa, & Yi, Liyu. 2001. Low-complexity fuzzy relational clustering algorithms for web mining. *Ieee transactions on fuzzy systems*, **9**(4), 595–607.

Kríž, Vincent, Hladká, Barbora, *et al.* . 2014. Statistical recognition of references in czech court decisions. *Pages 51–61 of :* Gelbukh, Alexander, Espinoza, Félix Castro, & Galicia-Haro, Sofía N. (eds), *Human-inspired computing and its applications : 13th mexican international conference on artificial intelligence, micai 2014, tuxtla gutiérrez, mexico, november 16-22, 2014. proceedings, part i.* Cham : Springer International Publishing.

Kuncheva, Ludmila I. 2004. *Combining pattern classifiers : methods and algorithms*. John Wiley & Sons.

Kusner, Matt, Sun, Yu, Kolkin, Nicholas, & Weinberger, Kilian. 2015. From word embeddings to document distances. *Pages 957–966 of : International conference on machine learning*.

Kvalseth, Tarald O. 1987. Entropy and correlation : Some comments. *Ieee transactions on systems, man, and cybernetics*, **17**(3), 517–519.

Lacroux, Alain. 2011. Les avantages et les limites de la méthode «partial least square»(pls) : une illustration empirique dans le domaine de la grh. *Revue de gestion des ressources humaines*, 45–64.

Lafferty, John, McCallum, Andrew, & Pereira, Fernando C. N. 2001. Conditional random fields : probabilistic models for segmenting and labeling sequence data. *International Conference on Machine Learning*.

Lample, Guillaume, Ballesteros, Miguel, *et al.* . 2016. Neural architectures for named entity recognition. *arxiv preprint*. arXiv :1603.01360.

Lan, Man, Tan, Chew Lim, Su, Jian, & Lu, Yue. 2009. Supervised and traditional term weighting methods for automatic text categorization. *Ieee transactions on pattern analysis and machine intelligence*, **31**(4), 721–735.

Le, Quoc, & Mikolov, Tomas. 2014a. Distributed representations of sentences and documents. *Pages 1188–1196 of : International conference on machine learning*.

Le, Quoc, & Mikolov, Tomas. 2014b. Distributed representations of sentences and documents. *Pages 1188–1196 of : International conference on machine learning*.

Li, Yaoyong, Zaragoza, Hugo, Herbrich, Ralf, Shawe-Taylor, John, & Kandola, Jaz. 2002. The perceptron algorithm with uneven margins. *Pages 379–386 of : Icml*, vol. 2.

Liu, Dong C., & Nocedal, Jorge. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, **45**(1), 503–528.

Liu, Huan, & Motoda, Hiroshi. 2012. *Feature selection for knowledge discovery and data mining*. Vol. 454. Springer Science & Business Media.

Liu, Jingjing, Pasupat, Panupong, Cyphers, Scott, & Glass, Jim. 2013. Asgard : A portable architecture for multilingual dialogue systems. *Pages 8386–8390 of : Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE.

Liu, Yushu, & Rayens, William. 2007. PLS and dimension reduction for classification. *Computational statistics*, **22**(2), 189–208.

Manning, Christopher D, Raghavan, Prabhakar, & Schütze, Hinrich. 2009a. Flat clustering. *Chap. 16, pages 349–375 of : Introduction to information retrieval*. Cambridge : Cambridge university press.

Manning, Christopher D, Raghavan, Prabhakar, & Schütze, Hinrich. 2009b. Scoring, term weighting and the vector space model. *Chap. 6, pages 109–133 of : Introduction to information retrieval*. Cambridge : Cambridge university press.

Martineau, Justin, Finin, Tim, *et al.* . 2009. Delta tfidf : An improved feature space for sentiment analysis. *Icwsm*, **9**, 106.

McCallum, Andrew Kachites. 2012. *MALLET : A Machine Learning for Language Toolkit*.

McCulloch, Warren S, & Pitts, Walter. 1943. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, **5**(4), 115–133.

Mikheev, Andrei, Moens, Marc, & Grover, Claire. 1999. Named entity recognition without gazetteers. *Pages 1–8 of : Proceedings of the ninth conference on european chapter of the association for computational linguistics*. Association for Computational Linguistics.

Mikolov, Tomas, Chen, Kai, Corrado, Greg, & Dean, Jeffrey. 2013. Efficient estimation of word representations in vector space. *arxiv preprint arxiv :1301.3781*.

Mussard, Stéphane, & Souissi-Benrejab, Fattouma. 2018. Gini-pls regressions. *Journal of quantitative economics*, April, 1–36.

Nadeau, David, & Sekine, Satoshi. 2007. A survey of named entity recognition and classification. *Lingvisticae investigationes*, **30**(1), 3–26.

Nallapati, Ramesh, Surdeanu, Mihai, & Manning, Christopher. 2010. Blind domain transfer for named entity recognition using generative latent topic models. *Pages 281–289 of : Proceedings of the nips 2010 workshop on transfer learning via rich generative models*.

Nefti, Samia, & Oussalah, Mourad. 2004. Probabilistic-fuzzy clustering algorithm. *Pages 4786–4791 of : 2004 ieee international conference on systems, man and cybernetics (ieee cat. no. 04ch37583)*, vol. 5. IEEE.

Ng, Hwee Tou, Goh, Wei Boon, & Low, Kok Leong. 1997. Feature selection, perceptron learning, and a usability case study for text categorization. *Pages 67–73 of : Acm sigir forum*, vol. 31. ACM.

Nguyen, Thien Huu, Cho, Kyunghyun, & Grishman, Ralph. 2016. Joint event extraction via recurrent neural networks. *Pages 300–309 of : Hlt-naacl*.

Nigam, Kamal, Lafferty, John, & McCallum, Andrew. 1999. Using maximum entropy for text classification. *Pages 61–67 of : Ijcai-99 workshop on machine learning for information filtering*, vol. 1.

Olkin, Ingram, & Yitzhaki, Shlomo. 1992. Gini regression analysis. *International statistical review/revue internationale de statistique*, 185–196.

Paatero, Pentti, & Tapper, Unto. 1994. Positive matrix factorization : A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, **5**(2), 111–126.

Pagliardini, Matteo, Gupta, Prakhar, & Jaggi, Martin. 2018. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. *In : Naacl 2018 - conference of the north american chapter of the association for computational linguistics*.

Palm, Rasmus Berg, Hovy, Dirk, Laws, Florian, & Winther, Ole. 2017. End-to-end information extraction without token-level supervision. *In : Proceedings of the workshop on speech-centric natural language processing*.

Palmer, David D, & Day, David S. 1997. A statistical profile of the named entity task. *Pages 190–193 of : Proceedings of the fifth conference on applied natural language processing*. Association for Computational Linguistics.

Paltoglou, Georgios, & Thelwall, Mike. 2010. A study of information retrieval weighting schemes for sentiment analysis. *Pages 1386–1395 of : Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics.

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.

Pennington, Jeffrey, Socher, Richard, & Manning, Christopher. 2014. Glove : Global vectors for word representation. *Pages 1532–1543 of : Proceedings of the 2014 conference on empirical methods in natural language processing (emnlp)*.

Persson, Caroline. 2012. *Machine learning for tagging of biomedical literature*. Closing project report, Technical University of Denmark,DTU Informatics.

Polifroni, Joe, & Mairesse, François. 2011. Using latent topic features for named entity extraction in search queries. *Pages 2129–2132 of : Interspeech*.

Price, Patti J. 1990. Evaluation of spoken language systems : The atis domain. *In : Speech and natural language : Proceedings of a workshop held at hidden valley, pennsylvania, june 24-27, 1990*.

Pudil, Pavel, Novovičová, Jana, & Kittler, Josef. 1994. Floating search methods in feature selection. *Pattern recognition letters*, **15**(11), 1119–1125.

Quinlan, J Ross. 1993. C4. 5 : Programming for machine learning. *Morgan kauffmann*, **38**, 48.

Rabiner, Lawrence R. 1989. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the ieee*, **77**(2), 257–286.

Raman, Baranidharan, & Ioerger, Thomas R. 2003. Enhancing learning using feature and example selection. *Texas a&m university, college station, tx, usa*.

Rand, William M. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the american statistical association*, **66**(336), 846–850.

Raschka, Sebastian. 2014. Naive Bayes and Text Classification I : Introduction and Theory. *arxiv preprint arxiv :1410.5329*.

Řehůřek, Radim, & Sojka, Petr. 2010. Software Framework for Topic Modelling with Large Corpora. *Pages 45–50 of : Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta : ELRA. `http://is.muni.cz/publication/884893/en`.

Rish, Irina. 2001. An empirical study of the naive bayes classifier. *Pages 41–46 of : Ijcai 2001 workshop on empirical methods in artificial intelligence*, vol. 3. IBM New York.

Rosenblatt, Frank. 1958. The perceptron : a probabilistic model for information storage and organization in the brain. *Psychological review*, **65**(6), 386.

Rousseeuw, Peter J. 1987. Silhouettes : a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, **20**, 53–65.

Ruparel, Nidhi H, Shahane, Nitin M, & Bhamare, Devyani P. 2013. Learning from small data set to build classification model : A survey. *Internationla journal of computer applications*, **975**(8887), 23–26.

Sabzi, Akhtar, Farjami, Yaghoub, & ZiHayat, Morteza. 2011. An improved fuzzy k-medoids clustering algorithm with optimized number of clusters. *Pages 206–210 of : Proceedings of the 11th International Conference on Hybrid Intelligent Systems (HIS)*. IEEE.

Salton, Gerard, & Buckley, Christopher. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management*, **24**(5), 513–523.

Salvador, Stan, & Chan, Philip. 2004. Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. *Pages 576–584 of : 16th ieee international conference on tools with artificial intelligence*. IEEE.

Schechtman, Edna, & Yitzhaki, Shlomo. 2003. A family of correlation coefficients based on the extended gini index. *The journal of economic inequality*, **1**(2), 129–146.

Schmid, Helmut. 1994. *Treetagger - a part-of-speech tagger for many languages*.

Schütze, Hinrich, Hull, David A, & Pedersen, Jan O. 1995. A comparison of classifiers and document representations for the routing problem. *Pages 229–237 of : Proceedings of the 18th annual international acm sigir conference on research and development in information retrieval*. ACM.

Sharnagat, Rahul. 2014. *Named entity recognition : A literature survey*. Tech. rept. Center For Indian Language Technology.

Shi, Jianbo, & Malik, Jitendra. 2000. Normalized cuts and image segmentation. *Departmental papers (cis)*, 107.

Sidorov, Grigori, Gelbukh, Alexander, Gómez-Adorno, Helena, & Pinto, David. 2014. Soft similarity and soft cosine measure : Similarity of features in vector space model. *Computación y sistemas*, **18**(3), 491–504.

Singh, Sonia, & Gupta, Priyanka. 2014. Comparative study ID3, CART and C4.5 decision tree algorithm : a survey. *Int j adv inf sci technol [internet]*, **27**, 97–103.

Siniakov, Peter. 2008. *Gropus an adaptive rule-based algorithm for information extraction*. Ph.D. thesis, Freie Universität Berlin.

Sparck Jones, Karen. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, **28**(1), 11–21.

Strehl, Alexander, Ghosh, Joydeep, & Mooney, Raymond. 2000. Impact of similarity measures on web-page clustering. *Page 64 of : Workshop on artificial intelligence for web search (aaai 2000)*, vol. 58.

Tenenhaus, Michel. 2005. La regression logistique PLS. *Chap. 12, pages 263–276 of :* Droesbeke, Jean-Jacques, Lejeune, Michel, & Saporta, Gilbert (eds), *Modèles statistiques pour données qualitatives*. Editions Technip.

Thorndike, Robert L. 1953. Who belongs in the family ? *Psychometrika*, **18**(4), 267–276.

Tulyakov, Sergey, Jaeger, Stefan, Govindaraju, Venu, & Doermann, David. 2008. Review of classifier combination methods. *Pages 361–386 of : Machine learning in document analysis and recognition*. Springer.

Van Asch, Vincent. 2013. Macro-and micro-averaged evaluation measures [[basic draft]]. *Belgium : CLiPS*, 1–27.

Vapnik, Vladimir N. 1995. The nature of statistical learning. *Theory*.

Viera, Anthony J, Garrett, Joanne M, *et al.* . 2005. Understanding interobserver agreement : the kappa statistic. *Fam med*, **37**(5), 360–363.

Vijaymeena, MK, & Kavitha, K. 2016. A survey on similarity measures in text mining. *Machine learning and applications : An international journal*, **3**(2), 19–28.

Vinh, Nguyen Xuan, Epps, Julien, & Bailey, James. 2010. Information theoretic measures for clusterings comparison : Variants, properties, normalization and correction for chance. *Journal of machine learning research*, **11**(Oct), 2837–2854.

Vinyals, Oriol, Fortunato, Meire, & Jaitly, Navdeep. 2015. Pointer networks. *Pages 2692–2700 of : Advances in neural information processing systems*.

Viterbi, Andrew James. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Ieee transactions on information theory*, **13**(2), 260–269.

Von Luxburg, Ulrike. 2007. A tutorial on spectral clustering. *Statistics and computing*, **17**(4), 395–416.

Wallach, Hanna M. 2004. *Conditional Random Fields : An Introduction*. Tech. rept. University of Pennsylvania Department of Computer and Information Science.

Waltl, Bernhard, Bonczek, Georg, & Matthes, Florian. 2018. Rule-based information extraction : Advantages, limitations, and perspectives. *Jusletter it*, Feb.

Wang, Fei, & Sun, Jimeng. 2015. Survey on distance metric learning and dimensionality reduction in data mining. *Data mining and knowledge discovery*, **29**(2), 534–564.

Wang, Sida, & Manning, Christopher D. 2012. Baselines and bigrams : Simple, good sentiment and topic classification. *Pages 90–94 of : Proceedings of the 50th annual meeting of the association for computational linguistics : Short papers-volume 2*. Association for Computational Linguistics.

Welch, Lloyd R. 2003. Hidden Markov models and the Baum-Welch algorithm. *Ieee information theory society newsletter*, **53**(4), 10–13.

Wold, Herman. 1966. Estimation of principal components and related models by iterative least squares. *Multivariate analysis*, 391–420.

Wu, Haibing, Gu, Xiaodong, & Gu, Yiwei. 2017. Balancing between over-weighting and under-weighting in supervised term weighting. *Information processing & management*, **53**(2), 547–557.

Wu, Harry, & Salton, Gerard. 1981. A comparison of search term weighting : term relevance vs. inverse document frequency. *Pages 30–39 of : Acm sigir forum*, vol. 16. ACM.

Wyner, Adam Z. 2010. Towards annotating and extracting textual legal case elements. *Informatica e diritto : special issue on legal ontologies and artificial intelligent techniques*, **19**(1-2), 9–18.

Xiao, Richard. 2010. Corpus creation. *Chap. 7, page 146–165 of :* Indurkhya, Nitin, & Damerau, Fred J . (eds), *Handbook of natural language processing*, second edn. Chapman and Hall.

Xie, Pengtao, & Xing, Eric P. 2013. Integrating document clustering and topic modeling. *In : Proceedings of the twenty-ninth conference on uncertainty in artificial intelligence (uai2013)*.

Yang, Bishan, & Mitchell, Tom. 2016. Joint Extraction of Events and Entities within a Document Context. *Pages 289–299 of : Proceedings of naacl-hlt*.

Yang, Yiming, & Pedersen, Jan O. 1997. A comparative study on feature selection in text categorization. *Pages 412–420 of : Icml*, vol. 97.

Zeng, Xue-Qiang, Wang, Ming-Wen, & Nie, Jian-Yun. 2007. Text classification based on partial least square analysis. *Pages 834–838 of : Proceedings of the 2007 acm symposium on applied computing*. ACM.

Zhu, Xiaojin. 2010. *Conditional random fields*. CS769 Spring 2010 Advanced Natural Language Processing.