

UE : FONDEMENTS THEORIQUES DE L’INFORMATIQUE (MT42)

EXPOSE SUR L’IMPLICATION DES MATHEMATIQUES DANS L’INFORMATIQUE

THEME :



Exposé Présenté par :

KOMBATE Jean

KPAKPASSIM EssoWédéou

SENAYA Sharon

Évalué par :

Dr EDARH BOSSOU

Implication des mathématiques dans le traitement automatique du langage naturel

Sommaire

[Introduction 2](#_Toc64052580)

[I. Comment fonctionne le TALN ? 2](#_Toc64052581)

[**a.** L’étiquetage morphosyntaxique 3](#_Toc64052582)

[**b.** Arbre syntaxique / Arborescence 3](#_Toc64052583)

[ Principe de fonctionnement de l’arbre syntaxique 3](#_Toc64052584)

[**c.** La sémantique 5](#_Toc64052585)

[II. Etude de quelques fonctions de base du TALN : 6](#_Toc64052586)

[ La racinisation 6](#_Toc64052587)

[a. Exposition de l’algorithme de porter 6](#_Toc64052588)

[ TF-IDF 8](#_Toc64052589)

[ Fréquence du terme (TF) 9](#_Toc64052590)

[ Fréquence inverse de document (IDF) 9](#_Toc64052591)

[ Calcul de TF-IDF 10](#_Toc64052592)

[ Exemple d’utilisation de TF-IDF 10](#_Toc64052593)

[III. Quelques outils du TALN 11](#_Toc64052594)

[Conclusion 13](#_Toc64052595)

[Bibliographie 14](#_Toc64052596)

# Introduction

Le **Langage Naturel** est le moyen de communication employé par les êtres humains de façon inné dans la vie quotidienne. L’anglais, l’espagnol et le français par exemple sont des exemples de [**langages naturels**](https://www.inbenta.com/en/languages-supported/). Ils sont construits selon une syntaxe, une grammaire et peuvent contenir beaucoup d’ambiguïtés.

Ils diffèrent ainsi des **Langages Formels**, utilisés pour transférer des informations à propos desquelles aucune ambiguïté n’est possible. Les mathématiques, les langages informatiques comme le Java ou le binaire sont des exemples de langages formels.

Les ordinateurs savent très bien interpréter les langages formels, mais un des plus grands challenges de l’informatique est de créer des solutions capables de comprendre le langage naturel. La discipline de l’intelligence artificielle nommée le **Traitement Automatique du Langage Naturel** (TALN ou NLP en anglais) est celle qui se consacre aux interactions entre les machines et l’humain s’attache précisément à cet objectif.

# Comment fonctionne le TALN ?

Qu’il s’agisse de traduction automatique ou d’une discussion avec un chatbot, les méthodes de TALN prêtent attention aux hiérarchies afin de mettre en cohérence les mots entre eux. Ce n’est pas chose facile car beaucoup de mots ont plusieurs sens : une « souris » par exemple peut définir l’animal, le petit boîtier servant à se diriger sur un écran d’ordinateur, ou même un morceau précis de viande d’agneau.

Le traitement automatique du langage naturel représente donc un défi colossal dans le domaine de l’informatique. Le langage peut en effet être à double sens et pour le comprendre, il est nécessaire de bien connaître le contexte dans lequel il s’insère. De nombreux utilisateurs ont d’ailleurs déjà fait l’expérience de conversations quelque peu chaotiques avec les chatbots qui sont fréquemment utilisés pour les chats de services clients. Toutefois, les ordinateurs comprennent de mieux en mieux le langage humain. Pour leur faire apprendre le langage, différents domaines d’application de la linguistique sont pris en compte :

* La **morphologie**, qui s’intéresse à la composition des mots et leur corrélation avec d’autres mots.
* La **syntaxe**, qui définit comment les mots sont agencés dans une phrase.
* La **sémantique**, qui correspond à la signification des mots et des groupes de mots
* La **pragmatique**, grâce à laquelle le contexte est pris en compte.
* Enfin la **phonologie**, qui s’occupe des sonorités de la langue orale, ce qui est important pour la reconnaissance vocale.

## L’étiquetage morphosyntaxique

Le traitement automatique du langage naturel prend dans un premier temps en considération la **morphologie** : cela repose sur **la fonction des mots**. À l’école primaire, nous avons assimilé une forme simplifiée de ce processus en apprenant à différencier un substantif d’un verbe ou d’un pronom. Ceci peut toutefois être compliqué pour les ordinateurs car un mot peut, suivant la phrase dans laquelle il intervient, changer de fonction.

Pour **passer outre ces ambiguïtés**, il existe différentes méthodes : les plus anciennes se basent sur de grands **corpus linguistiques** comme Brown Corpus [[1]](#footnote-1) ou le British National Corpus[[2]](#footnote-2). Ces corpus de textes rassemblent au total des millions de mots, qui sont étiquetés et à partir desquels des règles d’apprentissage pour l’étiquetage peuvent être déduites.

Les programmes d’étiquetage morphosyntaxique les plus récents utilisent des **algorithmes « autodidactes**», ce qui signifie qu’ils déduisent des règles automatiquement des corpus de textes disponibles et les utilisent pour définir les fonctions d’autres mots. L’un des meilleurs exemples de méthodes d’étiquetage basées sur les algorithmes, est le **Brill tagger**, qui alloue d’abord au mot sa fonction la plus utilisée pour ensuite déduire une autre fonction grâce à des règles définies. Une règle peut être par exemple : si le premier mot d’une phrase est un nom propre, le deuxième va probablement être un verbe.

## Arbre syntaxique / Arborescence

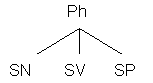
Dans un deuxième temps, un travail sur **la syntaxe** va être effectué pour comprendre **la structure des phrases**. La linguistique informatique utilise ainsi des arborescences, dans lesquelles les syntagmes vont être décomposés. On trouve par exemple des syntagmes nominaux qui peuvent se composer d’un nom propre ou substantif et d’un article, ou de syntagmes verbaux composés d’un verbe et d’une phrase nominale.

Cette décomposition de la phrase en syntagmes s’appelle en anglais le **parsing**, et le diagramme correspondant le **parser tree**.

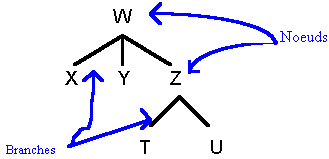
## Principe de fonctionnement de l’arbre syntaxique

En français, on parle habituellement d’arbre syntaxique. Chaque langue a sa propre grammaire ; ce qui signifie que l’ordre des mots, la structure des phrases fonctionnent différemment d’une langue à une autre. La grammaire d’une langue peut être  programmée manuellement pour un programme informatique ou peut être apprise à partir d’un corpus de texte sur lequel les structures de phrases sont annotées.

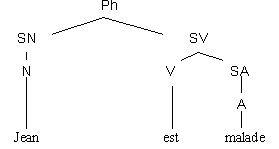
Graphiquement, les règles de réécriture se représentent sous forme d'arbres syntaxiques. Par exemple:



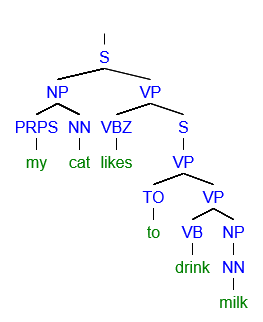
Un arbre est constitué de branches et de nœuds:



Chaque nœud a une étiquette qui correspond à un syntagme ou à une catégorie. Le syntagme est une projection alors que la catégorie est un nœud terminal. A chaque nœud terminal est associé un mot (ou item lexical).



***Arborescence pour la phrase*** : « Jean est malade ». La phrase (PH) est décomposée en un syntagme nominal (SN) et un syntagme verbal (SV) qui est lui-même ensuite décomposé.

[](https://www.ionos.fr/digitalguide/fileadmin/DigitalGuide/Screenshots/Baumdiagramm-fuer-einen-Satz.png)

***Arborescence version anglaise pour la phrase*** : « My cat likes to drink milk ». La phrase (S) est décomposée en un syntagme nominal (NP) et un syntagme verbal (VP) qui est lui-même ensuite redécomposé.

## La sémantique

Dans un troisième temps, le TALN va amener les linguistes informaticiens à s’intéresser à **la sémantique**. Même si un mot n’a qu’une seule fonction grammaticale et s’inscrit dans une syntaxe claire, il peut arriver qu’il ait **plusieurs significations**. On pourra distinguer des sens différents au mot « opéra » ou « addition » par exemple ci-dessous :

* L’opéra était très bien mis en scène.
* Il n’y avait plus d’opéra au menu.
* L’addition était salée.
* Pour cet exercice, une addition est nécessaire.

Pour une personne « réelle » et non un ordinateur, il n’y pas de grande ambiguïté dans les deux phrases : alors que dans la première, l’opéra se réfère à une œuvre musicale, il s’agit bien sûr dans la deuxième du nom commun d’un gâteau. Cette différenciation est néanmoins plus complexe pour un ordinateur.

Plus troublant encore, la phrase « Je loue un appartement » ; on peut en effet se demander si la personne définie par « je » est locataire (et donc paie pour bénéficier d’une surface habitable) ou propriétaire (et donc reçoit de l’argent en échange d’un logement proposé à une autre personne).

Afin de comprendre la signification du mot, par exemple de « opéra » dans notre premier exemple, les ordinateurs vont s‘intéresser aux mots qui sont placés avant ou après. Ainsi, la locution « mis en scène » va pouvoir lui supposer qu’il s’agit de l’œuvre musicale tandis que le mot « menu » va être assez explicite pour situer la phrase dans l’univers culinaire. Cet apprentissage est issu généralement des corpus de textes qui permettent de corriger la signification de chaque mot.

# Etude de quelques fonctions de base du TALN :

# La racinisation

La racinisation ou désuffixation est un procédé de transformation des flexions en leur radical ou racine. La racine d’un mot correspond à la partie du mot restante une fois que l’on a supprimé son (ses) préfixe(s) et suffixe(s), à savoir son radical.

La racinisation est un procédé très usité dans les applications de traitement automatique du langage naturel, par exemple dans la traduction automatique, la recherche d'information (reconnaissance d'entités) et l'indexation des moteurs de recherche. Les divers algorithmes de racinisation procèdent en deux étapes :

* Un pas de désuffixation qui consiste à ôter aux mots des terminaisons prédéfinies les plus longues possibles,
* Un pas de recodage qui ajoute aux racines obtenues des terminaisons prédéfinies.

Nous nous pencherons sur celle de Porter qui fait les deux étapes en simultané.

L'algorithme de Porter se compose d'une cinquantaine de règles de racinisation/désuffixation classées en sept phases successives (traitement des pluriels et verbes à la troisième personne du singulier, traitement du passé et du progressif, ...). Les mots à analyser passent par tous les stades et, dans le cas où plusieurs règles pourraient leur être appliquées, c'est toujours celle comprenant le suffixe le plus long qui est choisie. La racinisation est accompagnée, dans la même étape, de règles de recodage. Ainsi, par exemple, "troubling" deviendra "troubl" par enlèvement du suffixe marqueur du progressif -ing et sera ensuite transformé en "trouble" et par application de la règle "bl" devient "ble". Cet algorithme comprend aussi cinq règles de contexte, qui indiquent les conditions dans lesquelles un suffixe devra être supprimé. La terminaison en -ing, par exemple, ne sera enlevée que si le radical comporte au moins une voyelle. De cette manière, "troubling" deviendra "troubl", nous l'avons vu, alors que "sing" restera "sing".

## Exposition de l’algorithme de porter

Soit {\displaystyle \scriptstyle v}v représentant une voyelle ('y' est considéré comme une voyelle s'il est précédé par une consonne), *{\displaystyle \scriptstyle c}c* représentant une consonne; et soit {\displaystyle \scriptstyle V}V représentant une suite de voyelles, {\displaystyle \scriptstyle C}C représentant une suite de consonnes, alors, un mot en anglais peut être de l'une des 4 formes suivantes:

{\displaystyle \scriptstyle CVCV\ldots V}

* CVCV…C
* CVCV…V
* VCVC…C
* VCVC…V

Ce qui peut se présenter par C?VCVC…V? ou C ?(VC)m v ?, où m est appelé la mesure d’un mot. Des valeurs différentes présentent des mots différents :

* m =0 : tree, ivy
* m =1 : troubles, oats, trees, ivy
* m =2 : troubles, private, oaten, orrery

Les règles de désuffixation/racinisation sont exprimées sous la forme (condition) S1→S 2 ce qui signifie que si un mot se termine par {\displaystyle \scriptstyle S\_{1}}S 1 et que le préfixe satisfait la condition alors le suffixe {\displaystyle \scriptstyle S\_{1}}S 1 est remplacé par {\displaystyle \scriptstyle S\_{2}}S 2

* {\displaystyle \scriptstyle ^{\*}e}\*e : le préfixe se termine par la lettre {\displaystyle \scriptstyle e}
* {\displaystyle \scriptstyle ^{\*}v^{\*}}\*v\* : le préfixe contient une voyelle
* {\displaystyle \scriptstyle ^{\*}d}\*d : le préfixe se termine par une consonne doublée
* {\displaystyle \scriptstyle ^{\*}o}\*o : le préfixe se termine par {\displaystyle \scriptstyle cvc}cvc où le second {\displaystyle \scriptstyle c}c n'est ni {\displaystyle \scriptstyle w}w, ni  {\displaystyle \scriptstyle x}x, ni {\displaystyle \scriptstyle y}y.

Il est possible d'utiliser des opérateurs booléens: et, ou, non

|  |  |  |  |
| --- | --- | --- | --- |
| **Racines obtenues par le raciniseur de Porter** | | | |
| Étape 1 | A | * SSES → SS * IES → I * SS → SS * S → | caresses → caress  ponies → poni  caress → caress  cats → cat |
| B | * (m>0) EED → EE * (\*v\*) ED → * (\*v\*) ING → | feed → feed,  agreed → agree  plastered → plaster,  bled → bled  motoring → motor,  sing → sing |
| c | * (\*v\*) Y → I | happy → happi,  sky → sky |
| Étape 2 | | * (m>0) ATIONAL → ATE * (m>0) TIONAL → TION * (m>0) ENCI → ENCE * (m>0) ANCI → ANCE | relational → relate  conditional → condition, rational → rational  valenci → valence  hesitansi → hesitance |
| Étape 3 | | * (m>0) ICATE → IC * (m>0) ATIVE → * (m>0) ALIZE → AL * (m>0) ICITI → IC | triplicate → triplic  formative → form  formalize → formal  electriciti → electric |
| Étape 4 | | * (m>1) AL → * (m>1) ANCE → * (m>1) ENCE → * (m>1) ER → | revival → reviv  allowance → allow  inference → infer  airliner → airlin |
| Étape 5 | | * (m>1) E → * (m=1 and not \*o) E → * (m>1 and \*d and \*L)   → lettre non doublée | probate → probat,  rate → rate  cease → ceas  controll → control,  roll → roll |

Testons cet algorithme avec 2 mots: Généralisations et Oscillateurs

**Généralisations**

Étape 1: **Généralisation**

Étape 2: **Généraliser**

Étape 3: **Général**

Étape 4: **Géner**

**Oscillateurs**

Étape 1: **Oscillateur**

Étape 2: **Osciller**

Étape 4: **Oscill**

Étape 5: **Oscil**

L'algorithme de Porter est distribué librement et a été implanté dans de nombreux langages. Cependant, il est efficace pour l'anglais mais pas très adapté au français.

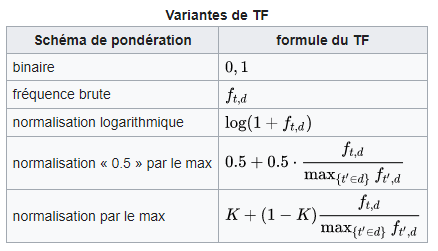
# TF-IDF

Le **TF-IDF** (de l'anglais *term frequency-inverse document frequency*) est une méthode de pondération souvent utilisée en recherche d'information et en particulier dans la fouille de textes. Cette mesure statistique permet d'évaluer l'importance d'un terme contenu dans un document, relativement à une collection ou un corpus. Le poids augmente proportionnellement au nombre d'occurrences du mot dans le document. Il varie également en fonction de la fréquence du mot dans le corpus. Des variantes de la formule originale sont souvent utilisées dans des moteurs de recherche pour apprécier la pertinence d'un document en fonction des critères de recherche de l'utilisateur.

## Fréquence du terme (TF)

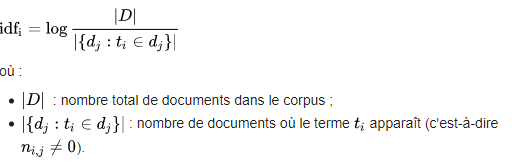
La fréquence « brute » d'un terme est simplement le nombre d'occurrences de ce terme dans le document considéré (on parle de « fréquence » par abus de langage). On peut choisir cette fréquence brute pour exprimer la fréquence d'un terme.

Des variantes ont été proposées. Un choix plus simple, dit « binaire », est de mettre 1 si le terme apparaît dans le document et 0 sinon. À l'opposé, on peut normaliser logarithmiquement la fréquence brute pour amortir les écarts. Une normalisation courante pour prendre en compte la longueur du document est de normaliser par la fréquence brute maximale du document.



## Fréquence inverse de document (IDF)

La fréquence inverse de document (*inverse document frequency*) est une mesure de l'importance du terme dans l'ensemble du corpus. Dans le schéma TF-IDF, elle vise à donner un poids plus important aux termes les moins fréquents, considérés comme plus discriminants. Elle consiste à calculer le logarithme (en base 10 ou en base 2) de l'inverse de la proportion de documents du corpus qui contiennent le terme :



## Calcul de TF-IDF

Finalement, le poids s'obtient en multipliant les deux mesures :



## Exemple d’utilisation de TF-IDF

|  |  |  |
| --- | --- | --- |
| Document 1 | Document 2 | Document 3 |
| Son nom est célébré par le bocage **qui** frémit, et par le ruisseau **qui** murmure, les vents l’emportent jusqu’à l’arc céleste, l’arc de grâce et de consolation que sa main tendit dans les nuages. | À peine distinguait-on deux buts à l’extrémité de la carrière : des chênes ombrageaient l’un, autour de l’autre des palmiers se dessinaient dans l’éclat du soir. | Ah ! le beau temps de mes travaux poétiques ! les beaux jours que j’ai passés près de toi ! Les premiers, inépuisables de joie, de paix et de liberté ; les derniers, empreints d’une mélancolie **qui** eut bien aussi ses charmes. |

L'exemple porte sur le document 1 (soit {\displaystyle d\_{1}} d 1) et le terme analysé est « qui » (soit {\displaystyle t\_{1}}t 1 = qui). La ponctuation et l'apostrophe sont ignorées.

Détails du calcul : la plupart des termes apparaissent une fois (21 termes), l apparaît 3 fois et arc, de, et, le, les, par et qui (2 fois). Le dénominateur est donc

3 + 7\*2 + 21 = 38. Cette somme est le nombre de mots dans le document.

**Calculons TF :**

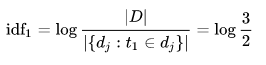
TF(t) = Nombre d'apparition du terme t dans le document / Nombre total de termes dans le document



Détails du calcul : la plupart des termes apparaissent une fois (21 termes), *arc*, *de*, *et*, *le*, *les*, *par* et *qui* apparaissent 2 fois (7 termes) et *l* apparaît 3 fois (1 terme). Le dénominateur est donc 21\*1 + 7\*2 + 1\*3 = 38. Cette somme correspond au nombre de mots dans le document.

**Calculons IDF :**

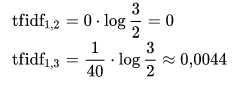
Le terme « qui » n'apparaît pas dans le deuxième document. Ainsi :



On obtient :



Pour les autres documents :



Le document 1 se trouve être le plus pertinent au vu du résultat obtenu.{\displaystyle \mathrm {idf\_{i}} =\log {\frac {|D|}{|\{d\_{j}:t\_{i}\in d\_{j}\}|}}}

# Quelques outils du TALN

Pour s’initier au traitement automatique du langage naturel, il existe des **outils et aides pratiques en ligne**. Mais quel est l’outil le plus adapté pour vous ? Cela va dépendre de la langue et de la méthode TALN que vous souhaitez utiliser. Parmi les **outils open-source** les plus connus, on trouve :

* Le [**Natural Language Toolkit**](http://www.nltk.org/) est un ensemble d’outils TALN en langage Python. L’outil propose un accès à plus de 100 corpus de textes, parmi lesquels des textes en anglais, portugais, polonais, néerlandais, catalan et basque. De plus, le kit peut effectuer le traitement de différents textes, comme l’étiquetage morpho-syntaxique, l’arbre syntaxique, la segmentation (tokenization en anglais, ce qui constitue souvent la première étape du TALN) et la synthèse de texte. Le kit d’outils TALN comporte également une introduction à la programmation et une documentation détaillée. Il est ainsi bien adapté aux étudiants, doctorants et chercheurs.
* [**Stanford NLP Group Software**](http://nlp.stanford.edu/software/): l’un des groupes de recherche les plus importants dans le domaine du traitement automatique du langage naturel**.**De nombreux outils sont proposés. Ils permettent de définir la forme de base des mots (segmentation en unité), la fonction des mots (étiquetage morpho-syntaxique) et la structure des phrases (arbre syntaxique). De plus, il existe des outils pour les process compliqués comme le [**deep learning**](https://www.ionos.fr/digitalguide/web-marketing/search-engine-marketing/deep-learning/) pour lequel le contexte de la phrase est pris en compte. [**Stanford CoreNLP**](http://stanfordnlp.github.io/CoreNLP/) présente la plupart des fonctions de base. L’ensemble des programmes du Standford NLP sont écrits en langage Java et sont disponibles en français, anglais, allemand, espagnol et chinois.
* Pour le traitement du langage vocal, on trouve le kit CSLU. Ces outils présentent, entre autres fonctions, la **reconnaissance vocale** et la retransmission de textes à l’oral (par voix de synthèse). Ils comprennent également des outils d’entraînement avec lesquels les enfants peuvent apprendre des nouveaux mots de vocabulaire, et avec lesquels les personnes sourdes peuvent s’exercer à parler. Les outils sont ainsi adaptés aux jeunes élèves, aux étudiants, aux chercheurs et bien sûr à toute autre personne intéressée.
* [**Visualtext**](http://www.textanalysis.com/Products/products.html) est un ensemble d’outils écrits dans un langage de programmation propre au TALN : le **langage NLP++.** Ce langage de programmation a surtout été développé pour ce que l’on appelle les analystes Deep Text. Visualtext sert surtout à **extraire des informations** depuis une grande quantité de textes. Il permet par exemple de **résumer des textes longs** mais aussi de regrouper des évènements sur un thème précis à partir de plusieurs sites Web et de créer un aperçu. Visualtext peut être utilisé gratuitement pour des fins non-commerciales.

# Conclusion

Les mathématiques ont permis d’avoir un progrès fulgurant dans le traitement du langage naturel qui est un domaine d’expertise compliqué nécessitant le traitement d’une large batterie d’informations par les ordinateurs. La marge de progrès à faire pour ce domaine de est encore grande ; cependant l’ironie, le sarcasme et les métaphores humoristiques sont particulièrement difficiles à comprendre pour les ordinateurs, malgré de nombreuses recherches dans le domaine et un long travail de classification.

# Bibliographie

* <https://www.stat4decision.com/fr/traitement-langage-naturel-francais-tal-nlp/>
* <https://www.ionos.fr/digitalguide/web-marketing/vendre-sur-internet/le-traitement-automatique-du-langage-naturel-taln/>
* <https://www.inbenta.com/fr/technologie/ia-nlp/technologies-du-langage-naturel/>
* <https://fr.wikipedia.org/wiki/Traitement_automatique_des_langues#:~:text=TALN)%2C%20ou%20traitement%20automatique%20de,langue%20naturelle%20pour%20diverses%20applications>.
* <https://fr.qaz.wiki/wiki/Brown_Corpus>
* <https://fr.wikipedia.org/wiki/TF-IDF>
* <https://fr.wikipedia.org/wiki/Racinisation#D%C3%A9tail_de_l'algorithme_de_Porter[4>]

1. *Le Brown University Standard Corpus of Present-Day American English (ou sBrown Corpus) est une collection électronique d'échantillons de textes d'anglais américain compilés par Henry Kučera et W.Nelson Francis à l'Université Brown, dans le Rhode Island. , il s'agit d'un corpus de langue générale contenant 500 échantillons d'anglais, totalisant environ un million de mots, compilé à partir d'ouvrages publiés aux États-Unis en 1961* [↑](#footnote-ref-1)
2. *Le British National Corpus (BNC) est une collection de 100 millions de mots d'échantillons de langue écrite et parlée provenant d'un large éventail de sources, conçue pour représenter un large échantillon de l'anglais britannique de la fin du XXe siècle, à la fois parlé et écrit. La dernière édition est l'édition BNC XML, publiée en 2007.* [↑](#footnote-ref-2)