  

**Parcours : MASTER 1 INFORMATIQUE POUR LES SCIENCES IPS**

**HMIN201 : TER**

**MISE EN CORRESPENDANCE DE VOCABULAIRES DU DOMAINE DE LA MUSIQUE**

*Travail de recherche afin d’aligner des vocabulaires musicaux à travers un algorithme Python*

**L’équipe de recherche : BENDJOUDI Nassim,**

**DJERROUD Ilyes,**

**KEBLOUTI Wissam Loubna**

**Encadré par :** Mr. Konstantin TODOROV

2017/2018

Remerciement :

Nous remercions **Mr Konstantin** [**TODOROV**](https://moodle.umontpellier.fr/user/view.php?id=63756&amp;course=1), encadreur et initiateur du projet, malgré sa charge de travail importante, il nous a encadrés tout au long de notre projet. Nous tenant également à le remercie pour sa con- fiance, son aide et son encouragement.

Aussi, nos sincères gratitudes à **l’équipe de laboratoire de LIRMM** Montpellier, qui nous ont donné l’opportunité de collaborer avec eux afin d’acquérir une expérience dans le travail en multitâches et en équipe.

Nous remercions également **Mr Pierre POMPIDOR** pour l’aide qu’il a pu nous apporter avec ses remarques constructives.

Enfin, nos remerciements les plus sincères sont adressés à tous nos proches et amis, qui nous ont toujours soutenus et encouragés au cours de la réalisation de ce projet. Merci à tous et à toutes.

Ici table des matière …….

INTRODUCTION :

Le Web sémantique est un projet du W3C qui utilise des métadonnées pour permettre aux ordinateurs de comprendre l'information sur Internet. Cela permettrait aux ordinateurs de faire plus de travail pour trouver, partager et combiner des informations sur Internet.

C’est est une idée de l'inventeur du World Wide Web, Tim Berners Lee. Il veut rendre le web plus intuitif sur la façon de répondre aux besoins d'un utilisateur. La sémantique des informations et des services est définie dans le Web Ontology Language et dans les schémas RDF. Ceux-ci sont utilisés pour donner une description formelle des concepts, des termes et des relations dans un domaine donné de la connaissance.

« J'ai un rêve pour le Web dans lequel les ordinateurs deviennent capables d'analyser toutes les données, le contenu, les liens et les transactions entre les personnes et les machines. Un 'Web sémantique', qui devrait rendre cela possible, n'a pas encore vu le jour, mais quand il le sera, les mécanismes quotidiens du commerce, de la bureaucratie et de notre vie quotidienne seront gérés par des machines parlant aux machines. » *Tim Berners Lee*

Dans le Chapitre I, nous nous proposons de faire l’état de l’art des procédés et des méthodes d’alignement des termes d’une manière générale et plus particulièrement par le biais du langage python, nous étudions le rôle de ses méthodes et nous essayons par la suite d’en choisir la plus pertinente pour un bon résultat.

Le Chapitre II est consacré au projet DOREMUS, ses partenaires, ses idées, son objectif.

Enfin, nous abordons dans le Chapitre III les différentes taches d’améliorations apportées sur l’algorithme, les difficultés présentées, les algorithme implémentés, notamment le choix du langage de la programmation, le résultat final de l’alignement.

**Contexte du projet :**

Ce projet rentre dans le cadre du master 1 Informatique pour les sciences (IPS), plus précisément dans L’unité d’enseignement (HMIN201) intitulée **Travail d’étude et recherche (TER),** qui permet deformer les étudiants, par la pratique, la recherche et la conception ainsi que l’élaboration et la coopération.

# Synthèse bibliographique :

## Les mesures de similarités :

La mesure de similarité est un paramètre important dans l'exploration de texte. La mesure de la similarité nous permet de trouver la similitude existante entre les documents texte. L'approche terme-fréquence (TF) a été utilisée pour la représentation d'un document texte où le document est considéré comme un vecteur avec un ensemble de termes apparaissant dans le document et leurs fréquences. Il y a deux problèmes de synonymie et de polysémie, associés à l'approche. Les synonymes sont des termes ayant la même signification entre eux et deux termes qui peuvent être échangés dans un contexte donné sont dits synonymes par rapport au contexte. Par exemple, une voiture, une automobile et un véhicule peuvent être considérés comme synonymes dans un document particulier. La polysémie renvoie à l'ambiguïté lexicale lorsque le même mot est utilisé pour exprimer deux ou plusieurs significations dans des contextes différents. Pour distinguer correctement les occurrences d'un même terme dans des contextes différents avec des significations différentes, une approche riche en connaissances utilisant le traitement du langage naturel est une nécessité.

De nombreux travaux ont cherché à s’appuyer sur des textes pour construire des ontologies (Cimiano, 2006). Deux grandes méthodes ont été proposées. La première consiste à regrouper les mots du corpus en classes sémantiques, avec l’idée que ces classes sémantiques représentent les concepts centraux du domaine reflétés par le corpus. Cette analyse distributionnelle « à la Harris » est une technique exploitée depuis longtemps pour la construction de classes sémantiques de mots. Elle consiste à rapprocher les mots sur la base de contextes qu’ils partagent, en faisant l’hypothèse que les mots les plus proches sémantiquement sont employés de manière similaire et tendent à apparaître dans les mêmes contextes. Malheureusement les classes sémantiques obtenues sont généralement assez bruitées et difficiles à traduire directement en concepts.

L’autre approche repose sur l’analyse terminologique du corpus, la liste des termes du domaine donnant elle-même un aperçu de son « vocabulaire conceptuel ». Travailler sur des termes extraits par un extracteur de termes plutôt que sur des mots se justifie dans la mesure où les mots composant un corpus ne sont pas tous pertinents (mots vides, mots très fréquents, etc.) et où les termes complexes perdent de leur intérêt si on les découpe (par exemple, dans le domaine de l’astronomie, le terme « trou noir » a beaucoup plus de sens que les mots « trou » et « noir » considérés séparément). Cette deuxième approche a été mise en œuvre dans des outils d’acquisition d’ontologies comme TERMINAE (Szulman et al., 2002) mais elle n’offre qu’une aide limitée à l’ingénieur de la connaissance qui doit souvent analyser et explorer de longues listes de termes pour identifier ceux qui renvoient à des concepts clefs du domaine. Pour la première approche Curran (2004) décompose les mesures de similarité existantes en deux composantes : les mesures et les poids. En effet, les approches de similarité distributionnelle sont fondées sur l’hypothèse que la similarité sémantique est reflétée par la similarité des contextes. Or, dans un contexte, tous les mots n’ont pas la même importance. Il parait donc judicieux de pondérer leur impact avant l’application d’une mesure de similarité.

### String matching :

En informatique, les algorithmes de String Matching, parfois appelés algorithmes de correspondance de chaînes, sont une classe importante d'algorithmes de chaînes qui tentent de trouver un endroit où une ou plusieurs chaînes (également appelées modèles) se trouvent dans une chaîne ou un texte plus volumineux.

Soit Σ un alphabet (ensemble fini). L'exemple le plus basique de recherche de chaînes est celui où le motif et le texte recherché sont des tableaux d'éléments de Σ. Le Σ peut être un alphabet humain habituel (par exemple, les lettres A à Z dans l'alphabet latin). D'autres applications peuvent utiliser l'alphabet binaire (Σ = {0,1}) ou l'alphabet d'ADN (Σ = {A, C, G, T}) en bio-informatique.

### Jaro-Winkler :

La distance de Jaro-Winkler mesure la similarité entre deux [chaînes de caractères](http://dictionnaire.sensagent.leparisien.fr/Chaîne%20de%20caractères/fr-fr/). Il s'agit d'une variante proposée en 1999 par [William E. Winkler](http://fr.wikipedia.org/w/index.php?title=William_E._Winkler&action=edit&redlink=1), découlant de la distance de Jaro (1989, [Matthew A. Jaro](http://fr.wikipedia.org/w/index.php?title=Matthew_A._Jaro&action=edit&redlink=1)) qui est principalement utilisée dans la détection de doublons.

Plus la distance de Jaro-Winkler entre deux chaînes est élevée, plus elles sont similaires. Cette mesure est particulièrement adaptée au traitement de chaînes courtes comme des noms ou des mots de passe. Le résultat est normalisé de façon à avoir une mesure entre 0 et 1, le zéro représentant l'absence de similarité.

#### Jaro :

**La distance de Jaro entre chaînes s_1 et s_2 est définie par :**

d_j = \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right)

où :

* |s_i| Est la longueur de la chaîne de caractères s_i ;
* m Est le nombre de *caractères correspondants* (voir ci-dessous) ;
* t Est le nombre de *transpositions* (voir ci-dessous).

Deux caractères identiques de s_1 et de s_2 sont considérés comme *correspondants* si leur éloignement (i.e. la différence entre leurs positions dans leurs chaînes respectives) ne dépasse pas :

\left\lfloor\frac{\max(|s_1|,|s_2|)}{2}\right\rfloor-1.

Le nombre de transpositions est obtenu en comparant le i -ème caractère *correspondant* de s_1 avec le i -ème caractère *correspondant* de s_2. Le nombre de fois où ces caractères sont différents, divisé par deux, donne le nombre de *transposition*

#### Distance de Jaro-Winkler :

La méthode introduite par Winkler utilise un *coefficient de préfixe* p qui favorise les chaînes commençant par un préfixe de longueur \ell(avec \ell \le 4). En considérant deux chaînes s_1 et s_2, leur distance de Jaro-Winkler d_w est :

d_w = d_j + (\ell p (1 - d_j))

Où :

* d_j Est la distance de Jaro entre s_1 et s_2
* \ell Est la longueur du préfixe commun (maximum 4 caractères)
* p Est un coefficient qui permet de favoriser les chaînes avec un préfixe commun. Winkler propose pour valeur p = 0.1
* **Exemples**

Soit deux chaînes s_1 *MARTHA* et s_2  *MARHTA*. La table de correspondance est :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | M | A | R | T | H | A |
| M | 1 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | 1 | 0 | 0 | 0 | 0 |
| R | 0 | 0 | 1 | 0 | 0 | 0 |
| H | 0 | 0 | 0 | 0 | **1** | 0 |
| T | 0 | 0 | 0 | **1** | 0 | 0 |
| A | 0 | 0 | 0 | 0 | 0 | 1 |

* m = 6 (Nombre de 1 dans la table)
* |s_1| = 6
* |s_2| = 6
* Les caractères *correspondants* sont {M, A,R,T,H,A} pour s_1 et {M,A,R,H,T,A} pour s_2. En considérant ces ensembles ordonnés, on a donc 2 couples (T/H et H/T) de caractères *correspondants* différents, soit deux demi-transpositions. D'où t = \frac{2}{2} = 1

La distance de Jaro est :

d_j = \frac{1}{3}\left(\frac{6}{6} + \frac{6}{6} + \frac{6-1}{6}\right) = 0.944

La distance de Jaro-Winkler avec p = 0.1 avec un préfixe de longueur \ell=3 devient

d_w = 0.944 + (3 * 0.1 (1 - 0.944)) = 0.961~

### Levenshtein :

Levenshtein distance (LD) est une mesure de la similarité entre deux chaînes, que nous appellerons la chaîne source (s) et la chaîne cible (t). La distance est le nombre de suppressions, d'insertions ou de substitutions nécessaires pour transformer s en t. Par exemple,

Si (s) est "test" et (t) est "test", alors LD (s, t) = 0, car aucune transformation n'est nécessaire. Les chaînes sont déjà identiques.

Si (s) est "test" et (t) est "tente", alors LD (s, t) = 1, car une substitution (change "s" en "n") est suffisante pour transformer (s) en (t).

* Plus la distance de Levenshtein est grande, plus les chaines sont différentes.

Levenshtein distance est nommé d'après le scientifique russe *Vladimir Levenshtein*, qui a conçu l'algorithme en 1965. Si vous ne pouvez pas épeler ou prononcer Levenshtein, la métrique est aussi parfois appelée distance d'édition.

L'algorithme de distance de Levenshtein a été utilisé dans :

* Vérification orthographique
* Reconnaissance de la parole
* Analyse de l'ADN
* Détection de plagiat

De telles distances ont été étendues pour s'appliquer également à la comparaison phonétique, à l'analyse lexicale, aux comparaisons grammaticales et autres.

Exemple : « kitten » et « sitting » sont à distance 3.

kitten → sitten (substitution de "s" à "k")

sitten → sittin (substitution de "i" à "e")

sittin → sitting (insertion de "g" à la fin).

#### L’algorithme :

L’algorithme ci-dessous, dû à Wagner et Fischer (1974), permet de calculer la distance de Levenshtein entre deux chaînes de caractères courtes. Cet algorithme est un exemple de [programmation dynamique](https://fr.wikipedia.org/wiki/Programmation_dynamique) (solution de type du bas en haut), qui utilise une matrice de dimension {\displaystyle (n+1)\times (m+1)}où *n* et *m* sont les dimensions des deux chaînes de caractères. Dans le [pseudo-code](https://fr.wikipedia.org/wiki/Pseudo-code) suivant, la chaîne *chaine1* est de longueur *longueurChaine1* et *chaine2*, de longueur *longueurChaine2*. Cet algorithme renvoie un entier positif ou nul. Par définition d'une distance au sens mathématique du terme, l'algorithme renvoie 0 si et seulement si les deux chaînes sont égales. Pour deux chaînes de longueurs données, la valeur maximale de la distance est la longueur de la chaîne la plus longue.

entier DistanceDeLevenshtein(caractere chaine1[1..longueurChaine1],

caractere chaine2[1..longueurChaine2])

*// d est un tableau de longueurChaine1+1 rangées et longueurChaine2+1 colonnes*

*// d est indexé à partir de 0, les chaînes à partir de 1*

déclarer entier d[0..longueurChaine1, 0..longueurChaine2]

*// i et j itèrent sur chaine1 et chaine2*

déclarer entier i, j, coûtSubstitution

pour i de 0 à longueurChaine1

d[i, 0] := i

pour j de 0 à longueurChaine2

d[0, j] := j

pour i de 1 à longueurChaine1

pour j de 1 à longueurChaine2

si chaine1[i] = chaine2[j] alors coûtSubstitution := 0

sinon coûtSubstitution := 1

d[i, j] := minimum(

d[i-1, j ] + 1, *// effacement du nouveau caractère de chaine1*

d[i, j-1] + 1, *// insertion dans chaine1 du nouveau caractère de chaine2*

d[i-1, j-1] + coûtSubstitution *// substitution*

)

renvoyer d[longueurChaine1, longueurChaine2]

L’[invariant](https://fr.wikipedia.org/wiki/Invariant) est qu’on peut transformer le segment initial chaine1[1..i] en chaine2[1..j] en utilisant un nombre minimal de d[i, j] opérations. L’algorithme achevé, la solution est contenue dans la dernière position à droite de la rangée du bas de la matrice.

#### Fonctionnement de l’algorithme :

Soit n la longueur de la chaîne1 (ici n=5)  
Soit m la longueur de la chaîne2 (ici m=6)

Si n=0 alors retourner d=m et quitter  
Si m=0 alors retourner d=n et quitter

Construire une matrice M de n+1 lignes et m+1 colonnes.  
Initialiser la première ligne par la matrice ligne [ 0,1,….., m-1, m] et la première colonne par la matrice colonne [ 0,1,….., n-1, n]

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **C** | **H** | **I** | **E** | **N** | **S** |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| **N** | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **I** | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| **C** | 3 | 0 | 0 | 0 | 0 | 0 | 0 |
| **H** | 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| **E** | 5 | 0 | 0 | 0 | 0 | 0 | 0 |

Soit Cout(i, j)=0 si A(i)=B(j) et Cout(i, j)=1 si A(i)!=B(j)

On a donc ici la matrice Cout:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **C** | **H** | **I** | **E** | **N** | **S** |
| **N** | 1 | 1 | 1 | 1 | 0 | 1 |
| **I** | 1 | 1 | 0 | 1 | 1 | 1 |
| **C** | 0 | 1 | 1 | 1 | 1 | 1 |
| **H** | 1 | 0 | 1 | 1 | 1 | 1 |
| **E** | 1 | 1 | 1 | 0 | 1 | 1 |

On remplit ensuite la matrice M en utilisant la règle suivante  M[i, j] est égale au minimum entre les éléments suivants :

* L’élément directement au-dessus et on ajoute 1: M[i-1, j] + 1. (effacement)
* L’élément directement avant et on ajoute 1: M[i, j-1] + 1. (insertion)
* Le diagonal précédent plus le coût: M[i-1, j-1] + Cout(i-1, j-1). (substitution)

Attention ! Il s'agit de Cout(i-1, j-1) et non de Cout(i, j) car la matrice Cout est moins grande que la matrice M, ce qui entraîne un décalage.

Dans notre cas, le remplissage de la première ligne donne alors:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **C** | **H** | **I** | **E** | **N** | **S** |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| **N** | 1 | 1 | 2 | 3 | 4 | 4 | 5 |
| **I** | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| **C** | 3 | 0 | 0 | 0 | 0 | 0 | 0 |
| **H** | 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| **E** | 5 | 0 | 0 | 0 | 0 | 0 | 0 |

Nous réitérons cette opération jusqu'à remplir la matrice :

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **C** | **H** | **I** | **E** | **N** | **S** |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| **N** | **1** | 1 | 2 | 3 | 4 | 4 | 5 |
| **I** | **2** | 2 | 2 | 2 | 3 | 4 | 5 |
| **C** | 3 | **2** | 3 | 3 | 3 | 4 | 5 |
| **H** | 4 | 3 | **2** | **3** | 4 | 4 | 5 |
| **E** | 5 | 4 | 3 | 3 | **3** | **4** | **5** |

La distance de Levenshtein entre les chaînes 1 et 2 se retrouve en M[n, m].

Ici, on retrouve bien les 5 opérations trouvées de manière intuitive, la dernière matrice fournit aussi explicitement une des suites d'opérations permettant de passer d'une chaîne de caractères à l'autre (Il existe 3 suites possibles

# DOREMUS - DOing REusable MUSical data:

La musique est partout, jouée, enregistrée, diffusée, écoutée, vendue, offerte, louée. Les fichiers de musique enregistrée sont partout, stockés, streamés, échangés, vendus. Mais rien n’est plus difficile aujourd’hui que de trouver sur le web l’histoire d’une œuvre musicale, son compositeur, son origine culturelle, ses paroliers, ses influences, ses reprises, ses interprétations... Pourtant ces connaissances existent et sont décrites finement dans les systèmes d’information de quelques institutions culturelles et média dans le monde.

## Objectifs, idées, partenaires :

Le premier objectif du projet DOREMUS est de permettre aux institutions culturelles, aux éditeurs et distributeurs, ainsi qu’aux communautés de passionnés de disposer de modèles de connaissances communs (ontologies), de référentiels partagés et multilingues ainsi que de méthodes pour publier, partager, connecter, contextualiser, enrichir les catalogues d’œuvres et d’événements musicaux dans le web des données.   
Le projet s’appuie sur l’expertise de catalogage des œuvres musicales de Radio France, de la BnF, de la Cité de la musique, du projet collaboratif Discotheka de la société Meaning Engines et sur un corpus de modèles préexistants, en particulier FRBRoo. Les extensions et spécialisations d’ontologies et les vocabulaires qui résulteront du projet devront permettre de décrire les œuvres et événements musicaux, de la musique traditionnelle à la musique classique, leurs relations avec les créateurs, leur environnement culturel, les événements liés à leur création, leurs interprétations, leur fonction sociale, les instruments, les genres musicaux... tout en s’inscrivant dans la continuité des catalogues d’œuvres existants.   
  
 La description des œuvres musicales ne prend de sens que si elle est mise à disposition des publics comme élément d’éducation et de culture, utilisée comme outils facilitant la programmation d’émissions et d’événements musicaux, permettant de suggérer de nouvelles interprétations ou servant de source d’inspiration pour d’autres œuvres.   
  
 Le second objectif du projet est en effet de construire un outil d’assistance à la sélection d’œuvres musicales. L’outil doit pouvoir suggérer une programmation musicale originale pour une radio spécialisée, choisir des œuvres ou des interprétations particulières pour illustrer la biographie d’un artiste, une période historique, une culture ou un genre musical. Les publics visés par la Cité de la musique, Radio France, la BnF ou le projet Discotheka sont très diversifiés. Ils vont de l’enfant construisant sa culture musicale au mélomane averti en passant au par le curieux découvrant l’univers de la musique à travers son identité culturelle. Le projet dispose de ressources très riches pour mettre au point ce service : historique des programmations musicales d’émissions de radio, historiques de concerts, œuvres sélectionnées pour des disques, playlists.   
  
 Le troisième objectif du projet est de construire et valider les outils pédagogiques qui permettront le déploiement des standards, référentiels et technologies dans les institutions culturelles. Le consortium se donne un objectif fort de formation et de dissémination de compétences avec de nombreuses communications dans les conférences internationales, une interaction constante avec les groupes de travail internationaux travaillant sur ces ontologies et référentiels, des échanges avec Europeana, la Library of Congress et la BBC qui soutiennent le projet. 

Le projet intègre une équipe de trois doctorants. Le projet produira des résultats concrets, réutilisables et produits pour une diffusion et utilisation internationale : modèles et référentiels pour les œuvres musicales, jeux de données, méthodes et outils pour la connexion des jeux de données, système de recommandation adapté aux œuvres musicales, typologie des usages et publics pour ces outils, documents pédagogiques. L’ensemble des résultats sera publié avec des licences ouvertes.

## Partenaires :

* BNF BIBLIOTHEQUE NATIONALE DE France
* MEANING ENGINES DISCOTHEKA SARL
* EURECOM EURECOM
* GERIICO GROUPE D'ETUDE ET DE RECHERCHE INTERDISCIPLINAIRE SUR L'INFORMATION ET LA COMMUNICATION
* CITE DE LA MUSIQUE LA CITE DE LA MUSIQUE
* UM2-LIRMM LABORATOIRE D'INFORMATIQUE ROBOTIQUE ET MICROELECTRONIQUE DE MONTPELLIER
* OUROUK OUROUK
* RADIO FRANCE SOCIETE NATIONALE DE RADIODIFFUSION - RADIO France
* Aide de l'ANR 993 249 euros
* Début et durée du projet scientifique octobre 2014 - 36 mois