POLYTECH TOURS
64 Avenue Jean Portalis
37200 TOURS, France
Tél +33 (0)2 47 36 14 14
www.polytech.univ-tours.fr

Polytech Tours

Rapport de stage 2021-2022

Projet StationTV

CONFIDENTIEL

Entreprise :

Laboratoire LIFAT

60 Rue du plat d'Etain, 37000, Tours, France

Tuteur Entreprise :

DELALANDRE Mathieu

Maître de conférences

Entreprise :

CHAN Pierre-Louis

Promo DI4 2019-2022

Tuteur Académique :

MONMARCHE Nicolas

Enseignant Chercheur

Rédigé par CHAN Pierre-Louis
17/08/2021

Table des matières

0.1 – Le projet, décomposé 2
0.2 – Le stage dans ce projet 4
I.1 – Contexte 5
I.2 – XMLTV : C'est quoi ? 5
I.3 – XMLTV : Parsing 7
II.1 – Contexte 9
II.2 – Analyse générale 9
II.3 – Extractions 12
III.1 – Contexte 15
III.2 – Objectif et approche du problème 15
III.3 – Recherche de N 18
III.4 – Résultats 21
III.5 – Et les images ? 22
IV.1 – Contexte 23
IV.1 – Mise en place 24
IV.2 – Aller plus loin 25
IV.3 – Résultat de l'augmentation 25
V.1 – Contexte 26
V.2 – Premières extractions 26
V.3 – Extraction des données du mois 2 26
Conclusion vis-à-vis du projet 28
Conclusion personnelle 28
Sites utiles 29
Annexes 30
Court Résumé 36
Mots-Clés 36

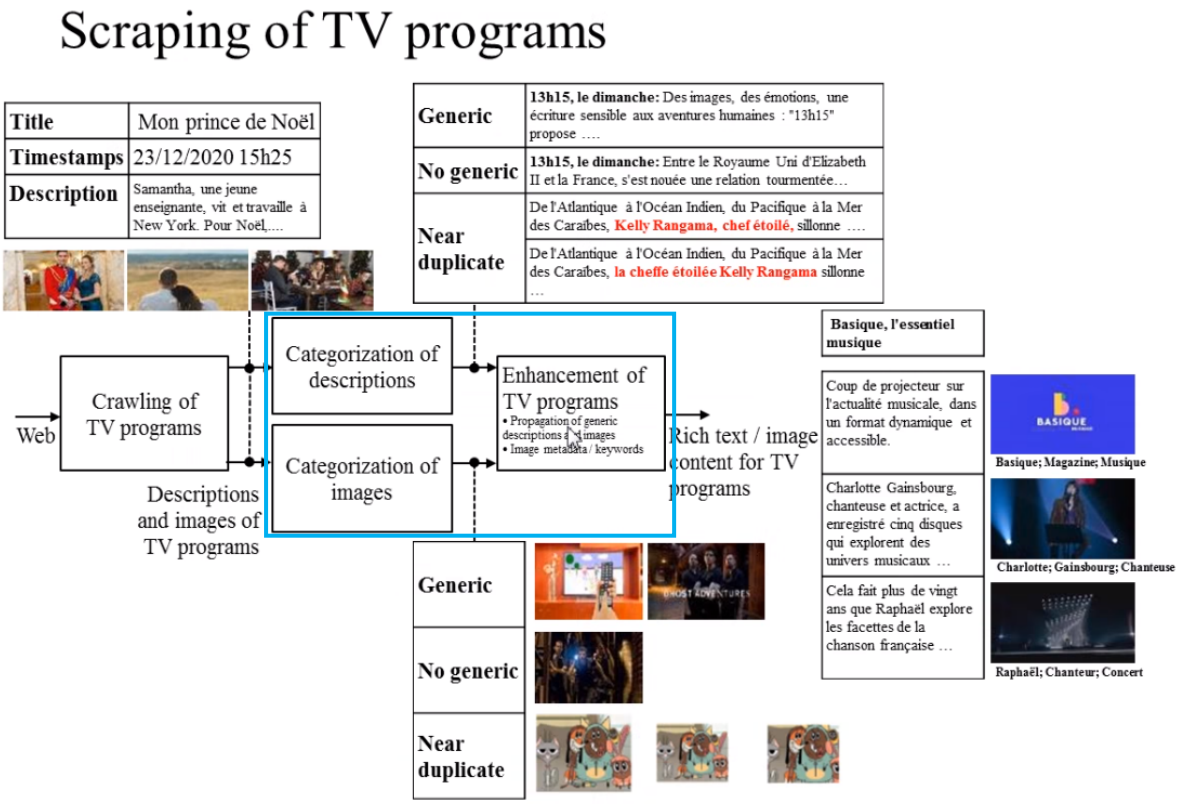
Introduction au Projet StationTV

0.1 – Le projet, décomposé

Le projet StationTV, comme rapidement présenté dans le court résumé de la 4e de couverture, peut se décomposer en plusieurs parties, afin d'expliciter plus clairement chacune d'entre elles. Le pipeline général du projet dans lequel se situe le stage commence tout d'abord par une étape de crawling, afin de récupérer les données de programmes TV, sous format XMLTV particulièrement. L'on va donc parcourir différents sites web afin de récupérer des informations de cette manière.

Ce projet est réalisé avec l'aide de l'équipe StationTV, composée de Célie, Antoine, moi-même, étant trois stagiaires sur ce projet, ainsi que Hao Le, F. Rayar et M. Delalandre, nos contacts et encadrants.

Représentation schématique avec légendes du pipeline



Après cette étape de collection d'informations, l'on se retrouve avec des données non traitées, qu'il faut catégoriser au niveau des descriptions, mais aussi au niveau des images. L'on catégorise selon la ressemblance et l'utilisation des images, ou la ressemblance des résumés. Avec ces informations, l'on peut ensuite utiliser ces catégorisations pour enrichir les données d'origine (par exemple, ajouter un résumé générique là où il n'y en a pas), afin d'avoir un taux de couverture maximum, ou minimiser les entrées sans informations. Avec ces données enrichies, l'on peut ensuite s'en servir dans les divers services numériques du projet StationTV.

0.2 – Le stage dans ce projet

Dans ce projet, ma partie consiste particulièrement en l'analyse, la catégorisation, et l'augmentation des données, à partir d'un fichier XMLTV fourni et déjà récupéré. D'autres stagiaires étaient présents durant ma période de stage, pendant laquelle j'ai pu interagir avec eux et leur fournir mes résultats, afin d'avancer dans la mise en place des services numériques. Je me chargeais donc de la partie « back-end » du projet. Sur le schéma, l'on peut retrouver ma partie encadrée en bleu.

Partie I : Parsing et XMLTV

I.1 – Contexte

La première partie principale de ce stage consiste en une courte partie de parsing, durant laquelle l'on m'a fourni des données sous un format particulier, le XMLTV. Les données ensuite extraites seraient utilisées par d'autres parties du projet, notamment par deux autres stagiaires. Pour cette raison, l'étape de lecture et d'enrichissement du fichier xml apparaissait comme critique pour la bonne conduite de l'ensemble du projet.

I.2 – XMLTV : C'est quoi ?

Le format XMLTV est basiquement un format spécial dérivé des fichiers XML. Il s'agit donc d'un fichier XML suivant des balises et une syntaxe bien spécifique. Pour faire court, il est utilisé afin de décrire le flux TV de diverses chaînes, du point de vue de l'utilisateur. Chaque programme est marqué de balises **<programme>**, contenant diverses informations à son sujet, comme les dates et heures de début/fin du programme, le titre, le langage, les mots-clés, la description, etc… Il est aussi possible d'ajouter des balises **<channel>** afin de décrire en détail les chaînes contenues, mais ces balises n'étaient pas présentes dans le fichier à étudier lors de ce projet.

Un fichier XMLTV contient généralement un grand nombre de programmes représentés comme ci-dessous (page suivante), tous avec une quantité variée d'informations. Généralement, il est possible de décrire le contenu d'une chaîne de télévision (ou plusieurs) sur une certaine période de plusieurs jours à l'aide d'un seul fichier XMLTV.

Le fichier que l'on a étudié dans ce stage contient les données de la période du 1er janvier 2021 au 16 mai 2021. Il y a dans tout cela 141 824 entrées de programmes, représentant 7500 titres uniques. Le fichier fait 69.1 Mo sous son format XML.

*Exemple de balise **<programme>** sous le format XMLTV[2] :*

```
<programme start="20080715080000 -0600" stop="20080715090000 -0600"
channel="I10759.labs.zap2it.com">
    <title lang="en">Rachael Ray</title>
    <desc lang="en">Actresses Kim Raver, Brooke Shields and Lindsay Price
("Lipstick Jungle"); women in their 40s tell why they got breast implants; a
30-minute meal.</desc>
    <credits>
      <presenter>Rachael Ray</presenter>
    </credits>
    <date>20080306</date>
    <category lang="en">Talk</category>
    <category lang="en">Series</category>
    <episode-num system="dd_progid">EP00847333.0303</episode-num>
    <episode-num system="onscreen">2119</episode-num>
    <audio>
      <stereo>stereo</stereo>
    </audio>
    <previously-shown start="20080306000000" />
    <subtitles type="teletext" />
  </programme>
```

I.3 – XMLTV : Parsing

La première étape de mon stage fut donc de lire le contenu de ce fichier afin de pouvoir l'en extraire les données, dans le but de les analyser de différentes manières. Pour ce faire, j'ai mis en place plusieurs classes différentes, afin de pouvoir stocker les informations dans un format fixe, et simple d'utilisation.

Le fichier ayant été généré à partir d'une collecte de données HTML, ces dernières sont sujettes à l'apparition de code pour les caractères spéciaux (e.g. &euro pour €). Ces codes contiennent des caractères spéciaux (&, #, ;) non compatibles avec un formatage XML, un premier passage a dû être réalisé, afin de les retirer. Ensuite seulement, il fut possible de parse le code XML du fichier.

Les deux classes donc créées dans cet objectif sont les suivantes : **Channel** et **Program**.

*Format de la classe **Program** :*

```
Program Attributes:
    - program_title          #Titre du programme
    - program_channel        #Chaîne sur laquelle le programme est diffusé
    - program_date           #Date de diffusion du programme (sans heure)
    - program_description    #Description du programme (str)
    - program_previouslyShown #Date (et heure) de la dernière diffusion
    - program_start          #Date (et heure) de début de diffusion
    - program_stop           #Date (et heure) de fin de diffusion
    - program_icons          #Liste des icones
    - program_categories     #Liste des catégories
    - program_other          #Dictionnaire au format {clé : valeur}
    - program_shortName      #Nom condensé du programme
Une méthode spécifique à Program a été définie, afin de faciliter le traitement
des données avec cette classe, en plus des getters/setters, __str__ et
__repr__.
    - csvData : Cette méthode permet de directement renvoyer une chaîne de
      caractères contenant la majorité des informations du programme, sur
      une seule ligne, afin de générer des fichiers CSVs sur une liste de
      programmes.
```

*Format de la classe **Channel** :*

```
Channel Attributes:
    - channel_ID            #ID de la chaîne trouvé sur le XMLTV
    - channel_Icon          #Liste des icônes de la chaîne (URLs)
    - channel_displayNames  #Liste des noms affichés de la chaîne
    - channel_programList   #Liste des programmes diffusés sur la chaîne
Aucune méthode spéciale en dehors des getters/setters, __str__ et __repr__
n'ont été définies pour Channel.
```

En plus de ces deux classes, deux méthodes supplémentaires ont été définies, qui se révèleront utiles à plusieurs reprises à travers ce projet. Les deux vont de pair ensemble car il s'agit de deux méthodes de simplification de caractères et de chaîne de caractères respectivement. Nommées **charSimplification** et **stringSimplification**, la seconde est définie à l'aide de la première. Elles permettent de simplifier et transformer toute chaîne de caractères en retirant les espaces, convertissant certains caractères spéciaux en lettres simples, et retirant tout simplement certains caractères, en accord avec l'annexe « tableau de simplification ASCII ». Les attributs **program_shortName** ont été générés à l'aide de ces deux méthodes. Cela nous permet d'avoir des noms uniques pour chaque programme, dans le cas ou plusieurs « météo » seraient diffusées sur plusieurs chaînes différentes.

Pour ce qui est du code directement, la librarie ElementTree a été utilisée afin de simplifier le parsing du fichier XMLTV. Cela nous permet de réduire le code pour lire le fichier à une simple ligne, puis d'extraire les données du résultat de cette ligne. Car ElementTree nous renvoie une arborescence, qu'il nous suffit ensuite de parcourir.

Partie II : Analyse et extraction

II.1 – Contexte

La première partie de ce projet étant terminée, nous avons désormais un accès libre et rapide aux données en détail. Cela va nous permettre de commencer le gros du projet : l'analyse de ces dernières. L'on recherche diverses informations dans ces données, et l'on va en extraire quelques échantillons, afin de les fournir aux autres stagiaires sur le projet.

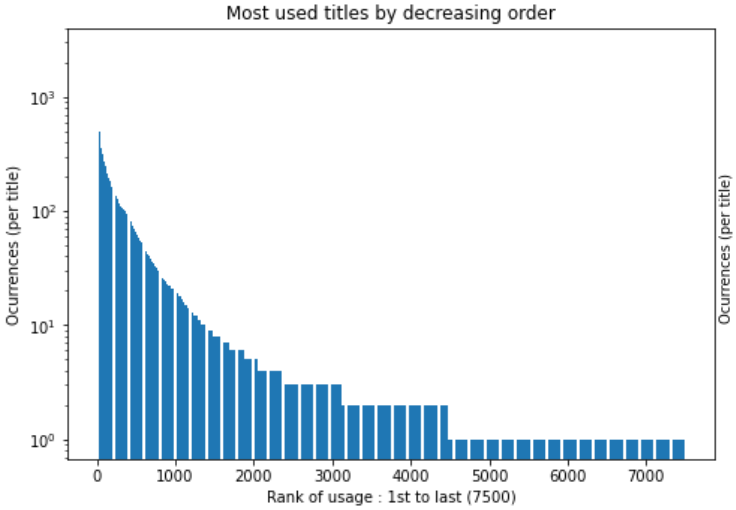
II.2 – Analyse générale

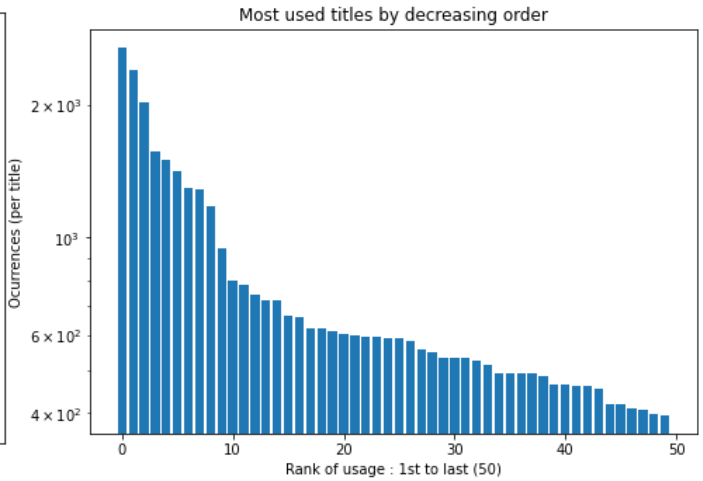
Pour commencer dans cette partie, j'ai dû réaliser quelques analyses basiques, comme trier les programmes selon leur fréquence d'apparition, afin d'extraire des programmes fréquents ou récurrents, et isoler les programmes uniques, ou diffusés qu'un faible nombre de fois.

La majeure partie de ces analyses ont été réalisées à l'aide de dictionnaires python, en usant de leur système de clé pour associer un nom de programme à un nombre d'occurrences. Une boucle passait dans la liste complète de programmes, et mettait à jour l'entrée de chaque programme en passant, afin de, par exemple, compter le nombre d'occurrences de ce programme. Cela nous permet aussi ensuite de retrouver la valeur de chaque programme assez facilement, tant que l'on connaît le nom et la chaîne de ce programme. L'on utilise « shortName + channelID » en tant que clé, afin d'éviter de compter ensemble plusieurs programmes similaires (notamment les météos) passant sur plusieurs chaînes.

Une première extraction des données a été donc réalisée afin de séparer les deux fréquences de programme, afin de pouvoir ensuite travailler sur des échantillons plus contrôlés. Quelques analyses supplémentaires afin d'obtenir des chiffres sur lesquels se référer ont été réalisées, notamment le nombre de programmes uniques parmi les fréquents ou non fréquents, le compte d'apparition de chaque programme, etc…

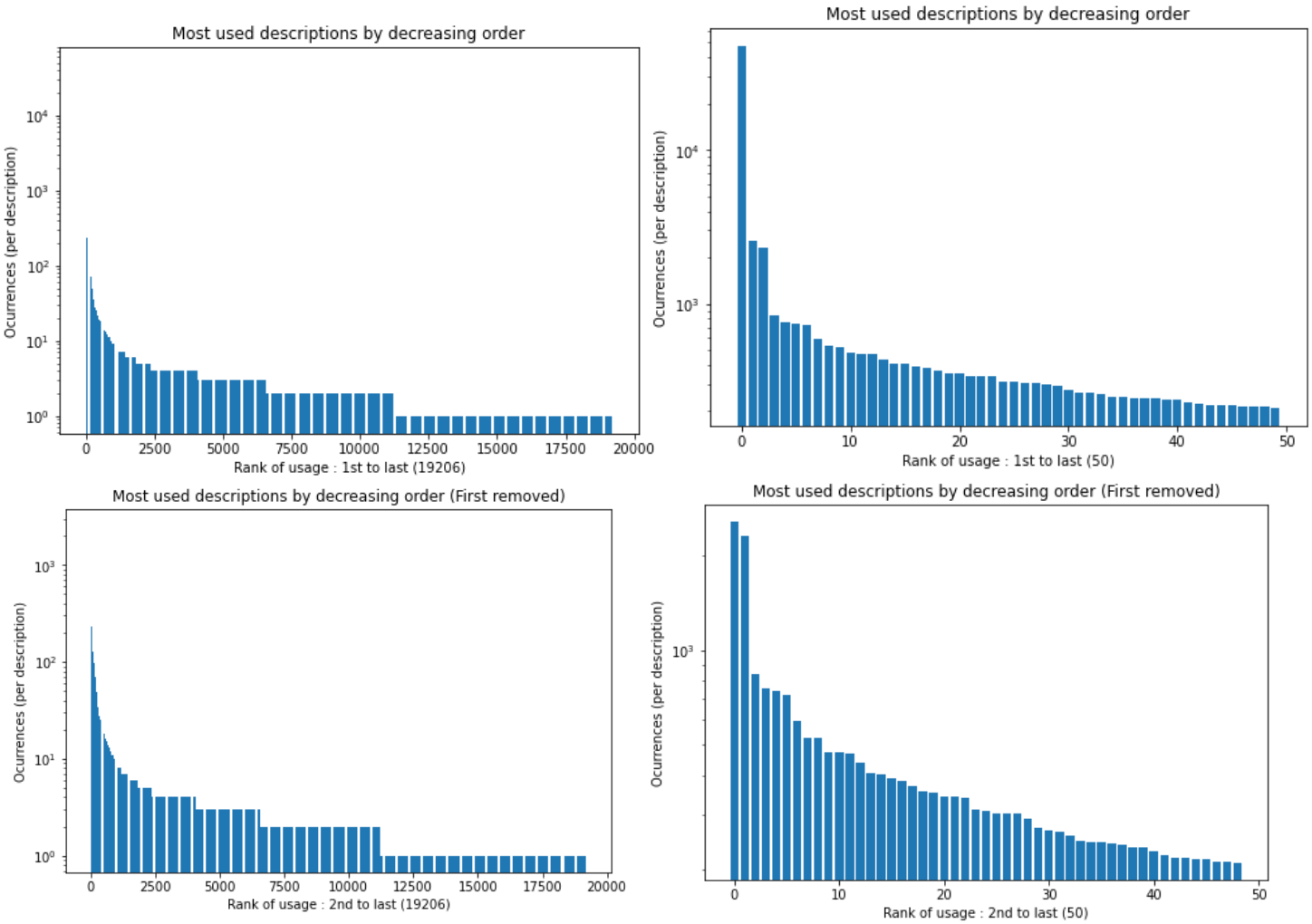
De cette première analyse, quelques graphiques ont été tracés afin de mieux se représenter les données (graphiques générés à l'aide de la librairie Python matplotlib) :





L'on remarque à l'aide de ces deux premiers graphiques, représentant respectivement les 7500 et 50 titres les plus utilisés dans le fichier XMLTV, sur une échelle logarithmique, que quelques titres sont énormément utilisés, et l'on retrouve énormément de titres uniques. On notera aussi, afin de donner une échelle, qu'il y a très exactement 141 824 instances de programmes dans ce fichier, et que ces programmes se partagent 7500 titres uniques. Par titre unique, l'on définit les **shortName** générés précédemment (simplification ASCII de la concaténation de l'ID de la chaîne et du titre du programme, eg. « meteoc192apiteleramafr »). L'instance la plus utilisée (la première), « lamatinalec226apiteleramafr », représente 2704 instances, ou environ 1,9% de l'entièreté des programmes du fichier. Comme autres chiffres intéressants, l'on peut relever que les 25 premiers éléments représentent 26 278 instances, équivalent à environ 18,5% de l'entièreté des instances. Les 200 premiers éléments représentent plus de 52% des instances, ou 73 957 programmes.

La même chose a été réalisée avec les descriptions, nous donnant un total de 19 206 descriptions uniques, avec la plus populaire : l'absence de description, avec 47 555 occurrences, ou environ 33,5% des programmes.



Les deux instances de descriptions les plus populaire après l'absence de description sont « Tous les matins, la matinale de CNews présente l'actualité avec des invités et à travers plusieurs rubriques, dont des rubriques politiques, économiques, sanitaires, sportives et culturelles. » (La matinale de CNews) et « Replay des émissions de la chaîne TV CNEWS. Retour sur l'actualité de ces derniers jours avec la rediffusion des magazines de la chaîne. », à respectivement 2 566 et 2 303 occurrences.

II.3 – Extractions

Les premières extractions de données et générations de .csv réalisées ont donc été en lien avec la mesure de fréquence des programmes. Quelques fichiers ont été générés à partir de tri décroissant sur la fréquence, et l'on a surtout séparé les programmes dits « fréquents » des programmes dits « non fréquents » avec une limite très simple : plus de 4 occurrences sur ces 4 mois de données signifient un potentiel programme récurrent, l'on l'envoie donc dans les « fréquents », même si cela n'est pas exactement véridique.

Une autre extraction de données ayant été réalisée fut l'extraction de toutes les icônes uniques utilisés dans ces données, afin d'avoir une liste sur laquelle travailler plus tard. Pour information, il y a 6396 icônes uniques pour 141 824 instances de programmes, ou pour 7500 programmes différents.

De même, quelques extractions de données spécifiques ont été réalisées afin de les transmettre aux autres collègues travaillant également sur ce projet, notamment Célie, qui a reçu de ma part une liste de descriptions les plus utilisées, triées par ordre de fréquence. Une autre liste fut demandée plus tard, avec la liste de tous les résumés uniques du fichier XMLTV. Au fil du stage, quelques erreurs furent relevées dans ces listes, ces dernières ayant été corrigées au fur et à mesure.

Tous ces fichiers respectaient à peu de choses près le même format (.CSV), afin de rendre la lecture du fichier simple et rapide. Sur les pages suivantes, quelques exemples et extraits des fichiers.

Extrait de la liste de programmes « fréquents » avec doublons (lignes séparées d'un double saut de ligne afin de les différencier)

```
Full title;Channel ID;Condensed Name;Description;Categories;Icones
Programmes de la nuit
;C192.api.telerama.fr;programmesdelanuitc192apiteleramafr;None;[' Programme
indéterminé '];[];

TFou ;C192.api.telerama.fr;tfouc192apiteleramafr;Une émission jeunesse qui
propose les meilleures séries de dessins animés du moment. ;[' Magazine
'];['https://television.telerama.fr/sites/tr_master/files/sheet_media/media/d13
2093f961e7163e30c7d86d808256a25270431.jpg'];

Météo ;C192.api.telerama.fr;meteoc192apiteleramafr;None;[' Météo '];[];

Les douze coups de
midi;C192.api.telerama.fr;lesdouzecoupsdemidic192apiteleramafr;None;[' Jeu
'];[];

Petits plats en équilibre
;C192.api.telerama.fr;petitsplatsenequilibrec192apiteleramafr;None;[' Magazine
'];[];

Journal ;C192.api.telerama.fr;journalc192apiteleramafr;None;[' Journal '];[];

…
```

Extrait de la liste d'icônes uniques, avec ligne exemple :

```
Icon URL;Status (intra/inter);#Programs that used it;Array of programs that
used it
https://television.telerama.fr/sites/tr_master/files/sheet_media/media/f854f625
4b467e0c03f554a57351b98c9d85b14c.jpg;reused|inter-programme;58; [Liste des
shortName des programmes utilisant cet icône]
```

Extrait de la première liste de données envoyée à Célie, aka liste des programmes fréquents triés de manière décroissante sur la fréquence

```
Title;Occurrences (Pair title + program);Frequency
(occ/day);Category;Description;

lamatinalec226apiteleramafr;2566;26.183673469387756;Generic;"Tous les matins, la
matinale de CNews présente l'actualité avec des invités et à travers plusieurs
rubriques, dont des rubriques politiques, économiques, sanitaires, sportives et
culturelles. ";

lebestofdenuitc226apiteleramafr;2303;23.5;Generic;"Replay des émissions de la
chaîne TV CNEWS. Retour sur l'actualité de ces derniers jours avec la rediffusion
des magazines de la chaîne. ";

alvinnnetleschipmunksc482apiteleramafr;1499;15.295918367346939;No Desc;"None";

boucledenuitc226apiteleramafr;1295;13.214285714285714;No Desc;"None";

bienvenuechezlesloudc482apiteleramafr;1176;12.0;No Desc;"None";
```

Partie III : Jaccard et N-Grammes

III.1 – Contexte

Une fois ces étapes terminées, qui étaient en réalité préliminaires afin de pouvoir fournir les données aux collègues, il était donc possible de commencer à travailler sur ma partie du pipeline. Comme représenté sur le schéma du rapport, j'ai donc deux parties en parallèle à travailler qui viennent ensuite se rejoindre sur l'augmentation du fichier XMLTV.

Avec la courte durée du stage, et au fur et à mesure que je creusais dans le problème de détection de résumés quasi-similaires, la décision de se concentrer sur cette partie pour mettre à part la partie sur les icônes, car pour les icônes, il aurait fallu mettre en place un crawler pour télécharger l'entièreté icônes, mais aussi les analyser visuellement afin de déterminer s'il existait effectivement des quasi-duplicata. L'on s'est donc contentés de trier entre générique et non générique, en fonction de l'utilisation de l'icône au travers des données.

III.2 – Objectif et approche du problème

Notre objectif pour cette partie, est donc de diviser les descriptions uniques entre trois catégories fixes : **Unique** (= non generic), des descriptions n'ayant lieu qu'une fois ou qu'un nombre très restreint de fois, **Générique** (= generic), des descriptions ayant plusieurs occurrences, fréquentes. Et enfin **Similaires** (= near duplicate), des descriptions se ressemblant à quelques mots près, comme une description avec sa correction qui rajoute un mot.

Extrait du schéma du pipeline du projet : exemple de tri de descriptions

Generic	**13h15, le dimanche:** Des images, des émotions, une écriture sensible aux aventures humaines : "13h15" propose ....
No generic	**13h15, le dimanche:** Entre le Royaume Uni d'Elizabeth II et la France, s'est nouée une relation tourmentée...
Near duplicate	De l'Atlantique à l'Océan Indien, du Pacifique à la Mer des Caraïbes, **Kelly Rangama, chef étoilé,** sillonne ....
	De l'Atlantique à l'Océan Indien, du Pacifique à la Mer des Caraïbes, **la cheffe étoilée Kelly Rangama** sillonne ...

Afin de séparer et de trier toutes ces descriptions, l'on a plusieurs options. Les premières analyses les plus basiques, seraient d'étudier le nombre d'occurrences de chaque description, afin de séparer **Unique** de **Générique**. Cependant, cela ne nous permet pas et ne suffit clairement pas pour extraire les descriptions **Similaires**. Effectivement, s'il y a simplement plusieurs versions d'une description similaire, l'on va simplement se retrouver avec plusieurs **Uniques**.

Pour compléter cette classification, nous allons analyser le vocabulaire des descriptions, notamment l'étude des textes à l'aide des N-Grammes (ensemble de N mots se suivant dans une description), et d'ensuite une étude statistique des ensembles de N-Grammes de chaque description. Pour ce faire, il fut envisagé d'utiliser une librairie NPL (Natural Language Processing) afin d'extraire aisément les N-Grammes, mais cela n'avait que peu d'intérêt, de charger une si grosse librairie juste pour une utilisation dans le pipeline entier. Je me suis donc rabattu sur la fonction CountVectorizer de scikit learn, qui prend en entrée une liste de textes et quelques paramètres, et qui permet de générer deux éléments : Une liste complète des « tokens » trouvés dans l'ensemble des textes, ou un vocabulaire si l'on utilise des 1-grammes, par exemple. Le second élément est une matrice de dimensions Nb Textes x Taille du Vocabulaire, avec chaque élément $x_{i,j}$ représentant le nombre d'occurrences de l'élément à la position j dans le vocabulaire, dans le texte en position i dans la liste d'entrée.

Avec ces données, il devient possible de réaliser l'étude statistique précédemment citée : calculer l'indice de Jaccard. Il s'agit d'un coefficient permettant de mesurer la similarité entre deux ensembles définis, et l'on va s'en servir pour comparer le vocabulaire ou l'ensemble des N-Grammes de deux descriptions d'un même programme. Cela nous permettra d'avoir une valeur à observer pour déterminer s'il s'agit d'une description complètement différente ou effectivement d'un cas de descriptions **Similaires**, ou une mise à jour. L'on va aussi étudier les différences de vocabulaires entre deux programmes, en prenant en compte l'entièreté des descriptions concernant ce programme.

L'indice de Jaccard est relativement simple à calculer, il s'agit simplement de la valeur suivante, pour deux ensembles A et B :

$$J\,(A,B) = \frac{|\,A\ \cap B\,|}{|\,A\ \cup B\,|}$$

Schéma représentant des exemples d'utilisation de l'indice de Jaccard :

Cet indice, pour deux ensembles A et B, nous donne une valeur entre 0 et 1, et plus la valeur se rapproche de 1, plus les deux ensembles sont similaires, comme visible sur le schéma ci-dessus. L'indice de Jaccard peut être utilisé dans diverses domaines, notamment dans l'analyse visuelle des images avec l'aide de réseaux de neurones, afin de comparer les prédictions du réseau des résultats attendus (un indice élevé = un bon score). L'on va s'en servir dans ce projet afin de comparer les deux textes ayant généré les deux ensembles A et B.

III.3 – Recherche de N

Jusqu'à présent, l'on parlait de N-Grammes. Cependant, il a été rapide de remarquer qu'essayer de traiter plusieurs N-Grammes en même temps pose rapidement un souci avec la méthode CountVectorizer, car cela génère une matrice énorme avec la quantité de données à traiter (matrice d'environ 32Go si l'on essaie de faire une matrice avec les N-Grammes avec N = [1-8]). Ainsi, il a fallu de mon côté, traiter chaque cas de N un par un, et stocker les données. Pour cela, j'ai simplement stocké une matrice de résultats de taille [Nombre de Descriptions]² sous forme de .CSV.

Extrait du fichier de résultats pour les 4-grammes, inter-programmes.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	Program Long Name;Array of Similarity																	
2	tfouc192apit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	icitoutcomm	None	0.00484051	0	0	0.00075429	0.00022619	0	0	0	0.00020194	0	0	0	0.00255319	0	0	0
4	demainnous	0.00484051	None	0	0	0.00018012	0.00042863	0	0	0	0.00038484	0	0	0	0.0020059	0	0	0
5	districtzc192	0	0	None	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	vendreditou	0	0	0	None	0	0	0	0	0	0	0	0	0	0	0	0	0
7	reportagesd	0.00075429	0.00018012	0	0	None	0.00053248	0	0	0	0.00124636	0	0	0	0.00017538	0	0.0013986	0
8	lesdocsduwe	0.00022619	0.00042863	0	0	0.00053248	None	0	0	0	0.0013132	0	0	0	0.0002076	0	0	0
9	ninjawarrior	0	0	0	0	0	0	None	0	0	0	0	0	0	0	0	0	0
10	automotoc19	0	0	0	0	0	0	0	None	0	0	0	0	0	0	0	0	0
11	telefootc192	0	0	0	0	0	0	0	0	None	0	0	0	0	0	0	0	0
12	grandsreport	0.00020194	0.00038484	0	0	0.00124636	0.0013132	0	0	0	None	0	0	0	0.00037404	0.00093284	0.00092851	0
13	septahuitlife	0	0	0	0	0	0	0	0	0	0	None	0	0	0	0	0	0
14	septahuitc19	0	0	0	0	0	0	0	0	0	0	0	None	0	0	0	0	0
15	petitssecrets	0	0	0	0	0	0	0	0	0	0	0	0	None	0	0	0	0
16	lesfeuxdelar	0.00255319	0.0020059	0	0	0.00017538	0.0002076	0	0	0	0.00037404	0	0	0	None	0	0	0
17	aucoeurdesr	0	0	0	0	0	0	0	0	0	0.00093284	0	0	0	0	None	0	0
18	famillesnom	0	0	0	0	0.0013986	0	0	0	0	0.00092851	0	0	0	0	0	None	0
19	cestcantelou	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	None

Ces valeurs nous permettent donc de comparer les descriptions d'un programme à celles d'un autre, mais cela n'est pas très utile. L'on peut limite déterminer si certains programmes ont le même genre de description qu'un autre, mais cela n'est pas ce que l'on recherche. Il est tout de même utile de remarquer que très peu d'indices vont au-delà de 0.001, ce qui nous dit que très peu de descriptions usent des mêmes formulations ou vocabulaire. Les fichiers de résultats pour N allant de 2 à 8 ont été générés, ce qui nous fait

énormément de résultats. Plus l'on monte en N, plus les valeurs se rapprochent de 0, car l'on cherche des formulations similaires, plus que de simples paires de mots.

Les résultats intra-programmes sont cependant plus intéressants. L'on remarquera que dans le format du fichier, il y a aussi le compte d'apparition par description, ce qui pourra nous servir dans notre classification, car l'on peut en déduire qu'une description **Générique** est probablement utilisée très fréquemment au sein du même programme. L'on remarque aussi que, avec le premier exemple, l'on a déjà un cas d'une description majoritaire, avec 70 occurrences (en rouge dans l'exemple sur la page suivante), contre 2 occurrences pour la seconde description. Le second exemple est encore un court exemple, avec 5 descriptions. L'on a omis les matrices de résultats après N = 2, car sinon la ligne du fichier ferait une page entière. Quelques retours à la ligne ont également été ajoutés, dans la tentative de clarifier un peu plus la lecture des matrices.

Ces indices, valeurs et résultats seront ensuite utilisés afin de finalement faire une séparation entre nos trois catégories. Mais avant cela, il fallait déterminer quelle valeur de N est la plus intéressante afin de ne pas avoir à recalculer tous les N-Grammes à chaque étude que l'on souhaite réaliser, mais surtout pour rendre le problème de séparation entre les catégories, afin qu'il y ait une réelle distance entre un cas **Unique** et **Similaire** (= near duplicate).

Après quelques analyses des résultats inter et intra programme, l'on a déterminé qu'un N élevé est le plus intéressant, et l'on s'est fixé sur N = 8.

Format du fichier de résultats pour l'étude intra-programme, quelques courts exemples de lignes :

Full Title;Array of Descriptions;Array of Descriptions' occurrences;Matrix 1-gram;Matrix 2-grams;Matrix 3-grams;Matrix 4-grams;Matrix 5-grams;Matrix 6-grams;Matrix 7-grams;Matrix 8-grams

"mikeuneviedechienc80apiteleramafr";"[Liste de descriptions]";"[70, 2]";"[[None, 0.13559322033898305], [0.13559322033898305, None]]";"[[None, 0.0], [0.0, None]]";"[[None, 0.0], [0.0, None]]";"[[None, 0.0], [0.0, None]]";"[[None, 0.0], [0.0, None]]";"[[None, 0.0], [0.0, None]]";"[[None, 0.0], [0.0, None]]";"[[None, 0.0], [0.0, None]]";

"laptitelibrairiec80apiteleramafr";"[Liste de descriptions]";"[85, 147, 11, 29, 1]";"[[None, 0.8260869565217391, 0.0625, 0.9, 0.030303030303030304],
[0.8260869565217391, None, 0.058823529411764705, 0.7391304347826086, 0.02857142857142857],
[0.0625, 0.058823529411764705, None, 0.06666666666666667, 0.6470588235294118],
[0.9, 0.7391304347826086, 0.06666666666666667, None, 0.03225806451612903],
[0.030303030303030304, 0.02857142857142857, 0.6470588235294118, 0.03225806451612903, None]]";"
[[None, 0.6923076923076923, 0.029411764705882353, 0.8181818181818182, 0.0],
[0.6923076923076923, None, 0.027777777777777776, 0.5555555555555556, 0.0],
[0.029411764705882353, 0.027777777777777776, None, 0.03125, 0.47368421052631576],
[0.8181818181818182, 0.5555555555555556, 0.03125, None, 0.0],
[0.0, 0.0, 0.47368421052631576, 0.0, None]]";

III.4 – Résultats

Enfin, une fois toutes ces étapes de recherche, d'analyse de résultats, et de calcul réalisés et implémentés, l'on applique donc notre séparation aux données. Les résultats de cette séparation ont été stockés dans un .CSV, encore une fois, afin de les mettre de côté, car nous allons nous en servir pour l'augmentation des données, dans la partie suivante du projet.

Des exemples de résultats sont disponibles ci-dessous, avec quelques commentaires.

Format du fichier de résultats et quelques lignes d'exemple :

```
Condensed Name ; Description ; Occurrences ; Category ; Comments ;

"tfouc192apiteleramafr";'Une émission jeunesse qui propose les meilleures
séries de dessins animés du moment. ;"116";"Generic";"Major use in regard to
occurrences.";
"tfouc192apiteleramafr";"Au programme : Dora l'exploratrice. - Abby Hatcher. -
Agent Binky. - Loup. - Chouette pas chouette. ";"1";"Near duplicate";"Possible
update";
"tfouc192apiteleramafr";"Au sommaire : Dora l'exploratrice. - Abby Hatcher. -
Agent Binky. - Loup. - Chouette pas chouette. ";"1";"Near duplicate";"Possible
update";"
[…]
"icitoutcommencec192apiteleramafr";"Noémie et Antoine sont stupéfaits des
décisions de Claire tandis que Rose et Clotilde demandent des preuves de sa
bonne foi. ";"2";"Specific";"Multiple descriptions";
icitoutcommencec192apiteleramafr";""Louis est persuadé que sa mère lui a menti
toute sa vie tandis que Constance raconte à Guillaume l'étrange comportement de
sa femme. ";"2";"Specific";"Multiple descriptions";"
[…]
"simonc47apiteleramafr";""En jouant au foot avec Gaspard dans le parc, Simon
découvre un skate dans un buisson et l'essaie avec Gaspard. Mais ils perdent
l'équilibre et éjectent la planche que Ferdinand récupère et ne veut pas rendre
à Simon. Finalement, les deux copains se mettent d'accord pour en faire chacun
son tour. Gaspard, qui veut lui aussi en faire, se lance avec Ferdinand et ils
passent à fond devant Lou, qui reconnaît son skate… ";"3";"Specific";"Multiple
```

Les résultats sont donc formattés ainsi, avec en plus de leur catégorie, leur nombre d'occurrences dans les données, une catégorie « commentaire » a été rajoutée, qui donne un peu plus de détails sur comment la description s'est retrouvée triée dans cette catégorie.

III.5 – Et les images ?

Pour ce qui est des images, peu d'études ont été réalisées, mais il y a tout de même eu un peu de travail, même si l'on s'est accordés pour se concentrer sur la partie « descriptions » du projet. Les images ont tout d'abord été recensées, ainsi que leur nombre d'occurrences, comme pour les descriptions. Tout cela état déjà en place, grâce aux extractions réalisées plus tôt. Ensuite, il a été tenté d'ajouter des mots-clés aux images, en partant des mots clés des programmes dans lesquels ces images ont été trouvées, afin d'essayer de pouvoir les catégoriser plus précisément. Cependant, les données sont relativement mal documentées, car la majeure partie des mots clés, ou catégories, lorsqu'elles sont présentes, sont uniques. Il n'y a qu'une catégorie ou mot-clé par programme, généralement. Cela rend l'augmentation et l'analyse difficile. Une séparation rudimentaire sur le nombre d'occurrences et sur la base du fait que l'icône soit propre à un programme ou non a été réalisé, afin de séparer entre **Unique** ou **Générique**. Le travail sur les icônes s'arrête cependant ici.

Cependant, quelques chemins ont été envisagés afin d'aller plus loin : notamment, les métadonnées des images. Pour aller plus loin, il serait possible d'augmenter et d'utiliser les métadonnées des icônes après les avoir téléchargés. Ces dernières sont directement encodées dans le fichier image, ce qui nous permettrait d'obtenir d'autres mots-clés, de les associer avec d'autres programmes. Il serait également possible, comme cité précédemment, d'user de réseaux de neurones pour comparer des images similaires, afin de déterminer s'il s'agit de simples recadrages, afin de conserver la version la plus complète, plutôt qu'une variation inférieure en résolution et en données.

Partie IV : XMLTV et Augmentation

IV.1 – Contexte

Une fois les études, recherches, analyses, et catégorisations terminées, du côté images comme du côté descriptions, il devient possible d'user les résultats de ces catégorisations afin d'augmenter les données (= les compléter), pour obtenir un meilleur taux de couverture, et donc minimiser les « trous » de données. (Programmes sans descriptions ni icônes, par exemple.)

Extrait du schéma du pipeline du projet : Exemples d'augmentation des données

Coup de projecteur sur l'actualité musicale, dans un format dynamique et accessible.	Basique; Magazine; Musique
Charlotte Gainsbourg, chanteuse et actrice, a enregistré cinq disques qui explorent des univers musicaux …	Charlotte; Gainsbourg; Chanteuse
Cela fait plus de vingt ans que Raphaël explore les facettes de la chanson française …	Raphaël; Chanteur; Concert

L'objectif principal ici est d'observer les variations du taux de couverture entre avant que l'on ajoute des données, et après. L'on note donc les valeurs avant ici ;

- Taux de couverture des descriptions pré-augmentation : 66,47% environ
- Taux de couverture des images pré-augmentation : 49,37% environ

IV.1 – Mise en place

L'augmentation des données se fait de manière plutôt aisée, et ce, grâce au travail d'études et d'analyses déjà réalisé. L'on a déjà extrait tous les programmes et toutes les informations que l'on pouvait compléter, l'on a même trié ces informations et catégorisé avec des noms que l'on va pouvoir réutiliser désormais.

Pour les descriptions, il s'agit simplement de parcourir chaque programme, et si la description présente est considérée comme générique, l'on peut la mettre à jour en ajoutant la version plus récente, si l'on en possède une. S'il n'y a pas de description du tout, il suffit d'ajouter la même version, tant qu'elle est catégorisée comme générique. Cela dit, l'augmentation du taux de couverture ne sera pas titanesque, car après tout, l'on utilise les informations d'une source de données pour la compléter elle-même. Une meilleure solution serait d'utiliser plusieurs sources, l'on en parlera plus tard. Le taux de couverture des descriptions, avec cette augmentation, monte donc à 67,82% environ. Une petite hausse d'à peine 1,35%, mais une hausse tout de même. Sur les 141 824 instances, cela signifie tout de même un ajout de plus de 1 900 descriptions, ce qui n'est pas négligeable.

Pour les images, afin d'éviter une sur-augmentation à cause d'icônes trop génériques, seules les icônes des mêmes programmes ont été utilisés pour augmenter les programmes ayant le même titre. Si l'on faisait sur les mots-clés, ou en fonction de leur généricité, l'on se retrouverait probablement avec trop d'icônes sur beaucoup de programmes, à cause de certaines icônes pouvant être utilisées absolument partout. L'on monte à 51,02% après augmentation, ce qui est une petite hausse, mais une hausse tout de même, pour une technique simple qui n'utilise pas de métadonnées.

IV.2 – Aller plus loin

L'augmentation des données comme présenté ici, n'est qu'un début et une démonstration de faisabilité. Les méthodes mises en place ne sont encore que rudimentaires, il est possible d'aller plus loin, de rechercher un meilleur N pour les N-Grammes, d'utiliser de meilleurs seuils pour détecter un programme récurrent, ou même une meilleure méthode entièrement. Le plus efficace, cependant, serait simplement d'utiliser plusieurs sources, et de croiser les informations. S'il est possible d'extraire plusieurs résumés différents de plusieurs sites pour un même programme, l'on a alors plus de possibilités pour l'augmentation des données. De même pour les icônes.

IV.3 – Résultat de l'augmentation

Une fois toute cette augmentation réalisée, l'on a regénéré un nouveau fichier au format XMLTV, avec les nouvelles données. La différence de taille est présente entre les deux fichiers, bien que minime (69.1 Mo -> 69.9 Mo). Ce fichier augmenté sera ensuite utilisé pour d'autres missions, plus tard, reprenant mon travail.

Partie V : Extractions sur demande

V.1 – Contexte

A plusieurs reprises pendant ce projet et plus particulièrement ce stage, j'ai eu diverses demandes de la part de mon encadrant ou de la part de collègues pour extraire des données spécifiques, et les fournir plus tard. L'augmentation du fichier XMLTV d'origine était importante aussi, et représentait une extraction majeure, ce pourquoi elle a sa propre partie sur ce rapport.

V.2 – Premières extractions

Comme dit précédemment dans la partie II, quelques fichiers ont été extraits afin de fournir des données basiques à Célie, afin de lui permettre d'avoir une base d'information pour travailler sa partie plus proche du front. L'on ne va pas redétailler tout cela ici, c'est déjà fait dans la partie II.

V.3 – Extraction des données du mois 2

Durant la période pendant laquelle j'ai travaillé sur l'augmentation des données, une autre extraction de données m'a été demandée par M. Delalandre afin de fournir des données supplémentaires à Hao Le. L'on m'a d'abord demandé de réunir les données d'un mois particulier, avec les titres, chaînes, et descriptions **génériques** des programmes en question, dans la période du mois précisé. Hao a ensuite précisé le format.

Après quelques extractions rudimentaires afin de bien séparer les données du premier mois, j'ai fait un premier retour, mais j'avais mal interprété les consignes. En effet, il fallait que je fournisse, pour chaque programme présent durant le mois (en évitant les doublons), l'image la plus générique que je puisse associer à ce programme, et la description la plus générique provenant de ces programmes. En plus de cela, il faut rajouter un hashcode, et quelques informations supplémentaires, comme les catégories.

Ainsi, après correction, j'ai fourni les données demandées à l'origine. Cependant, une autre demande est parvenue post-fin de stage, au sujet de visiblement une confusion sur la méthode de hachage pour les hashcodes, car nous n'avons pas les mêmes hashcodes pour les mêmes données. Nous utilisons probablement deux méthodes différentes, ce qui est problématique.

> *Comparaison de format avant/après correction des instructions :*
>
> Format que j'ai interprété et renvoyé au début
>
> Channel ID ; Title ; Categories ; Description
>
> *C192.api.telerama.fr; TFou ;[' Magazine '];Au sommaire : Octonauts. - Top Wing Toutes Ailes Dehors ! - Chuggington. - Splat Harry. - Alice et Lewis. - Heidi. - Oum le dauphin blanc. - Droners. – Miraculous*
>
> Format après correction
>
> Hashcode; Channel ID; Program Title; Categories; Generic Icon Url; Generic Description;
>
> *4114510244243302070;C192.api.telerama.fr; TFou ;[' Magazine '];['https://television.telerama.fr/sites/tr_master/files/sheet_media/media/12bc7016a6325cda0126ec7389203fd852601506.jpg'];'Une émission jeunesse qui propose les meilleures séries de dessins animés du moment. ;*

Conclusion du stage

Conclusion vis-à-vis du projet

Ce projet, du moins dans le cadre de mon stage, était surtout un POC (Proof Of Concept), qui permettrait de poser les bases d'un pipeline à ensuite mettre réellement en place. Je dirais personnellement, après les retours que l'on a eu ensemble avec M. Delalandre, que le POC est réussi, et que l'objectif principal de stage est atteint. Cependant, il aurait été possible d'aller plus loin avec notamment un peu plus d'optimisation du côté N-Grammes et analyse.

Conclusion personnelle

Avec ce stage entièrement en distanciel, qui m'a forcé à m'adapter aux conditions de travail à distance, je pense avoir gagné en autonomie, mais aussi en compétences avec les quelques méthodes que je n'avais pas utilisé jusqu'à présent (études statistiques avec la distance de Jaccard, N-Grammes). Le fait que le stage ne soit pas vraiment dans le domaine de ma spécialité (ASR) m'a fait effectuer quelques recherches et utiliser d'autres technologies, que je n'aurais probablement pas vu autrement. Je pense personnellement que ce stage fut un très bon stage de 4A.

Bibliographie et Annexes

Sites utiles

[1] Laboratoire d'Informatique Fondamentale et Appliquée de Tours, site web https://lifat.univ-tours.fr/

[2] Wiki XMLTV, site web, http://wiki.xmltv.org/index.php/XMLTVFormat

Jason Brownlee, A Gentle Introduction to the Bag-Of-Words Model, site web, dernière mise à jour le 7/08/2019 au temps de rédaction, https://machinelearningmastery.com/gentle-introduction-bag-words-model/

Annexes

Tableau de simplification ASCII

ID char	Notation	Description	ID char à appliquer
0	NUL	Null char	0
1	SOH	Start of Heading	0
2	STX	Start of Text	0
3	ETX	End of Text	0
4	EOT	End of Transmission	0
5	ENQ	Enquiry	0
6	ACK	Acknowledgment	0
7	BEL	Bell	0
8	BS	Back Space	0
9	HT	Horizontal Tab	0
10	LF	Line Feed	0
11	VT	Vertical Tab	0
12	FF	Form Feed	0
13	CR	Carriage Return	0
14	SO	Shift Out / X-On	0
15	SI	Shift In / X-Off	0
16	DLE	Data Line Escape	0
17	DC1	Device Control 1 (oft. XON)	0
18	DC2	Device Control 2	0
19	DC3	Device Control 3 (oft. XOFF)	0
20	DC4	Device Control 4	0
21	NAK	Negative Acknowledgement	0
22	SYN	Synchronous Idle	0
23	ETB	End of Transmit Block	0
24	CAN	Cancel	0
25	EM	End of Medium	0
26	SUB	Substitute	0
27	ESC	Escape	0
28	FS	File Separator	0
29	GS	Group Separator	0
30	RS	Record Separator	0
31	US	Unit Separator	0
32		Space	0
33	!	Exclamation mark	0
34	"	Double quotes (or speech marks)	0
35	#	Number	0
36	$	Dollar	0
37	%	Per cent sign	0
38	&	Ampersand	0
39	'	Single quote	0
40	(	Open parenthesis (or open bracket)	0

41	)	Close parenthesis (or close bracket)	0
42	*	Asterisk	0
43	+	Plus	0
44	,	Comma	0
45	-	Hyphen	0
46	.	Period, dot or full stop	0
47	/	Slash or divide	0
48	0	Zero	48
49	1	One	49
50	2	Two	50
51	3	Three	51
52	4	Four	52
53	5	Five	53
54	6	Six	54
55	7	Seven	55
56	8	Eight	56
57	9	Nine	57
58	:	Colon	0
59	;	Semicolon	0
60	<	Less than (or open angled bracket)	0
61	=	Equals	0
62	>	Greater than (or close angled bracket)	0
63	?	Question mark	0
64	@	At symbol	0
65	A	Uppercase A	97
66	B	Uppercase B	98
67	C	Uppercase C	99
68	D	Uppercase D	100
69	E	Uppercase E	101
70	F	Uppercase F	102
71	G	Uppercase G	103
72	H	Uppercase H	104
73	I	Uppercase I	105
74	J	Uppercase J	106
75	K	Uppercase K	107
76	L	Uppercase L	108
77	M	Uppercase M	109
78	N	Uppercase N	110
79	O	Uppercase O	111
80	P	Uppercase P	112
81	Q	Uppercase Q	113
82	R	Uppercase R	114
83	S	Uppercase S	115
84	T	Uppercase T	116
85	U	Uppercase U	117

86	V	Uppercase V	118
87	W	Uppercase W	119
88	X	Uppercase X	120
89	Y	Uppercase Y	121
90	Z	Uppercase Z	122
91	[	Opening bracket	0
92	\	Backslash	0
93	]	Closing bracket	0
94	^	Caret - circumflex	0
95	_	Underscore	0
96	`	Grave accent	0
97	a	Lowercase a	97
98	b	Lowercase b	98
99	c	Lowercase c	99
100	d	Lowercase d	100
101	e	Lowercase e	101
102	f	Lowercase f	102
103	g	Lowercase g	103
104	h	Lowercase h	104
105	i	Lowercase i	105
106	j	Lowercase j	106
107	k	Lowercase k	107
108	l	Lowercase l	108
109	m	Lowercase m	109
110	n	Lowercase n	110
111	o	Lowercase o	111
112	p	Lowercase p	112
113	q	Lowercase q	113
114	r	Lowercase r	114
115	s	Lowercase s	115
116	t	Lowercase t	116
117	u	Lowercase u	117
118	v	Lowercase v	118
119	w	Lowercase w	119
120	x	Lowercase x	120
121	y	Lowercase y	121
122	z	Lowercase z	122
123	{	Opening brace	0
124	|	Vertical bar	0
125	}	Closing brace	0
126	~	Equivalency sign - tilde	0
127		Delete	0
128	€	Euro sign	0
129			0
130	,	Single low-9 quotation mark	0

131	ƒ	Latin small letter f with hook	0
132	„	Double low-9 quotation mark	0
133	…	Horizontal ellipsis	0
134	†	Dagger	0
135	‡	Double dagger	0
136	ˆ	Modifier letter circumflex accent	0
137	‰	Per mille sign	0
138	Š	Latin capital letter S with caron	0
139	‹	Single left-pointing angle quotation	0
140	Œ	Latin capital ligature OE	156
141			0
142	Ž	Latin capital letter Z with caron	122
143			0
144			0
145	‘	Left single quotation mark	0
146	’	Right single quotation mark	0
147	“	Left double quotation mark	0
148	”	Right double quotation mark	0
149	•	Bullet	0
150	–	En dash	0
151	—	Em dash	0
152	˜	Small tilde	0
153	™	Trade mark sign	0
154	š	Latin small letter S with caron	115
155	›	Single right-pointing angle quotation mark	0
156	œ	Latin small ligature oe	156
157			0
158	ž	Latin small letter z with caron	122
159	Ÿ	Latin capital letter Y with diaeresis	121
160		Non-breaking space	0
161	¡	Inverted exclamation mark	105
162	¢	Cent sign	0
163	£	Pound sign	0
164	¤	Currency sign	0
165	¥	Yen sign	0
166	¦	Pipe, Broken vertical bar	0
167	§	Section sign	0
168	¨	Spacing diaeresis - umlaut	0
169	©	Copyright sign	0
170	ª	Feminine ordinal indicator	0
171	«	Left double angle quotes	0
172	¬	Not sign	0
173		Soft hyphen	0
174	®	Registered trade mark sign	0

175	¯	Spacing macron - overline	0
176	°	Degree sign	0
177	±	Plus-or-minus sign	0
178	²	Superscript two - squared	0
179	³	Superscript three - cubed	0
180	´	Acute accent - spacing acute	0
181	µ	Micro sign	0
182	¶	Pilcrow sign - paragraph sign	0
183	·	Middle dot - Georgian comma	0
184	¸	Spacing cedilla	0
185	¹	Superscript one	0
186	º	Masculine ordinal indicator	0
187	»	Right double angle quotes	0
188	¼	Fraction one quarter	0
189	½	Fraction one half	0
190	¾	Fraction three quarters	0
191	¿	Inverted question mark	0
192	À	Latin capital letter A with grave	97
193	Á	Latin capital letter A with acute	97
194	Â	Latin capital letter A with circumflex	97
195	Ã	Latin capital letter A with tilde	97
196	Ä	Latin capital letter A with diaeresis	97
197	Å	Latin capital letter A with ring above	97
198	Æ	Latin capital letter AE	230
199	Ç	Latin capital letter C with cedilla	99
200	È	Latin capital letter E with grave	101
201	É	Latin capital letter E with acute	101
202	Ê	Latin capital letter E with circumflex	101
203	Ë	Latin capital letter E with diaeresis	101
204	Ì	Latin capital letter I with grave	105
205	Í	Latin capital letter I with acute	105
206	Î	Latin capital letter I with circumflex	105
207	Ï	Latin capital letter I with diaeresis	105
208	Ð	Latin capital letter ETH	100
209	Ñ	Latin capital letter N with tilde	110
210	Ò	Latin capital letter O with grave	111
211	Ó	Latin capital letter O with acute	111
212	Ô	Latin capital letter O with circumflex	111
213	Õ	Latin capital letter O with tilde	111
214	Ö	Latin capital letter O with diaeresis	111
215	×	Multiplication sign	0
216	Ø	Latin capital letter O with slash	111
217	Ù	Latin capital letter U with grave	117
218	Ú	Latin capital letter U with acute	117
219	Û	Latin capital letter U with circumflex	117

220	Ü	Latin capital letter U with diaeresis	117
221	Ý	Latin capital letter Y with acute	121
222	Þ	Latin capital letter THORN	0
223	ß	Latin small letter sharp s - ess-zed	0
224	à	Latin small letter a with grave	97
225	á	Latin small letter a with acute	97
226	â	Latin small letter a with circumflex	97
227	ã	Latin small letter a with tilde	97
228	ä	Latin small letter a with diaeresis	97
229	å	Latin small letter a with ring above	97
230	æ	Latin small letter ae	230
231	ç	Latin small letter c with cedilla	99
232	è	Latin small letter e with grave	101
233	é	Latin small letter e with acute	101
234	ê	Latin small letter e with circumflex	101
235	ë	Latin small letter e with diaeresis	101
236	ì	Latin small letter i with grave	105
237	í	Latin small letter i with acute	105
238	î	Latin small letter i with circumflex	105
239	ï	Latin small letter i with diaeresis	105
240	ð	Latin small letter eth	0
241	ñ	Latin small letter n with tilde	110
242	ò	Latin small letter o with grave	111
243	ó	Latin small letter o with acute	111
244	ô	Latin small letter o with circumflex	111
245	õ	Latin small letter o with tilde	111
246	ö	Latin small letter o with diaeresis	111
247	÷	Division sign	0
248	ø	Latin small letter o with slash	111
249	ù	Latin small letter u with grave	117
250	ú	Latin small letter u with acute	117
251	û	Latin small letter u with circumflex	117
252	ü	Latin small letter u with diaeresis	117
253	ý	Latin small letter y with acute	121
254	þ	Latin small letter thorn	0
255	ÿ	Latin small letter y with diaeresis	121

Stage : Projet StationTV

Court Résumé

Le projet StationTV, mené par le Laboratoire LIFAT[1] depuis 2019, est un ensemble de différentes parties, toutes dans l'objectif du développement de nouveaux services numériques autour de la Télévision, du Replay/VoD et d'Internet. Dans ce projet, j'ai pu prendre part à la partie « back-end », où j'ai eu diverses tâches en relation avec du traitement de données, et de l'analyse de ces dernières, avec comme objectif final l'augmentation des données. Plus particulièrement, la majeure partie de mes tâches dans ce stage sont en lien avec le traitement d'un fichier au format XMLTV[2]. Le projet a été réalisé en Python, avec l'aide d'Anaconda et sur le Jupyter Notebook.

Mots-Clés

XMLTV, Parsing, Analyse, Extraction, Augmentation de données, Preuve de Concept