Détection automatique des liens d'articles dans la *une* des journaux en ligne

Cédric$_1$ **Boscher**$_1$[**] — **Romain** $_2$ **Perrone**$_2$[*] — **Nada** $_3$ **Lasri**$_3$[*] — **Előd** $_4$ **Egyed-Zsigmond**$_4$[*,**] — **Sylvie** $_5$ **Calabretto**$_5$[*,**]

[*] *INSA de Lyon , 20 avenue Albert Einstein, 69100 Villeurbanne*
[**] *Université de Lyon, LIRIS UMR 5250 CNRS*
prenom.nom@insa-lyon.fr

RÉSUMÉ. *La détection automatique des liens d'articles dans la une des journaux en ligne est un sujet très peu étudié, bien qu'il s'agisse d'une étape clé pour extraire des informations à partir d'un journal. Dans cette étude, nous présentons une approche permettant de détecter les liens des articles présents sur un large éventail de pages web de journaux. Notre méthode s'appuie sur des techniques d'apprentissage automatique basée sur le texte des liens et les attributs DOM des balises liens. Notre méthode prend également en compte la notion d'éphémérité des articles de journaux, ainsi qu'un ensemble d'heuristiques permettant d'améliorer le rappel de notre approche par itérations sur la durée. L'algorithme produit une liste d'URL d'articles en sortie. Un des avantages de notre approche est qu'elle ne repose pas sur la structure initiale du DOM : elle donne donc de très bons résultats même face à des mises en page totalement différentes et évolutives.*

ABSTRACT. *In recent years, relatively little research has been done on the topic of online article detection links, although it is a key step in any news scraping task. In this article, we present a new approach to efficiently detect article links on a wide range of newspaper webpages. Our algorithm detects the links found on a webpage, discards the ones that are irrelevant thanks to a specially trained ML-classifier and hands down the DOM of the final elements to a clustering algorithm. The algorithm outputs a series of XPATH expressions that can then be used to retrieve the articles. We consider the online presence of the links to add an additional, time based, filter. Because our approach doesn't rely on the initial DOM structure, it can be applied on a wide variety of websites and can adapt to changes in their structure.*

MOTS-CLÉS : *recherche d'information, web scraping, classification*

KEYWORDS: *information retrieval, web scraping, classification*

1. Introduction

L'accès automatisé aux articles en ligne est un sujet sensible et important. Cela peut servir, par exemple, à automatiser des revues de presse. Afin d'extraire des informations de la presse en ligne, il est important de pouvoir y accéder. Alors que des services en ligne type Europresse existent, ils sont payants et peu propices à un traitement automatique. Nous souhaitons proposer une méthode qui permet de collecter des articles de journaux en ligne de manière automatique facilement. Pour cela, il faut au préalable identifier les liens des articles publiés par les journaux en ligne. Plusieurs techniques existent, à savoir : **1.** la détection d'articles à partir du flux RSS ; **2.** la détection d'articles à partir du site web principal du journal, dite *la une*. Une deuxième étape d'extraction du contenu des articles suit cette première phase.

L'un des inconvénients majeurs de la détection par flux RSS (Han *et al.*, 2009) est qu'elle est limitée à un petit nombre de journaux. Afin de pallier cette problématique, nous avons choisi de mettre en place une méthode d'extraction automatique de liens pointant vers les articles issus de la une des journaux. Extraire des liens d'articles à partir du contenu HTML d'une page est une tâche difficile, car ces derniers se trouvent souvent mêlés à du contenu annexe comme les menus de navigation, les encarts publicitaires et les en-têtes et pieds de page. Par ailleurs, la mise en page et la structure du DOM HTML des sites web des journaux varient souvent d'un journal à l'autre, ce qui complique encore l'extraction de liens vers les articles.

Dans ce papier, nous proposons une nouvelle approche permettant de détecter automatiquement les liens vers des articles à partir de la une des journaux.

2. Travaux connexes

Peu de travaux ont été publiés concernant l'extraction automatique des liens d'articles dans la une des journaux.

Jusqu'à présent, la quasi-majorité des recherches dans le domaine de l'extraction d'information s'est concentrée sur l'extraction du titre et du contenu de l'article **et non sur la détection des liens en amont**. La bibliothèque Neswpaper3k (Ou-Yang, 2019) offre une solution d'extraction du titre et du texte des articles qui fonctionne sur environ 70% des journaux que nous avons testés. (Zhou *et al.*, 2007) ont analysé les pages web sous la forme d'arbres visuels et ont extrait des descripteurs pour les différents blocs dans le but d'entraîner un algorithme d'apprentissage automatique pour arriver à un wrapper performant. Les auteurs de (Subercaze *et al.*, 2016), recherchent des sous-arbres fréquents dans la structure DOM des pages web afin d'extraire des commentaires d'utilisateurs. Notre méthode cherche également des structures qui se répètent, mais nous nous concentrons essentiellement sur les attributs et sur des hiérarchies de noeuds peu profondes.

Deux approches existent pour l'extraction automatique des liens d'articles. Ces dernières reposent sur la génération d'un wrapper qui permet d'identifier la balise lien.

La technique de *wrapper induction* (Kushmerick *et al.*, 1997) utilise l'apprentissage supervisé pour apprendre des règles d'extraction à partir d'un ensemble d'exemples annotés à la main. Les descripteurs extraits sont propres à chaque site internet : attributs d'une balise DOM, taille du texte, etc. Les inconvénients de cette technique sont : - le temps nécessaire à l'annotation des exemples à la main : il est nécessaire d'entraîner un wrapper pour chaque site internet; et - la maintenance du wrapper: en cas de changement sur le site internet, le wrapper doit être entraîné à nouveau.

À cause de la charge de travail importante qu'implique cette méthode, il est difficile d'extraire des informations à partir de nouveaux sites internet, étant donné que chaque nouveau site a sa propre structure et demande ici une annotation manuelle permettant d'entraîner un wrapper.

Une autre approche consiste à utiliser l'apprentissage non supervisé pour générer le wrapper. On parle d'*automated wrapper generation* (Xia *et al.*, 2011). L'extraction automatisée est possible car la plupart des objets web suivent des modèles fixes.

La méthode que nous avons mise en place génère automatiquement un wrapper (*automated wrapper generation*) capable de détecter les liens d'articles. Cela nécessite une annotation manuelle de liens sur quelques sites de journaux par langue supportée, une seule fois, pour l'entraînement d'un modèle TAL (voir section 3.1). Notre wrapper peut être utilisé sur des sites internet autres que ceux utilisés pour l'entraînement du modèle, à condition qu'ils soient dans la même langue.

Notre méthode offre un coût de maintenance réduit et une plus grande flexibilité face à des structures de pages différentes et changeantes.

3. Extraction des liens

Notre méthode se base sur trois approches complémentaires. L'une exploite les caractéristiques du texte des liens pointant vers des articles individuels, contenant en général un titre. Nous avons mis en place un modèle capable de reconnaître un titre d'article parmi d'autres textes de liens. La deuxième approche se base sur des caractéristiques des éléments DOM de la page HTML contenant les liens. Nous nous sommes rendus compte que l'étude de l'élément lien (balise <a>), son élément DOM parent et ses attributs fournissent des informations utiles pour pouvoir distinguer un lien vers un article d'autres liens. La troisième approche intègre la notion d'éphémérité des liens vers les articles sur la *une* du journal dans le temps. Nous avons observé qu'en visitant plusieurs fois la même page sur une durée de plusieurs jours, nous pouvons estimer la durée de vie des liens et ainsi améliorer leur classification.

3.1. *Approche basée sur le contenu textuel des liens*

Le principal problème concernant le scraping des *une* de journaux est la distinction entre un lien pertinent qui pointe vers un article individuel, et un lien non pertinent qui pointe vers des sous-rubriques, impressum, partenaires, ... Lorsque l'on parle de lien pertinent, nous définissons cela comme un lien du site de la *une* de journal qui mène vers la page web d'un article unique.

3.1.1. *Classification de textes*

L'analyse qualitative de la construction des pages web grâce à leur code HTML nous a permis de noter à quel point les pages de journaux en ligne sont construites de façon hétérogène. Tenter de reconnaître un lien pertinent en se basant sur la construction de la page, l'agencement et l'imbrication des balises aurait imposé de prendre en compte trop de cas particuliers. Cependant, en observant la structure des pages HTML des sites de journaux, une constante émerge : un lien, l'attribut *href* d'une balise de type <*a*>, est toujours associé à un texte, une phrase qui la plupart du temps est le titre de l'article pointé. Nous cherchons alors à classifier ce lien en nous basant sur le texte qu'il contient.

```
<a class="css-kej3w4"
href="https://www.nytimes.com/2020/12/15/us/coronavirus-vaccine
-doses-reserved.html">
With First Dibs, Rich Countries Have ‘Cleared the Shelves’
</a>
```

Ainsi, comme le montre l'exemple ci-dessus, le lien `https://www.nytimes.com/2020/12/15/us/coronavirus-vaccine-doses-reserved.html` a pour texte associé "**With First Dibs, Rich Countries Have ‘Cleared the Shelves**'".

A partir de ces textes, nous avons réalisé plusieurs classifications. Le but est de mettre au point un modèle capable de faire la distinction entre un titre d'article de journal et une phrase non pertinente. Par exemple, la phrase **"For the first time, Trump says he'll go if Electoral College votes for Biden"** doit être interprétée comme un titre d'article, et la phrase **"View your subscriber options"** doit être considérée comme une phrase non pertinente.

3.1.2. *Création d'un jeu de données d'apprentissage*

Pour mettre au point un modèle, nous avons eu besoin de données d'apprentissage. La tâche spécifique de scraping de *une* de journaux n'ayant pas encore été fortement explorée, nous n'avons pas pu trouver de jeu de données sur lequel entraîner notre modèle.
Pour pallier cela, nous avons construit nous même un jeu de données d'apprentissage pour des journaux anglophones.

À partir de chaque site, nous avons extrait tous les liens présents dans les balises <a>, et associés à du texte non vide. En récupérant ces couples (lien, texte), on insère

les données dans un document csv, qui sera validé à la main après une pré-annotation automatique.

À l'issue de ce travail, nous obtenons une classification binaire : la classe des liens d'articles de presse, annotée d'un 1 et la classe des liens non pertinents, annotée d'un 0.

Cette classification assistée sera nécessaire pour chaque langue que l'on souhaitera supporter, et il faudra entraîner un modèle distinct pour chaque langue. Alors, nos modèles seront capables de classer automatiquement les liens en se basant sur leur contenu textuel.

Pour entraîner notre modèle anglophone, nous avons choisi quatre sites web : *https://www.nytimes.com*, *https://www.reuters.com*, *https://www.foxnews.com*, *https://www.theguardian.com*.

Nous avons ensuite testé notre modèle sur *https://https://www.economist.com/*

Notre jeu de données totalise les nombres de liens précisés dans le tableau 1.

	Nombre de liens
Jeu d'entraînement	448
Jeu de validation	114
Jeu de test	77
Total	639

Tableau 1 – Taille du jeu de données de journaux anglophones obtenu

Nous avons implémenté deux techniques d'apprentissage pour classifier les liens.

3.1.3. *Modèle Naive Bayes*

Pour commencer, nous avons choisi d'appliquer une méthode d'apprentissage supervisé sur notre jeu de données. Face au large choix d'algorithmes existants, nous avons choisi de commencer par une des méthodes traditionnelles dans l'apprentissage automatique, le modèle Naive Bayes.
Pour entraîner notre modèle, nous nous appuyons sur un sac de mots : un vecteur de comptage pour chaque phrase. Nous appliquerons la méthode de pondération TD-IDF pour normaliser les entrées, qui seront ensuite fournies au modèle pour l'entraînement.

3.1.4. *Modèle BERT*

Nous avons également implémenté une méthode de classification basée sur des réseaux de neurones, afin de comparer ses performances à celles du modèle Naive Bayes; nous avons choisi d'étudier le classifieur BERT (Devlin *et al.*, 2019).
En effet, ce classifieur domine l'état de l'art dans une grande variété de tâches liées au TAL. À partir d'un modèle BERT pré-entraîné, nous entraînons un modèle BERT personnalisé grâce à notre jeu de données annoté (3.1.2).
La taille des textes que nous traitons est de l'ordre de quelques mots, nous avons donc décidé de choisir un modèle de type BertForSequenceClassification (Sanh *et al.*, 2019).
Vue la taille des données dont nous disposons, nous avons décidé de travailler en mode distilled avec le modèle TFDistilBertForSequenceClassification, qui est plus rapide et léger que celui mentionné précédemment. Deux classes seront retenues donc deux labels (0 et 1).Nous obtenons, pour notre jeu de donnés d'entraînement et de validation, les résultats du tableau 2.

Le classifieur basé sur BERT renvoie un score de prédiction entre 0 et 1. Expérimentalement, nous considérons un seuil minimal de 0.9 pour qu'un titre soit considéré comme un titre d'article.

	train accuracy	val accuracy
Epoch 1	0.829	0.920
Epoch 2	0.964	0.947
Epoch 3	0.976	0.947

Tableau 2 – Résultats intermédiaires du classifieur BERT lors de l'entraînement

Nous notons de bonnes performances à l'entraînement, avec une précision de 0.976 au bout de trois epoch. De plus, nous remarquons que la précision pour le jeu de donnés de validation est proche de la précision pour le jeu de données d'entraînement (0.976 et 0.947), ce qui permet d'appuyer l'efficacité de notre modèle.

3.1.5. *Application des modèles sur le jeu de test*

Nous effectuons un comparatif des deux modèles proposés en les appliquant au jeu de données de test décrit en 3.1.2.

	Précision	Rappel	F1
Naive Bayes	0.690	0.808	0.744
BERT	0.913	0.808	0.857

Tableau 3 – Comparaison des résultats obtenus pour les méthodes Naive Bayes et BERT sur le jeu de données de test

Le tableau 8 nous montre ici que la précision du modèle basé sur BERT est supérieure à celle du modèle Naive Bayes, passant de 0.690 à 0.913. En effet, le modèle basé sur Naive Bayes va détecter 29 liens articles, mais dont seulement 21 d'entre eux sont réellement des liens d'articles. En parallèle, BERT va en récupérer seulement 23, avec 21 vrais liens d'articles. En ce qui concerne le rappel, il est le même pour les deux méthodes; sur les 26 liens à récupérer, les deux méthodes vont manquer 5 de ces liens.

3.1.6. *Règle explicite*

Nous avons observé que dans la majorité des cas, les titres d'articles sont composés de 4 mots ou plus. Les liens non pertinents, au contraire, sont la plupart du temps associé à un texte succinct, avec des impératifs (Subscribe / Read full edition) et parfois seulement un mot, le nom d'une section. Pour améliorer la performance de notre algorithme, nous avons donc mis en place la condition suivante : **tout lien associé à un titre de moins de 4 mots est considéré comme non pertinent**, quelle que soit la prédiction réalisée par le modèle de classification. Nous pouvons observer

	Précision	Rappel	F1
Naive Bayes	0.690	0.808	0.744
Naive Bayes + règle	0.944	0.692	0.799
BERT	0.913	0.808	0.857
BERT + règle	1.000	0.731	0.844

Tableau 4 – Comparaison des résultats obtenus pour les méthodes Naive Bayes et BERT et la règle explicite sur le jeu de données de test

sur le tableau 4 que pour la méthode de Naive Bayes, la précision du classifieur est améliorée par la règle précédemment définie, passant de 0.690 à 0.944, ce qui est notable. Dans le cas du classifieur basé sur BERT, nous obtenons une précision de 1.0, au détriment du rappel qui diminue de 0.808 à 0.731.

3.2. *Approche basée sur les attributs DOM*

En complément de l'approche basée sur le contenu textuel des liens, nous avons implémenté une seconde approche qui exploite les attributs des éléments de la structure du DOM HTML. Nous identifions dans l'arbre HTML de la page ciblée les balises *<a>* contenant les liens extraits en 3.1., puis nous extrayons des caractéristiques DOM de ces liens que nous utiliserons dans le cadre d'un apprentissage non supervisé (clustering). Nous générons en sortie une liste d'expressions XPATH permettant de récupérer les liens d'article.

3.2.1. *Prétraitement*

Pour préparer l'étape de classification, il est nécessaire d'extraire les caractéristiques de l'arbre DOM de la page web. Ces descripteurs s'apparentent à un ensemble de couples (attribut, valeur), où *attribut* peut être de type : class, data, data-*, id ou avoir une valeur spécifiquement définie par l'utilisateur.

```
<a class="css-kej3w4"
href="https://www.nytimes.com/2020/12/15/us/coronavirus-vaccine
-doses-reserved.html"
data-uri="nyt://article/caecb59e-9fde-5ba0-9c24-8a85680e14e8"
data-story="nyt://article/caecb59e-9fde-5ba0-9c24-8a85680e14e8"
data-visited="">
</a>
```

Dans l'exemple suivant, on a :

balise	attribut	valeur
a	class	css-kej3w4
a	data-uri	nyt://article/caecb59e-9fde-5ba0-9c24-8a85680e14e8
a	data-story	nyt://article/caecb59e-9fde-5ba0-9c24-8a85680e14e8
a	href	https://www.nytimes.com/2020/12/15/us/coronavirus-vaccine-doses-reserved.html

Nous allons réaliser un apprentissage supervisé à partir des attributs des éléments <*a*> et de leur parent (div, span, li, ...). Tous ces attributs sont récupérés et ajoutés à la structure de données du tableau 5 que nous appellerons tableau *preprocessing*.

attribut	valeur	nombre
class	headline-link	19
class	ds-link-with-arrow	4
class	ds-link-with-arrow–minor	3
class	current-edition__flashes-item	3
data-analytics	graphic_detail:headline	1
data-analytics	economist_today:headline_5	1
data-analytics	weekly_edition:flash_2	1
data-analytics	economist_today:headline_7	1
data-analytics	economist_today:headline_3	1

Tableau 5 – Exemple de prétraitement réalisé à partir de la une de The Economist.

La colonne *nombre* comptabilise le nombre de fois où le couple (attribut,valeur) apparaît au niveau de la balise <*a*> ou de son parent. Cette valeur est essentielle, car elle permet de supprimer les attributs inutiles. Ce processus de suppression est détaillé

dans la section 3.2.2. Dans cet exemple la balise: a été rencontrée 19 fois.

Comme l'attribut $href$ est unique pour chaque article, il ne permet pas d'identifier des clusters. Nous ne le prendrons pas en compte dans nos traitements.

3.2.2. *Suppression des attributs inutiles*

Nous recherchons des articles présentant des attributs similaires (ex. class="headline-link"), c'est à dire des couples (attribut, valeur) présents plusieurs fois dans le document. Si un couple (attribut, valeur) n'est présent qu'une seule fois, nous pouvons l'éliminer. Pour chaque élément <a> du DOM : pour chaque couple *(attribut, valeur)* de l'élément et de son parent: on parcourt la page à la recherche de cette valeur, en s'assurant que l'attribut correspond. Si le compteur est <= 1, l'attribut est unique. Il peut être supprimé de la liste des attributs à considérer. Mais si la valeur contient un ou plusieurs nombres (ex. `economist_today:headline_3`), on parcourt à nouveau *preprocessing* en remplaçant les chiffres par l'expression regex « [0-9]+ ». En effet, les balises
`<a data-analytics="economist_today:headline_3">` et
`<a data-analytics="economist_today:headline_7">` peuvent être considérées comme similaires au chiffre près. Si le compteur est toujours <= 1 on supprime l'attribut de la liste des attributs à considérer.

Cette étape permet de se débarrasser des couples *(attribut, valeur)* n'ayant pas une portée générale.

3.2.3. *Clustering*

A partir des couples (attribut, valeur) retenus pour chaque élément <a> du DOM, nous cherchons à identifier des clusters.

Nous créons deux vecteurs (v_a et v_{parent}) représentant les deux niveaux de l'arborescence du DOM. Nous aurions pu considérer l'ensemble des couples (attribut,valeur) de l'arborescence DOM jusqu'à la balise <a>, mais en pratique les attributs de <a> et de son parent suffisent.

Chaque vecteur peut s'exprimer comme une liste finie d'éléments formatés ainsi : $balise.niveau.attribut = valeur$ avec $balise$ = p, div, a, h (équivalent à h1, h2, h3, h4, h5 et h6); et $niveau$ = 0 pour la balise <a> et 1 pour la balise parent.

Par exemple pour une balise parent <h2 name="test" class="hello"></h2>, nous obtenons le vecteur $v_{(texte)parent}$ suivant : (h.1.name=test, h.1.class=hello)

Pour obtenir un vecteur au format numérique, nous générons un sac de mots à partir de l'ensemble des $balise.niveau.attribut = valeur$ générées.

```
{
a.0.class=ds-link-with-arrow--minor,
a.0.class=ds-link-with-arrow,
a.0.data-analytics=in_context:headline_[0-9]+,
```

```
a.0.class=headline-link,
a.0.data-analytics=us_[0-9]+_election:headline_[0-9]+,
a.0.data-analytics=weekly_edition:flash_[0-9]+,
a.0.data-analytics=economist_today:headline_[0-9]+,
li.1.class=current-edition__flashes-item,
a.0.data-analytics=readers_favourites:headline_[0-9]+
}
```

Nous exprimons alors le vecteur v en encodage 1 parmi n, à partir de v_a et v_{parent}. La balise

```
<li class="current-edition__flashes-item">
    <a href="..." class="headline-link"> </a>
<li>
```

sera codée sous la forme :

```
(0,0,0,1,0,0,0,1,0)
```

Nous pouvons maintenant chercher à identifier des clusters à partir des vecteurs v générés.

Le nombre de clusters étant inconnu à l'avance on utilisera la méthode DBScan avec $eps = 0.5$, $min\_samples = 2$, obtenus expérimentalement. La similarité cosinus étant fréquemment utilisée pour mesurer la similarité entre deux vecteurs, nous l'utiliserons comme métrique de distance.

	cluster
['a.0.data-analytics=readers_favourites:headline_[0-9]+', 'a.0.data-analytics=economist_today:headline_[0-9]+', 'a.0.class=headline-link', 'a.0.data-analytics=us_[0-9]+_election:headline_[0-9]+', 'a.0.data-analytics=in_context:headline_[0-9]+']	1
['a.0.data-analytics=weekly_edition:flash_[0-9]+', 'li.1.class=current-edition__flashes-item', 'a.0.class=ds-link-with-arrow–minor', 'a.0.class=ds-link-with-arrow']	2

Tableau 6 – Clusters identifiés à partir de la une de The Economist.

Le tableau 6 montre les balises, attributs, valeurs apparaissant au moins une fois dans un cluster donné.

3.2.4. *Simplification du patron*

Une fois les clusters identifiés, on cherche à réduire la variance intra-groupe. Pour ce faire, on fixe un *seuil = 0.75*. Pour chaque dimension, on calcule la valeur moyenne

des vecteurs du cluster pour cette dimension. Si cette valeur moyenne est $> seuil$ on conserve les descripteurs associées à cette dimension. Sinon, on simplifie le patron (pattern) en ignorant ces descripteurs afin de réduire la variance intra-groupe.

Le tableau 7 correspond aux balises, attributs, valeurs retenues après simplification. Le tableau contient moins d'attributs que celui présenté dans le tableau 6: c'est exactement ce que l'on recherchait.

3.2.5. *Génération du XPATH*

Pour finir, on génère une expression XPATH à partir du patron simplifié comme illustré dans la colonne XPATH du tableau 7. Les expressions XPATH générées permettent de récupérer les liens d'articles sur la page. Nous utilisons la fonction *contains*, car les attributs peuvent avoir des valeurs composées.

attribut	XPATH	cluster
['a.0.class=headline-link']	//a[contains(@class, 'headline-link')]	1
['a.0.class=ds-link-with-arrow']	//a[contains(@class, 'ds-link-with-arrow')]	2

Tableau 7 – Clusters simplifiés (The Economist)

3.3. *Approche basée sur l'éphémérité temporelle des liens*

En complément des méthodes basées sur le contenu textuel des liens et sur les attributs DOM, nous avons implémenté une troisième approche basée sur la notion d'éphémérité des liens. Nous partons du principe que les articles à la une d'un journal ont une longévité (i.e. un temps limité au cours duquel l'article paraîtra sur la page principale du site), et sont amenés à apparaître ou disparaître de sa page principale au fil du temps. Cette méthode s'appuie ainsi sur la réalisation d'itérations à intervalle régulier de scraping sur la page principale d'un même journal en ligne sur une période de plusieurs jours, ainsi que sur l'élaboration d'heuristiques basées sur la longévité des liens, permettant ainsi de repêcher des liens d'articles potentiellement éliminés par les deux méthodes précédentes ou d'éliminer des liens qui seraient présents trop longtemps.

3.3.1. *Renforcement des méthodes Text Based et DOM*

Cette méthode est complémentaire aux méthodes basées sur le texte et sur les attributs DOM décrites dans cet article (3.1.4 et 3.2.3)

Dans un premier temps, nous proposons de combiner les méthodes Text-Based et DOM de la manière qui suit : le modèle BERT permet de détecter un ensemble de liens d'article sur la page; ces liens sont ensuite fournis à la méthode DOM

afin de réaliser un apprentissage non supervisé permettant d'extraire des expressions XPATH, qui peuvent être ainsi appliquées de manière à repêcher ou exclure des liens mal catégorisés par le modèle BERT. Cette méthode offre une meilleure précision que les méthodes Text-Based et DOM séparées, au prix d'une perte de rappel. Nous cherchons ici à déterminer des heuristiques permettant d'améliorer le rappel de la combinaison de ces deux approches.

Nous expérimentons cette approche hybride sur des journaux francophones.

Nous entraînons un modèle BERT sur les sources suivantes (80% entrainement, 20% validation) :

– Le Monde : *https://www.lemonde.fr* - 1200 liens annotés

– L'Équipe : *https://www.lequipe.fr* - 1000 liens annotés

– Europe 1 : *https://www.europe1.fr* - 600 liens annotés

En utilisant le modèle BERT entrainé, nous testons la méthode hybride Text-Based + DOM sur les données suivantes :

– Médiapart : *https://www.mediapart.fr* - 250 liens

– FranceInter : *https://www.franceinter.fr* - 400 liens

Source	Précision	Rappel
Médiapart	0,96	0,78
France Inter	0.897	0,796

Tableau 8 – Résultats obtenus pour la méthode Text Based + DOM sur des journaux francophones

Nous voyons que cette approche offre une bonne précision pour nos deux journaux de test; nous gagnerions toutefois à obtenir un meilleur rappel pour pouvoir exploiter cette approche de manière efficace sur davantage de journaux.

La nouvelle méthode prenant en compte l'éphémérité des liens, que nous décrivons dans la section suivante, devra permettre d'améliorer le rappel de la méthode Text Based + DOM.

3.3.2. *Heuristique basée sur l'éphémérité des liens*

Nous définissons une heuristique naïve : un lien éphémère (qui apparaît/disparaît de la page d'accueil d'un journal entre deux itérations de scraping) a de grandes chances d'être un article de presse et **doit être repêché** si la méthode BERT + DOM l'a précédemment éliminé.

Afin de dimensionner le concept d'éphémérité d'un article, nous scrapons 9 quotidiens francophones à raison d'une itération toutes les 2 heures pendant 7 jours; nous mesurons pour chaque journal la longévité d'un lien d'article en page d'accueil, pour

déterminer au bout de combien de temps on pourra affirmer l'éphémérité de la plupart des liens scrapés, et les repêcher le cas échéant.

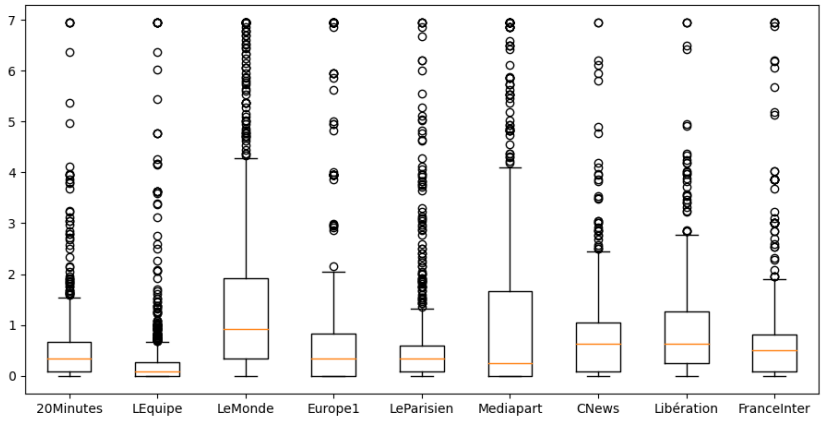

Figure 1 – Distribution de la longévité (en jours) des articles (vrais positifs et faux négatifs) scrapés sur une période de 7 jours pour des journaux francophones

Nous mesurons ainsi la longévité des liens catégorisés comme vrais positifs ou faux négatifs par l'approche Text-Based + DOM . La figure 1 nous montre que la plupart des liens d'article ont une longévité inférieure à 7 jours, donc qu'une période de 7 jours de scraping permet de confirmer l'éphémérité de la plupart des liens d'article. On observe toutefois qu'une minorité de liens ne disparaissent pas encore au bout de 7 jours. Cela représente de 0,5% des articles pour le journal L'Équipe à 4,5% des articles dans le cas du journal Le Monde. Ces liens ne seront pas considérés éphémères et ne seront alors pas repêchés, ce qui peut expliquer que cette méthode, à l'échelle d'une semaine, ne permette pas d'obtenir un rappel de 1.

3.3.3. *Renforcement de l'heuristique*

Cette heuristique naïve permet de repêcher un certain nombre de faux-négatifs et d'augmenter le rappel de l'approche. Toutefois, l'éphémérité d'un lien n'implique pas qu'il s'agisse d'un lien d'article, et cette hypothèse peut engendrer un certain nombre de faux positifs qui impactent négativement la précision du modèle, par exemple des **liens vers des commentaires** *(https://www.20minutes.fr/article/2973183/commentaires)*, des **liens vers des podcasts** *(https://www.franceinter.fr/emissions/la-drole-d-humeur-de-guillermo-guiz)* ou encore des **liens vers des rubriques temporaires** *(https://www.lemonde.fr/loi-securite-globale/)*

Pour éliminer la plupart de ces faux positifs, nous pouvons modifier notre heuristique et appliquer de nouvelles règles. Nous repêchons alors tous les liens éphémères, **sauf si** :

– L'URL est une sous-chaine d'une autre URL du jeu de donnée. Cela signifie généralement que le lien pointe vers une rubrique et ne doit pas être repêché.

– En remplaçant les nombres de 3 chiffres ou plus dans l'URL par le token **number** , le lien n'est pas unique dans le jeu de données.

– Le score BERT associé au titre était initialement trop faible. Nous ne repêchons alors pas les articles dont le titre a obtenu un score BERT inférieur ou égal à 0,2

4. Résultats

4.1. *Comparaison Text-Based, DOM et Text-Based + DOM sur journaux anglophones*

Nous souhaitons comparer les performances de 3 méthodes d'extraction des liens: méthode basée sur le texte (Text based) (3.1.4), méthode basée sur les attributs DOM (DOM) (3.2.3), méthode basée sur la combinaison des deux méthodes (Text + DOM)
.
Pour cela, nous avons choisi d'extraire les unes de quatre sites de journaux :

– Le New York Times: `https://www.nytimes.com`

– The Economist: `https://www.economist.com`

– The Financial Times: `https://www.ft.com`

– The Times UK `https://www.thetimes.co.uk`

Afin de garantir des résultats fiables, toutes les méthodes décrites ici simulent l'interaction du site web avec un navigateur web (le défilement jusqu'en bas de la page), afin de pouvoir récupérer la totalité des articles chargés dynamiquement (*lazy-loading*).

Text-based. L'ensemble des balises <a> présentes sur la page sont récupérées (à l'exclusion de celles présentes dans le header et le footer). Il s'agit ensuite de réaliser des prédictions : lien vers un article ou non, à partir du texte contenu dans ces balises (3.1.4)

DOM. L'ensemble des balises <a> présentes sur la page sont récupérées (à l'exclusion de celles présentes dans le header et le footer). Il s'agit ensuite d'extraire des attributs DOM pour effectuer un apprentissage non supervisé (clustering) et détecter les liens vers les articles (3.2.3).

Text-Based + DOM. Les deux approches (Text based et DOM) sont combinées.

L'ensemble des balises <a> présentes sur la page sont récupérées (à l'exclusion de celles présentes dans le header et le footer). Puis tous les liens détectés par la méthode

Text based sont fournis à la méthode DOM, qui récupère à son tour les liens vers les articles après un apprentissage non supervisé (clustering).

On observe ainsi que quel que soit le site, la méthode Text-Based + DOM offre une meilleure précision que la méthode Text-Based seule, et s'avère être l'approche avec la meilleure précision dans le cas du Financial Times

	Prec. FT	Rap. FT	Prec. T	Rap. T
Text-Based	0.956	**1.0**	0.906	**0.943**
DOM	0.956	0.93	**0.926**	0.927
Text-Based + DOM	**0.985**	0.92	0.915	0.926

Tableau 9 – Précision (Prec.) et Rappel (Rap.) pour le Financial Times (FT) et le Times UK (T)

4.2. *Comparaison des méthodes Text-Based + DOM et Text-Based + DOM + Heuristique temporelle sur journaux francophones*

Nous conduisons une analyse similaire sur un ensemble de journaux francophones, scrapés à raison d'une itération toutes les 2 heures pendant 7 jours. Nous annotons l'ensemble des liens collectés pour mesurer la précision et le rappel de nos méthodes sur ces journaux. La méthode Text-Based + DOM offre un rappel un peu moins bon sur les journaux francophones que nous avons sélectionnés :

– 20 Minutes: `https://www.20minutes.fr`

– L'Equipe: `https://www.lequipe.fr`

– Le Monde: `https://www.lemonde.fr`

– Le Europe 1: `https://www.europe1.fr`

– Le Parisien: `https://www.leparisien.fr`

– Médiapart: `https://www.mediapart.fr`

– CNews: `https://www.cnews.fr`

– Libération: `https://www.liberation.fr`

– France Inter: `https://www.franceinter.fr`

En revanche, la méthode Text-Based + DOM combinée à l'ensemble d'heuristiques prenant en compte l'éphémérité des liens (3.3.2 et 3.3.3) permet de repêcher un certain nombre de liens et ainsi améliorer le rappel pour la plupart des journaux analysés (jusqu'à +0.16 de rappel pour *Libération*).

Source	Prec. T+D	Rap. T+D	Prec. T+D+H	Rap. T+D+H
20 Minutes	0,994	0,892	0,993	**0,940**
L'Équipe	0,998	0,760	0,998	**0,994**
Le Monde	0,988	0,944	0,988	0,944
Europe 1	0,960	0,889	0,961	**0,912**
LeParisien	0,998	0,978	0,998	**0,992**
Mediapart	0,978	0,796	0,980	**0,921**
CNews	0,969	0,829	0,964	**0,947**
Libération	0,986	0,674	0,980	**0,839**
FranceInter	0,918	0,800	0,917	**0,865**

Tableau 10 – Comparatif de la précision et du rappel de la méthode Text + DOM sans (T+D) et avec heuristiques (T + D + H) liées à l'éphémérité des liens pour des journaux francophones

5. Conclusion et perspectives

Dans cet article, nous avons mis en oeuvre un système d'extraction automatique de liens d'articles de presse, combinant trois approches de classification de liens d'articles provenant de *une* de journaux. Nous avons appliqué notre méthode à de vrais journaux en ligne, afin d'évaluer sa capacité à discerner un lien d'article d'un lien non pertinent. L'approche que nous proposons s'avère générique et compatible avec la quasi-totalité des sites de journaux en ligne. Elle requiert l'annotation assistée d'un jeu de données de quelques centaines de liens dès lors que l'on souhaite supporter une nouvelle langue. Nous avons crée des modèles pour le Français, Anglais, Espagnol, Hongrois et Allemand avec, en moyenne, 30 minutes de temps nécessaire par langue pour une personne.

Dans l'objectif d'analyser continuellement et automatiquement le contenu de la presse en ligne, nous envisageons par la suite d'étudier l'extraction des caractéristiques des articles identifiés via la méthode que nous venons de décrire, en mettant en place une approche permettant d'extraire et d'exploiter le titre, la catégorie (si disponible), le texte, les entités nommées (Named Entity Recognition) et les relation entre entités nommées de chaque nouvel article. Dans la continuité de notre travail sur la détection de liens d'article, nous souhaiterons ainsi extraire l'information des articles provenant de tous types de journaux en ligne, de manière générique, tout en garantissant un coût de maintenance réduit pour l'inclusion de nouvelles sources d'informations. Des solutions d'extraction de contenu d'articles basées sur des techniques d'apprentissage automatique et indépendantes de la mise en page du journal (Wang *et al.*, 2009) nous permettraient de mettre en place une méthode d'extraction générique pour l'ensemble des journaux que nous souhaiterions analyser à l'avenir.

6. Bibliographie

Devlin J., Chang M.-W., Lee K., Toutanova K., “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.”, *in* J. Burstein, C. Doran, T. Solorio (eds), *NAACL-HLT (1)*, Association for Computational Linguistics, p. 4171-4186, 2019.

Han H., Noro T., Tokuda T., “An Automatic Web News Article Contents Extraction System Based on RSS Feeds”, *J. Web Eng.*, vol. 8, n° 3, p. 268–284, September, 2009.

Kushmerick N., Weld D. S., Doorenbos R. B., “Wrapper Induction for Information Extraction”, *IJCAI*, 1997.

Ou-Yang L., “Newspaper3K Python Library”, , `"https://newspaper.readthedocs.io/en/latest/"`, 2019. Online; accessed 11 March 2021.

Sanh V., Debut L., Chaumond J., Wolf T., “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”, *CoRR*, 2019.

Subercaze J., Gravier C., Laforest F., “Extraction de commentaires utilisateurs sur le Web”, *Revue des Nouvelles Technologies de l’Information*, vol. Extraction et Gestion des Connaissances, RNTI-E-30, p. 237-242, 2016.

Wang J., Chen C., Wang C., Pei J., Bu J., Guan Z., Zhang W., “Can we learn a template-independent wrapper for news article extraction from a single training site?”, p. 1345-1354, 01, 2009.

Xia Y., Yang Y., Zhang S., Yu H., “Automatic Wrapper Generation and Maintenance”, *Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation*, Institute of Digital Enhancement of Cognitive Processing, Waseda University, Singapore, p. 90-99, December, 2011.

Zhou D., Giles C., Zheng S., Li J., “Extracting Author Meta-Data from Web Using Visual Features”, *2007 7th IEEE International Conference on Data Mining Workshops*, IEEE Computer Society, Los Alamitos, CA, USA, p. 33-40, oct, 2007.