Received: 29 September 2016 | Revised: 16 January 2017 | Accepted: 19 January 2017

DOI: 10.1111/coin.12120

ORIGINAL ARTICLE

WILEY Computational Intelligence

From French Wikipedia to Erudit: A test case for cross-domain open information extraction

Fabrizio Gotti | **Philippe Langlais**

RALI, Université de Montréal, Montréal, Canada

Correspondence
Fabrizio Gotti, RALI, Université de Montréal, CP 6128 Succursale Centre-Ville, Montréal, H3C 3J7, Canada.
Email: gottif@iro.umontreal.ca

Funding information
Nuance Foundation

Abstract

In this paper, we describe an open information extraction pipeline based on REVERB for extracting knowledge from French text. We put it to the test by using the information triples extracted to build an entity classifier, ie, a system able to label a given instance with its type (for instance, Michel Foucault is a philosopher). The classifier requires little supervision. One novel aspect of this study is that we show how general domain information triples (extracted from French Wikipedia) can be used for deriving new knowledge from domain-specific documents unrelated to Wikipedia, in our case scholarly articles focusing on the humanities. We believe that the present study is the first that focuses on such a cross-domain, recall-oriented approach in open information extraction. While our system's performance shows room for improvement, manual assessments show that the task is quite hard, even for a human, in part because of the cross-domain aspect of the problem we tackle.

KEYWORDS

entity classification, named entities, natural language processing, open information extraction

Guest editors: Richard Khoury, Christopher Drummond, richard.khoury@ift.ulaval.ca; Christopher.Drummond@nrc-cnrc.gc.ca
Special issue of *Computational Intelligence* based on the 29th Canadian Conference on Artificial Intelligence (CanAI 2016) May 31 to June 3, 2016, Victoria, Canada

1 | INTRODUCTION

Extracting knowledge from a large set of mostly unstructured documents (such as the Web) and organizing it into a knowledge base (KB) is a key challenge in artificial intelligence.[1] Intuitively, such KBs should directly impact the quality of many NLP applications such as question answering or information retrieval. Since the work of pioneers such as Michele Banko and her collaborators,[2] open information extraction (OIE), the task of extracting knowledge from texts without much supervision (especially *not* a prescription of the kind of information to mine), has brought new hope for such an endeavour. It has given rise to a number of exciting realizations, many fostered by major search engine companies. One of the most striking projects is IBM's Watson question answering system,[3] which exploits the information extracted from over 200 million Web pages, and went on to win a 2011 *Jeopardy!* television game show against two (human) champions. The work of Microsoft on the Literome project[4] is another impressive realization, where information mined from scientific articles available in PUBMED* has been exploited for assisting medical researchers.

Many initiatives have been launched for acquiring large repositories of structured semantic knowledge about our world, including FreeBase,[5] YAGO,[6] DBpedia,[7] or more generally the linked open data.[8] Many such repositories are often collaborative. For instance, DBpedia is built automatically from Wikipedia, which is definitely a collaborative effort. A few initiatives are (almost) unsupervised, such as the NELL system,[9] which continuously learns to extract knowledge from Web pages.†

While these repositories are continuously growing, they still suffer from 2 main shortcomings. First, they lack coverage for specialized domains. There does not seem to be many repositories that would be useful for, say, developing a system to answer questions on network protocols. Second, they are mainly English-centric. One might argue that this is not an issue since semantics are not language specific, but this amounts to an oversimplification. Texts in a given language could very well yield some useful information (or points of view) that are glossed over or simply absent from English documents. More practically, concepts in KBs are associated with English strings (eg, ref:label in DBpedia) that systems can locate in texts, which limits portability to other languages. We are aware of multilingual initiatives, such as BabelNet,[10] but their coverage is poor. ConceptNet[11] is another semantic network encoding common sense knowledge in multiple languages. While the fifth incarnation of the tool also derives much of its generic knowledge from the Wiktionary,‡ the initial knowledge required a significant amount of human supervision, around 14 000 contributors to the Open Mind Common Sense project.

This study attempts to tackle some of these shortcomings. We describe a recall-oriented OIE system designed to extract knowledge (triples). We then put it to the test by assessing how general domain knowledge (from French Wikipedia) can be used for deriving new information in domain-specific documents (Érudit, a collection of scholarly papers in the humanities). We believe that the present study is the first one that focuses on such a cross-domain use in OIE. Although a few domain-specific OIE systems have been designed, such as the Literome project aforementioned, they mostly rely on a huge collection of domain-specific texts.

Significantly enough, we create an OIE tool for French text, based on the English tool REVERB.[12] We try to port an English-centric technology to another language and describe the adaptations

*http://www.ncbi.nlm.nih.gov/pubmed

†Some supervision was provided at the very beginning of the project to identify a number of interesting relations, and there is also human feedback after each iteration of the system in the form of ratings on some newly extracted facts.

‡http://conceptnet5.media.mit.edu/

required and the limitations encountered. Hopefully, this may serve as a basis for future adaptations of OIE systems to other languages.

We start by discussing related work in Section 2. We describe in Section 3 our effort to develop an OIE system for the French language. In Sections 4 and 5, we show how it is possible to derive and characterize entity types from triples extracted in Wikipedia and then use these types to classify entity instances found in triples obtained from another corpus. We conclude in Section 6.3.

2 | RELATED WORK

Since the seminal work conducted on TextRunner,[2] several toolkits have emerged to allow OIE in NLP applications. For instance, REVERB[12] relies on part-of-speech tagging and is available for analyzing English text. The WOE extractor[13] was one of the first to propose distant supervision (in their case, they used arguments in Wikipedia infoboxes) to mine patterns of interest. Other extractors, such as OLLIE,[14] exploit the dependency parse of sentences to extract more precise and more diversified relations.

Typically, OIE tools are designed for English, in part because the tools embedded in such a pipeline are more widely available for this language. Porting an extractor to another language has received very little attention. To our knowledge, the first non–English-centric open extractor, named, ExtrHech,[15] is designed to extract tuples from Spanish texts, thanks to regular expressions based on POS labels. Their system offers an accuracy comparable to the level of performance measured on extractors developed for English (at least on the test sets they considered). Our efforts to port REVERB to French are in line with the work that created ExtrHech (for another language). In Gamallo and Garcia,[16] the authors describe ArgOE, a rule-based extractor that they label "multilingual" in the sense that the extractor needs only a dependency parser (with a part-of-speech tagger) in the target language to get ported to this language. They tested their system on English, Spanish, and Portuguese, obtaining satisfactory performances overall. Very recently, Falke and colleagues[17] described an experiment where they ported PropS to German, a rule-based predicate-argument analyzer for English.[18] In their case, they manually adapted the rules built on top of a dependency parser. They observed that 38% of the rules can be easily ported (basically by changing the dependency type, the POS tags and the lemmas to their German equivalents), while 27% of the rule require substantial changes. In any case, it seems that the porting of an OIE system relies heavily on manual interventions through the "translation" of rules and the substitution of English NLP components with their target language counterparts, when they exist.

That being said, an issue with current OIE technology is that many (if not most) of the facts acquired are either uninformative and/or anaphoric, eg, (*she*, *continues*, *her study*). While anaphoric facts may be partially sanitized by coreference resolution, low informativeness is very problematic, in no small part because of the fact that the very measurement of a fact's informativeness is still an unresolved issue. For instance, Zhila and Gelbukh[15] describe the impressive efforts made at Google for building a huge collection of facts from as many sources as possible (including manually curated databases). While their researchers initially gathered 1.6 B tuples, they estimate that only 272 M of those facts are likely relevant. Naturally, 1 way to distinguish good tuples from spurious ones is to rely on the frequency with which they have been extracted in a large collection of texts.[19] However, this strategy is not a panacea: small corpora or domain specific texts may not yield sufficiently redundant triples to lend themselves to frequency-based filtering. This is the case for this study, and our work departs from others in OIE by the fact that we are precisely interested in extracting knowledge from a small, domain-specific collection of texts.

```
V | VP | V(noun | adj | adv | pron | det)*P
V = verb_particle? adv?
P = (prep | particle | inf marker)
```

FIGURE 1 Regular expression used in the (original) English version of the REVERB extraction engine. The special symbols ? and * indicate, respectively, "once or not at all" and "zero or more times." Adapted from Fader et al[12]

3 | PARTIAL ADAPTATION OF REVERB TO FRENCH

The original REVERB Open IE system[12] proceeds in 2 phases, both of which are language dependent.

1. The first stage reads POS-tagged and NP-chunked sentences and produces a (possibly empty) set of extraction triples (*arg1*, *r*, *arg2*), eg, (*President Obama*, *gave a talk at*, *the White House*). *arg1* can be considered as the "subject" of the triple; *r*, as the "relation" or "relation phrase," and *arg2* as the "object." To be extracted, a triple must satisfy a few constraints. A syntactic constraint stipulates that a relation *r* must match a regular expression based on parts of speech (shown in Figure 1). A lexical constraint learned on a large corpus removes overspecified relation phrases (eg, *are only interested in part of the solution for*), by filtering out relation phrases that appear with too few distinct argument pairs in a large corpus. If a relation is located, noun phrases to its right and left must also be present and valid.
2. A second stage uses a logistic regression classifier to filter out dubious or uninformative triples. REVERB's authors selected 19 features that are used to build this confidence function crucial to weed out uninformative and incoherent extractions (at the cost of recall).

Our strategy for porting REVERB to French is based on similar endeavors found in the literature, and described in Section 2. Most authors use an ad hoc approach where (1) they replace the black box NLP components from English to the target language and (2) manually adapt the rules and patterns found for English to the target language. We adopted a similar strategy and tried to be as faithful as possible to the original, English-based architecture while developing a French equivalent.

We wanted to take advantage of the extraction engine already provided by REVERB for our study on French text. In our case, however, we did not implement the French equivalents of the lexical constraint and the classifier. We wanted to keep as many French triples as possible, given that many of the processing steps they undergo in this study amount to filtering and noise elimination. Besides, our goal here is not to extract triples per se but to use their elements for a classification task. Thanks to the high quality of its API, REVERB lends itself very well to porting it to another language. For French, we had to make significant modifications, described below.

The preprocessing steps relying on Apache OpenNLP were adapted to use French statistical models for sentence segmentation, word tokenization, part-of-speech tagging, and noun phrase chunking. These models are trained on a large generic corpus[20] and are freely available on the Web[§] for the OpenNLP framework.

The empirical POS-based regular expression at the heart of relation extraction was changed from the original (shown in Figure 1 for reference) to the one shown in Figure 2, which is the result of our own attempts to capture as many relations as possible on a small development set. Aside from the fact that the French POS tag set differs slightly from the English one, there are other noteworthy differences. Figure 3 shows a sample extraction.

[§]sites.google.com/site/nicolashernandez/resources/opennlp

```
adv? cli* v pp? adv? pp? INF? adv? prep?
INF = (vinf | prep vinf)
```

FIGURE 2 Regular expression used in the REVERB extraction engine for French. adv: adverb, cli: clitic, v: verb, pp: past participle, vinf: infinitive verb

Dans la première partie de Surveiller et punir, Michel Foucault décrit le rôle politique et social du supplice durant l'époque qui précède les réformes pénales de l'âge classique.
(*Michel Foucault*, *décrit*, *le rôle politique*)
(*l'époque*, *précède*, *les réformes pénales de l'âge classique*)

FIGURE 3 A sample sentence extracted from the erudit corpus yields 2 extracted triples

- The English regular expression for the relation phrase is more flexible and more inclusive than its French counterpart. The pattern (noun | adj | adv | pron | det)* present in the English rule is absent from the French one. The latter favors a sequence with optional elements rather than the possible repetition of elements through the * pattern. We preferred a less inclusive approach in French because it greatly reduces the need for the aforementioned lexical constraint present in the English REVERB, at the cost of missing some French verb phrases containing these elements (eg, *faire partie de—be part of*). We felt that the added precision was worth this alteration.
- Optional adverbs at the beginning and end of the French pattern account for the possible negative form (*ne* V *pas—does not* V) of a relation phrase, and the multiple variations of this negative form (*ne* V *guère*, *ne* V *jamais*, etc).
- Clitics are present in the French pattern. Indeed, clitics frequently occur in French verb phrases, eg, *ils s'y sont vus* (*they saw each other there*), whose POS tags are CLI (*ils*) CLI (*s'*) CLI (*y*) V (*sont*) PP (*revus*).

In REVERB, the *arg1s* and *arg2s* are extracted from noun phrase chunks to the (respectively) left and right of the relation phrase. Since there may be numerous argument candidates to the left or right of such a relation phrase, handcrafted rules are necessary to select a single *arg1* and a single *arg2* from the list of candidates. For *arg1*s, both French and English implementations ignore existential subjects (eg, *there*), wh—words (eg, *who*), prepositions, and some pronouns. The French version features an additional exclusion rule: whenever a sentence like *Les immigrants établis en Chine expédient de l'argent—Immigrants settled in China send money*, the *arg1* candidate *Chine* is recognized as a complement and rejected in favor of *immigrants*. In this case, the triple becomes (*immigrants*, *send*, *money*). The rules processing the *arg2*s are almost identical in French and English and eliminate wh—words in the candidates.

3.1 | A canonical representation for relations

Since the relations in the extracted triples play a major role in the classification task we describe in Section 5, limiting the number of superficial variations for a single relation is desirable. For instance, in English, the relations in (*apple*, *fell from*, *tree*) and (*apple*, *falling from*, *tree*) could be represented by the infinitive form *fall from*. The need for this step is compounded by the fact that French is a highly flexional language where a single verb (at the core of relation phrases) can take many different forms. Verbs in *r* are therefore lemmatized in this study, using an in-house resource.[¶]

[¶]http://rali.iro.umontreal.ca/rali/en/bde-bdf

Original: (*Étienne Gilson*, *en propose, la définition suivante*)	***r*** = proposer
Original: (*Socrate, n'accorde que peu de, importance*)	***r*** = accorder
Original: (*Frédéric Paulhan, être nommé en, 1881*)	***r*** = nommer

FIGURE 4 Examples of relation simplifications for 3 triples pertaining to philosophers

Since we still had at this point quite a lot of variation, we decided to proceed even more aggressively with a (definitely debatable) strategy that consisted in keeping only the last verb for each relation phrase. We reasoned that what interests us at this point is whether a given *arg1* or *arg2* is associated with a given verb. The finer subtleties of meaning that auxiliary verbs and adverbs bring could be ignored. For example, from the triple (*Kant*, *has not been trying to establish*, *this*), we can derive the fact that *Kant* and *establish* are a valid collocation of an argument with a relation. The negative form of the original relation phrase does not change this.

This allows us to do away with adverbs and relation phrases. This simplifies those phrases, at the cost of alterations in meaning. This includes the removal of the reflexive clitic *se* in verb phrases, eg, *se lancer vers—throw oneself at* becomes *lancer—throw*. Note that this is already the case in the original version of REVERB, where reflexive pronouns (*yourself*, *herself*, etc) are excluded from *arg2*s. Our working hypothesis here is that the benefit of having less variations in relations (and therefore, less sparse distributions in relations) outweigh the cost of the semantic distortions brought on by these simplifications. Figure 4 shows examples of these simplifications.

4 | TRIPLE EXTRACTION

4.1 | Corpora

We extracted triples from 2 distinct French corpora, called Wiki and Erudit. The Wiki corpus is a text serialization of all French Wikipedia articles as of June 2014—1.5 million articles, in total.[#] The Erudit corpus is derived from the online collection of scholarly and cultural journals curated by the Érudit Consortium, which consists of 158 journals, mostly in the humanities.[‖] We extracted the raw text from 19 000 XML documents in French, a task made easy by the principled tagging effort performed on these documents by the Érudit team.

There is a stark contrast in topics and style between the two corpora. It is noteworthy that one of the editing guidelines for contributors in Wikipedia is to "make technical articles understandable"** to the widest audience possible. Scientific papers in Erudit do not have this constraint and can be quite complex. Furthermore, the point of view in Wikipedia must be neutral (all significant views must be represented equally without bias), while scientific papers usually revolve around their authors' main point and its corresponding argumentation.

The statistics for these two corpora are shown in Table 1. It is worth noting that, beside their dissimilar domains, the 2 corpora also differ sharply in their nature by their average sentence length, most likely because of the comparative verbosity of scholarly papers that make up the erudit corpus.

[#]http://rali.iro.umontreal.ca/rali/en/wikipedia-dump

[‖]http://erudit.org/revue/

**https://en.wikipedia.org/wiki/Wikipedia:Make_technical_articles_understandable

TABLE 1 Corpus statistics for the Wikipedia corpus and the corpus derived from Erudit

Corpus	Domain	Docs	Sentences	Tokens	Forms	Tokens/Sent
Wiki	Generic	1.5 M	31.1 M	668 M	28.7 M	20
Erudit	Humanities	19 k	2.8 M	96 M	2.8 M	34

TABLE 2 Extraction statistics for corpora Wiki and Erudit

Corpus	Raw triples	Triples w/o pronouns	Relations	arg1s	arg2s
Wiki	30.4 M	20.8 M	1.2 M	7.2 M	7.4 M
Erudit	4.7 M	3.1 M	0.4 M	1.3 M	1.4 M

We only use triples without pronouns in this study, losing about a third of the original triples. The remaining statistics indicate the number of different relations, arg1s and arg2s found in these filtered triplets.

4.2 | Extraction of triples

We performed the extraction of triples (*arg1*, *r*, *arg2*) using the modified version of REVERB described in Section 3. Table 2 shows the extraction statistics.

The triples in both corpora follow a Zipfian distribution. For Wiki, the frequency of a triple can be approximated by $freq = 10453 \times rank^{-0.716}$, with a coefficient of determination $R^2 = 0.997$, although the 3 most frequent triples are overrepresented. They have a frequency of about 30 000 each. The most frequent stems from the extraction (*The evolution in population*, *is known*, *throughout*). We found out that they are the product of Wikipedia templates on demographics. Wikipedia templates produce repetitive text that might be needed on any number of articles or pages.[††]

The most frequently occurring relations in corpus Erudit are *être*, *avoir*, *faire*, and *devenir* (*be, have, do,* and *become*), involved in 17% of all triples extracted. The arguments *arg1* and *arg2* are dominated by pronouns. For both corpora, the 10 most frequent *arg1*s are all pronouns. Since these anaphora render their triplets uninformative, we filtered them using a simple blacklist, losing a third of the raw triples.

5 | CLASSIFICATION OF ENTITY INSTANCES

In this study, we attempt to classify entity instances found in extracted triples (eg, *Michel Foucault*) into entity types (eg, *auteur—author*). Entity instances are not limited to traditional named entities. For example, we would also like to classify an *article* as an *étude* (*scientific study*). An additional challenge stems from the fact that we use the large, generic Wiki corpus to define the entity types and then proceed to classify instances found in triples extracted from the Erudit corpus, which is different in content and style. The process for defining types is illustrated in Figure 5 and explained in the following 2 sections.

5.1 | Selection of entity types

We start by defining a set of entity types in a loosely supervised way. We filter the triples (*arg1*, *r*, *arg2*) from Wiki by keeping only those that satisfy the constraints that *r* must be

[††]https://en.wikipedia.org/wiki/Help:Template

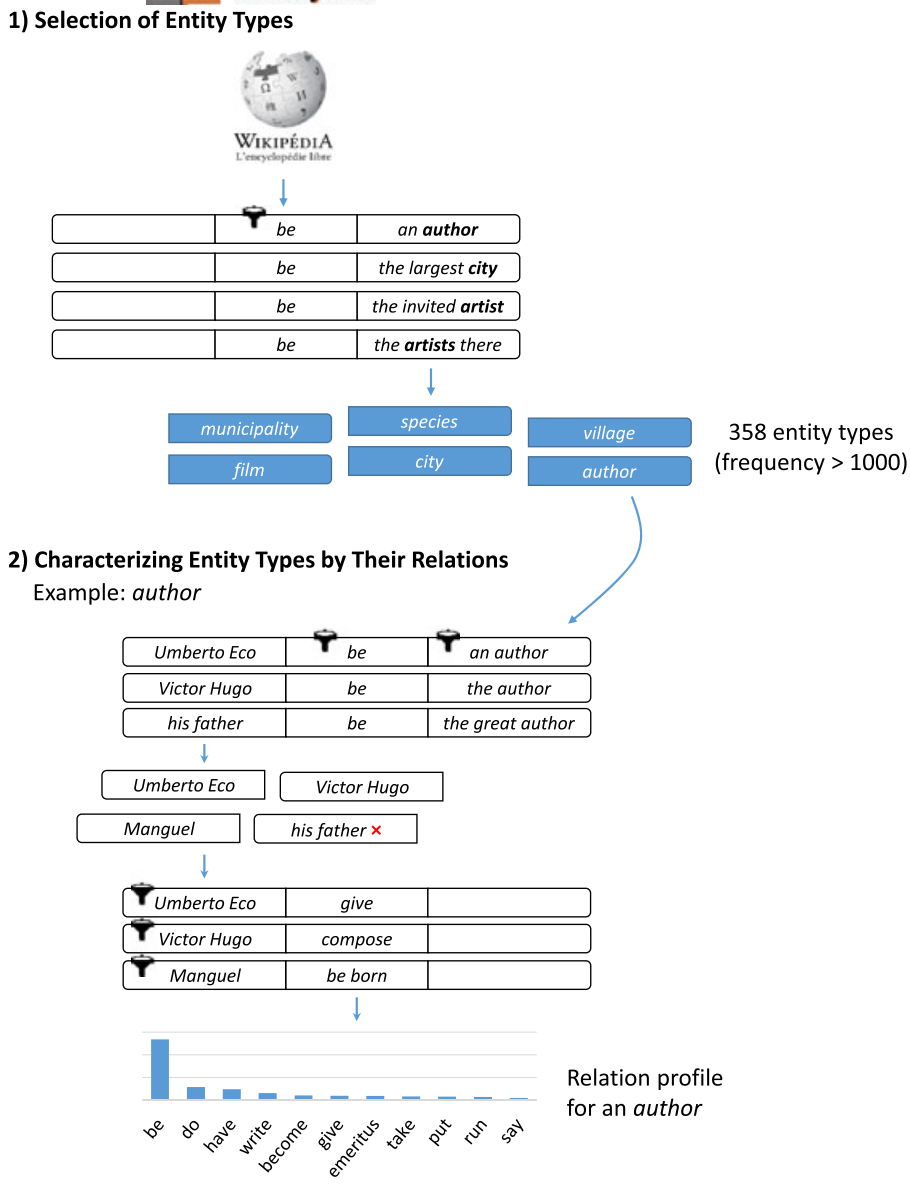

FIGURE 5 The process we used to identify entity types (top of the figure, Section 5.1) and to create their respective relation profiles (bottom of the picture, for the type *author*, Section 5.2) [Color figure can be viewed at wileyonlinelibrary.com]

the verb *être* (*to be*), and *arg2* must contain a common noun *t*. All such elements *t* whose frequency is greater than an empirically devised threshold of 1000 are kept without any further supervision and constitute the set *T* of entity types considered in this study. This step is illustrated in the top part of Figure 5.

The set *T* contains 358 "naturally occurring" types. The 5 most frequent types are *commune* (*municipality*), *espèce* (*species*), *village*, *film*, and *ville* (*city*), presumably reflecting the relatively large number of Wikipedia articles devoted to these topics. Some topics have overlapping meanings, like *commune* and *ville*. Other such overlapping topics comprise *philosophe* (*philosopher*), *auteur* (*author*), and *écrivain* (*writer*). We did not regroup or filter out these overlapping categories. On a

TABLE 3 Instances for 2 types: *auteur* (*author*) and *période* (*period of time*)

Instances for *Author*	Instances for *Period of Time*
Farid Boudjellal	l'estive (*grazing period*)
Jean-Florian Collin	1890 (*1890*)
Richard Matheson	le contexte des noms de domaine (*domain name context*)
Bernard Yaméogo	la première restauration (*First Restoration*)
Hanns Heinz Ewers	le chalcolithique (*Copper Age*)
Thomas Norton	… 152 instances overall
… 416 instances overall	

related note, we consider entity types as nonhierarchical: even when a type t_1 semantically subsumes a type t_2, they are considered completely distinct.

5.2 | Characterizing entity types by their relations

Once the set T of entity types is built, we characterize each $t \in T$ by the relations where t is involved, reasoning that different instances of a given type t must have similar relations and that these relations should differ from those involved with instances of a different type.

We therefore seek to build a relation profile P_t for each t. At its simplest, this profile will include relations (ie, verbs) and their associated count. We find the relations of interest in the triples extracted from the Wiki corpus.

We start by identifying a set of instances for each t. We gather all *arg1*s from triples (*arg1*, *r*, *arg2*) where *r* is the verb *être*, *arg2* contains the common noun *t*, and *arg2* is not a hapax. This works reasonably well, but instances are often contaminated with ubiquitous instances. For example, a third of all 358 topics contain the instance *son père* (*his father*). Manual examination revealed that anaphora is to blame for this phenomenon, since these pervasive instances are associated with multiple types (as in *his father was a physicist* and *his father was a sportsman*). To compensate for this, we removed entity instances appearing in more than 2% of the 358 topics, an upper limit we deemed "reasonable" on the number of types an instance can belong to. There are 237 instances per type on average (min, 1; max, 4953). Table 3 shows a random sample of the instances identified in the Wiki corpus for the types *auteur* (*author*) and *période* (*period of time*). A manual evaluation of half a dozen instance lists reveals a 90% precision. Errors vary from slight inaccuracies in classification (eg, while *Jean-Florian Collin* has written a few books, he is primarily known as an architect and politician, not as an *author*) to flagrant extraction issues (eg, *le contexte des noms de domaine* [*domain name context*] is not a *period of time*).

From the relatively precise list of instances for each type *t*, we are able to inspect triples containing these instances at the *arg1* position and gather all corresponding relations to build a relation profile P_t.

For the entity type *auteur* (*author*), the 10 most frequent relations gathered are *be*, *do*, *have*, *write*, *become*, *give*, *emeritus*, *take*, *put*, *run*, and *say*.[‡‡] A systematic error during part-of-speech tagging gives rise to the erroneous *emeritus*, where the French word *émérite* is mislabeled as a verb rather than as an adjective. For *période* (*period of time*), the most frequent relations are *be*, *mark*, *see*, *follow*, *have*, *do*, *become*, *put*, *av*, and *know*. Here, *av* is also due to an unfortunate tagging error.

We distinctly face a situation where uninformative relations (*be*, *do*, *have*, etc) appear in all profiles. At the same time, more type-specific relations emerge (*write*, *say*, for an *author*; *mark* and *follow* for a *period of time*).

[‡‡]We translate the French relations for the sake of clarity.

TABLE 4 Relation profile excerpt for the entity type *auteur* (*author*)

Author		Period of Time	
Relation	**Frequency**	**Relation**	**Frequency**
run	293	*gothic*	364
diverge	38	*happen*	220
report	197	*mark*	1709
speak	239	*start*	218
author	95	*follow*	1405
write	635	*change*	95
969 relations	$\sum = 33\ 671$	1029 relations	$\sum = 41\ 954$

For each relation, a translation is provided for convenience. Relations are listed in decreasing order of tf–idf score. The relation *gothique* (*gothic*) is due to a POS-tagging error.

To get a better sense of the quality of these relation profiles,[§§] we add a tf–idf score to each relation in a given profile P_t. We computed tf–idf by considering each profile as a document, populated with words that correspond to the relations. In other words, for each relation in a given P_t, tf is the relation count, and idf is proportional to the inverse of the number of profiles containing the relation. Table 4 shows an excerpt of the profile for the entity type *auteur* (*author*).

5.3 | Extracting entity instances from Erudit to create dev and test sets

Our goal is to label entity instances with the correct entity type. While these types were extracted from the corpus Wiki, we select instances from the corpus Erudit, in an effort to assess how well the entity types from one corpus generalize to another one, with a different writing style (see Section 4.1).

We performed triple extraction with the adapted REVERB previously described on Erudit. We then selected all *arg1*s whose frequency is greater than 50. We extracted their relation profiles as explained in Section 5.2.

To create a data set, we manually labeled the 120 instances featuring the most relations in their relation profile. We discarded instances that were ambiguous (eg, the last name *Tremblay* is not enough to identify the entity), were the result of extraction errors (eg, an adjective mislabeled as a noun), or simply did not belong to any types (eg, *snow* and *orientation*). Each instance received a label consisting of the entity types extracted in Section 5.2 to which it belongs. On average, an instance is labeled with 3.89 types. For example, *Michel Foucault* received the labels *philosopher*, *author*, and *writer*.

The classification algorithms described in Section 5.4 were tuned on 20 of these instances while the other 100 constituted the test set. We did not need a large number of development instances, since our classifiers do not require the production of a model. The test set was used to assess the impact of a few parameters and to fine-tune some of them.

5.4 | Classifiers

Our goal is to classify entity instances (eg, *Michel Foucault*, *article*) extracted from Erudit into entity types (eg, *author*, *scientific study*) characterized by the relation profile observed in the corpus Wiki. We tried 4 different approaches to find the closest relation profile for a given instance. Let $P_t = \{r_{t1},$

[§§]We also compute these scores for classification purposes (see Section 5.3).

$r_{t2}, \ldots, r_{tm}\}$ be the set of all m relations for a given type t. Let $P_i = \{r_{i1}, r_{i2}, \ldots, r_{in}\}$ be the set of all n relations for a given instance i. Let sim be a similarity function between 2 profiles. We attribute a type t^* to a given instance i by finding

$$t^* = \underset{t}{\operatorname{argmax}}\ \operatorname{sim}(P_t, P_i).$$

The set of all possible types t includes all the types manually identified during labeling (34 types) to which we added 16 other types (distractors), randomly selected from the set T of all 358 types identified in Section 5.2, for a total of 50 possible types available to each classifier.

For each similarity metric, we added a hyperparameter λ that constrains the number of relations we consider in a profile when computing a similarity. The value λ specifies that only the λ-most frequent relations in P_t and the λ-most frequent relations in P_i should be used. The others are ignored. This allows the algorithm to focus on the most represented relations in each profile. The optimal λ for a given similarity metric is found by exhaustive search on the development data set. We also investigated results without any such filter, ie, where we consider all relations.

jaccard is our baseline and consists in computing the generalized Jaccard similarity coefficient between the 2 relation profiles.

We first derive a frequency vector for each profile. For example, for the type profile P_t, we obtain $\mathbf{v}_t = < freq(r_{t1}), freq(r_{t2}), \ldots freq(r_{tj}), \ldots >$, where $freq(r_{tj})$ is the count of the relation r_j, for the profile t. We do the same for a given instance i and obtain $\mathbf{v}_i$. The similarity can then be computed using the following formula:

$$\text{jaccard}(\boldsymbol{v_t}, \boldsymbol{v_i}) = \frac{\sum_j \min(\boldsymbol{v_t}[j], \boldsymbol{v_i}[j])}{\sum_j \max(\boldsymbol{v_t}[j], \boldsymbol{v_i}[j])}.$$

inter is another simple similarity metric for which $\operatorname{sim}(P_t, P_i) = |P_t \cap P_i|$, where the sets are obviously influenced by λ.

cos is the cosine similarity between 2 profiles and is computed using the frequency vectors $\mathbf{v}_t$ and $\mathbf{v}_i$ discussed above. We also tried a variant **cos-bin** where, instead of the frequencies of the relations, the vectors are encoded with 1 (the relation is present in profile) or 0 (the relation is absent). If **cos** and **cos-bin** give similar results, this usually indicates that the frequency scores carry little information.

tfidf makes use of the tf–idf scores we introduced in Section 5.2. The similarity function here is comparable to the information retrieval scenario where the relations in P_i constitute the query, and the different P_t are each a document in a collection. The "most relevant" P_t is therefore the most similar. The similarity score for a given P_i is the sum of the tf–idf scores for each relation found in P_t.

Finally, **kl** is a comparison of the relation distributions in P_t and P_i using the Kullback–Leibler divergence $D_{\mathrm{KL}}(Q_t \,\|\, Q_i)$, where Q_t and Q_i are the probability distributions (relative frequencies) of relations in P_t and P_i, respectively. KL divergence does not handle 0s in these distributions, so we smooth by replacing them with a small value $\varepsilon = 0.1$ whose value was found when tuning on the development set.

Formally, given λ and ε, for a type profile P_t and an instance profile P_i, we truncate the corresponding relation vectors $\mathbf{v}_t$ and $\mathbf{v}_i$ presented above to get λ-sized vectors

$$\boldsymbol{c}_t = \langle freq(r_{t1}), freq(r_{t2}), \ldots freq(r_{t\lambda}) \rangle,$$

$$\boldsymbol{c}_i = \langle freq^*(r_{i1}), freq^*(r_{i2}), \ldots freq^*(r_{i\lambda}) \rangle,$$

where

$$freq^*(r) = \begin{cases} r, & r \neq 0 \\ \varepsilon, & \text{otherwise} \end{cases}.$$

We then have the corresponding distributions

$$Q_t = \boldsymbol{c}_t / \|\boldsymbol{c}_t\|_1 \text{ and } Q_i = \boldsymbol{c}_i / \|\boldsymbol{c}_i\|_1.$$

We compute the Kullback–Leibler divergence between Q_t and Q_i by summing over the relations r present in Q_t, like so

$$D_{KL}(Q_t \| Q_i) = \sum_r Q_t(r) \times \log \frac{Q_t(r)}{Q_i(r)}.$$

When $Q_t(r)$ is zero, the rth term of the sum is also zero. The similarity function is the inverse of the divergence, ie, $\text{sim}(P_t, P_i) = -D_{\text{KL}}(Q_t \,\|\, Q_i)$.

We also experimented by throwing out the dimensions with 0 values, with disappointing results (not presented here). We also experimented with the symmetrized divergence $D_{\text{KL}}(Q_t \,\|\, Q_i) + D_{\text{KL}}(Q_i \,\|\, Q_t)$, which is standard practice, without any improvement over the results we present below.

All in all, we tested 85 different classifiers (different λ values make for distinct classifiers within the same family of algorithms).

5.5 | Results

The classification errors for all similarity metrics are presented in Table 5. An instance is considered misclassified if the closest type t^* returned by a given similarity metric is not part of the labels attributed manually (1-best evaluation). The results on the test set are unsatisfactory for all metrics, except the Kullback–Leibler divergence, with a classification error rate of 37% on the test set, lower than random (92%). With an average of 3.9 correct types per instances, and 50 types to choose from, the random error rate is $1 - (3.9/50) = 92.2\%$. The KL divergence also outperforms the Jaccard baseline (55%). The **kl** classifier also generalizes reasonably well from dev set to test set, with relatively little degradation in performance.

When we consider the 2 best types output by **kl** ($\lambda = 100$), the error rate decreases slightly, at 35%. When using the 3 closest types, it becomes 28%. This shows that the errors produced by this algorithm are not due to an unfortunate choice between a few *ex aequo* or near-*ex aequo* candidates.

Since we had quite a few (weak) classifiers, it was a natural extension to try to make them vote to create simple but potentially interesting metaclassifiers. The algorithms **vote-maj-*n*** ($1 < n < 5$) classify a given instance by taking the majority vote of the 85 classifiers described above. n represents the size n of the n-best candidate list a given classifier votes for.

Another simple way to create the voting committee is to choose among the voting algorithms those that perform best on the dev set. The metaclassifiers **vote-maj-top-*p*** ($1\% < p < 100\%$) form the voting committee by selecting the classifiers among the p best performing algorithms. The committee is formed on the development set and used as is on the test set.

Finally, we manually created a voting committee **vote-maj-top-besttune** consisting of the 6 algorithms that performed best on the development set, ie, **jaccard** ($\lambda = 50$), **inter** ($\lambda = 50$), **cos** ($\lambda = 100$), **cos-bin** ($\lambda = 100$), **tfidf** ($\lambda = 500$), and **kl** ($\lambda = 100$).

The results for these voting algorithms appear in the last 3 lines of Table 5. Unfortunately, they do not yield better results than **kl** ($\lambda = 100$). For **vote-maj-top-*p*** and **vote-maj-top-besttune**, a superior performance on the development set is obviously observed, since we select the best classifiers on this very data set.

TABLE 5 Classification results for 6 similarity metrics on the development set and the test set

	Classification Error, %	
Similarity metric	**Dev** ($n = 20$)	**Test** ($n = 100$)
random	92	92
most frequent (*city*)	90	91
jaccard ($\lambda = 50$)	55	57
inter ($\lambda = 50$)	55	67
cos ($\lambda = 100$)	50	64
cos-bin ($\lambda = 100$)	50	77
tfidf ($\lambda = 500$)	35	52
kl ($\lambda = 100$)	**30**	**37**
vote-maj-n ($n = 2$)	40	38
vote-maj-top-p ($p = 10\%$)	25	44
vot-maj-top-besttune	20	40

The **kl** algorithm (Kullback–Leibler divergence) yields the best results among single algorithms. The ensemble classifier **vote-maj-top-*p*** does not improve significantly upon this. The **random** classifier picks a solution at random while **most frequent** always picks the most frequent label (*city*). The latter two are provided for comparison.

5.6 | Error description

We manually inspected the results for **kl** to describe the kinds of errors the similarity function led to. We identify 2 kinds of errors: "hard" and "soft" errors (for lack of better terms). Hard errors occur when an entity is unambiguously mislabeled, eg, *Socrates* is a *municipality*. Soft errors arise when the predicted type is not entirely incompatible with the instance to label, eg, *United States* is a *group*. This is admittedly a subjective exercise, but readers can judge for themselves by examining a sample of this partition in Table 6. Overall, of 37 errors, we found that 22 were hard and 15 were soft. Had we tolerated the latter, the classification error for **kl** on the test set would have fallen to 22%. These 22 hard errors contain 11 misclassifications where a country (eg, instances *Belgium*, *Germany*, and *Canada*) is classified as a *city*.

The parameter λ significantly affects the performance of the kl similarity metric. There seems to be an optimal value in the range $\lambda \geq 75$, at around 40% classification error, outside of which the performance is poor. Both the development and test sets exhibit this behavior.

Further manual examination of the "hard" classification errors fails to produce a clear, general picture of what kind of instances or types do not lend themselves to accurate classification, for a given algorithm or algorithm family. On the contrary, it seems that each error has its own specific explanation.

An interesting example of such a phenomenon concerns the instance *autochtones—aboriginal people* in the development set. While the reference states that the entity type should be *people* or *group*, none of the 85 algorithm variants presented earlier manage to label the instance correctly. The closest any of them come is when some of them produce the type *characters*. We closely examined both instance and type and discovered that the relational profile for the instance *aboriginal people* (extracted from the scholarly erudit) is strikingly different from Wikipedia's *people* type. While the latter most frequently involves *people* with warfare-related verbs like *take*, *occupy*, *lose*, *conquer*, and *attack*, Erudit associates most frequently *aboriginal people* with less aggressive verbs, like *live*,

TABLE 6 A sample of a few instances, their manual labels, and the predicted type by the kl algorithm

Instance	Type Labels	Predicted Type (kl)
1960	*year, period of time*	*period of time*
Aragon	*artist, author, writer*	*author*
article	*scientific study, book, compilation*	*scientific study*
Belgium	*country, place, places, toponym*	*city*[a]
aboriginal people	*people, group*	*characters*[b]
Germany	*country, place, places, toponym*	*city*[a]
Health Canada	*organisation, association, organism, group*	*physician*[b]
Michel Foucault	*philosopher, writer, author*	*author*
prime minister	*minister, president, master*	*minister*
Socrates	*philosopher, writer, author*	*city*[a]
text	*scientific study, book, compilation*	*book*

[a]"Hard" classification errors.

[b]"Soft" classification errors.

represent, *constitute*, *present*, *allow*, etc. Intuitively then, the instance is actually closer in meaning to a more generic "human" type like *characters*. To us, this specific error is therefore attributable to the dissimilarity between the corpora Wiki and Erudit. In other terms, this is a problem of domain adaptation.

Another such "hard" problem for the classifiers is the instance *cinéaste—film maker*, almost never correctly classified as an *artist*. Once again, there seems to be a contrast between what Wikipedia considers an artist (with the most frequent relations including *interpret*, *sing*, *play*, *write*, *produce*, *release*, *record*, *give*, *present*, and *show*) and what a *film maker* actually does in Erudit (with a profile where *all* of the frequent relations mentioned above are absent). It is clear that Wikipedia engenders an all-encompassing portrait of an *artist*, with relations hinting at the fact that they may be actors, singers, novelists, record artists, etc. In contrast, a cursory examination of the scientific articles in Érudit that mention film makers show that they focus on the careers and oeuvres of a few persons, but never adopt a generic point of view where they would explain, say, what a film maker does or why he is considered an artist. The problem is compounded by the fact that the relation profile is very sparse for this instance, with only about 100 relations in total.

6 | ERROR ANALYSIS

6.1 | Type relation frequency and classification error

Since the frequency of a given entity type directly influences the number of relations found for this type and therefore the estimated distribution of its relations, we investigated whether there is a correlation between the frequency of an entity type and the classification errors for this type. Intuitively, less frequent types, with less relations to propose to an algorithm for the purpose of classification, must be more difficult to accurately predict.

To test this hypothesis, for a given algorithm, we examined the correlation between $sum\{\mathbf{v}_t\}$, the cumulative frequency of relations for a given type t, and precision $prec_t$, the percentage of instances classified as type t where this classification is correct. We performed this evaluation on the test data

set. We did this for each algorithm separately, but we lumped together the different variants of a single algorithm produced by variations in λ. In other words, for **kl**, we have data points for kl (λ = 5), kl (λ = 10), etc. This allows us to get more data points for a more significant statistical analysis. Naturally, the value of λ directly influences $sum\{\mathbf{v}_t\}$, since it truncates $\mathbf{v}_t$.

However, for all algorithms described in Section 5.4, we did not find any correlation between $sum\{\mathbf{v}_t\}$ and $prec_t$, with all coefficients of determination R^2 inferior to 0.1. Figure 6 shows how this cumulative frequency is not related to the classification precision: Types with few relations (left-hand side of the graph) are produced with a precision comparable to those with a large number of relation (right-hand side of the graph). For example, the most frequent entity type *ville—city* (frequency = 144 000) is only produced with 40% precision, while *rédacteur—writer* (frequency = 196) is produced with perfect precision.

We performed the same analysis with recall and F-measure and found the same absence of correlation, for all algorithms presented. This seems to indicate that the nature of the relations found influences the classification error much more than their mere frequency. (This is why algorithms like **cos-bin** and **tfidf** were selected in the first place to perform classification, yet they do not seem to fully compensate for this.)

6.2 | Instance relation frequency and classification error

The intuition explained in the previous section also holds for instances: We can test whether instances that are sparsely populated are more prone to being incorrectly classified by a given algorithm and vice versa.

We proceed differently from entity types this time, since it is not possible to compute a precision rate by instance like we did by type in the previous section. This time, we create groups of instances i that share similar $sum\{\mathbf{v}_i\}$, and we check whether groups with low $sum\{\mathbf{v}_i\}$ fare better or worse than groups with high $sum\{\mathbf{v}_i\}$. We performed this on the test data set (100 instances). Visually, this is represented in Figure 7, for all algorithm families presented. It is clear from this graph that there is no discernible relation between $sum\{\mathbf{v}_i\}$ and the error rate when classifying. In other words, for a given

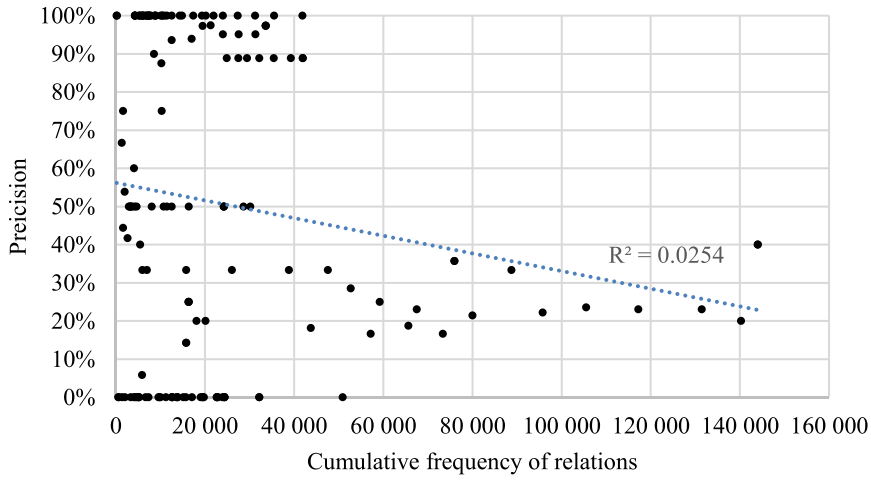

FIGURE 6 $prec_t$ vs the cumulative frequency $sum\{\mathbf{v}_t\}$ of relations for different types, for the algorithm kl, on the test set. The graph shows that the frequency of relations for a given type t does not influence the precision with which kl labels instances belonging to the type t [Color figure can be viewed at wileyonlinelibrary.com]

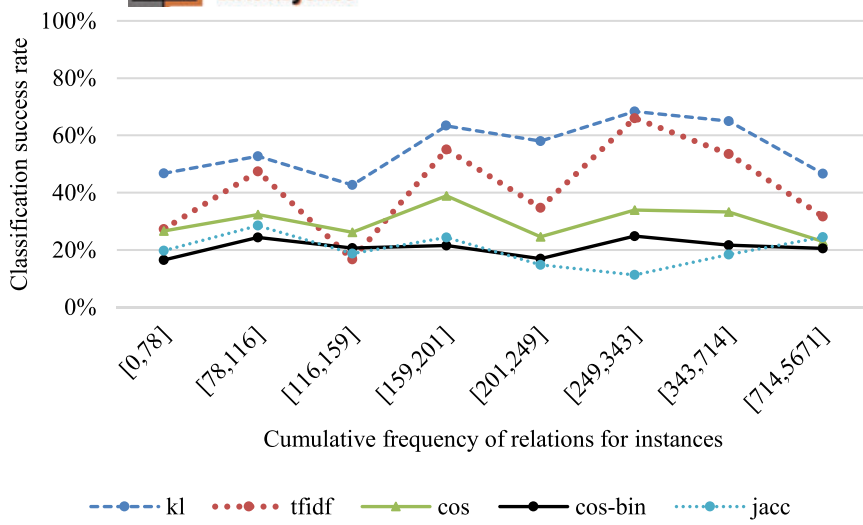

FIGURE 7 Classification success rate (1 – error rate) for different groups of instances sharing a bracket of $sum\{\mathbf{v}_i\}$. Each bracket aggregates 200 instances from the test data set [Color figure can be viewed at wileyonlinelibrary.com]

algorithm, classification errors cannot be clearly attributable to sparse instances. The graph further shows how **kl** clearly dominates the other algorithms for all the brackets proposed, although **tfidf** comes a close second for 3 brackets.

Had we had, say, an algorithm that had performed better for low $sum\{\mathbf{v}_i\}$ and another that conversely had performed better on instances with high $sum\{\mathbf{v}_i\}$, we could have devised a hybrid meta-algorithm that would have used 1 algorithm or the other depending on the instance being classified. As it stands, though, **kl** is the clear winner here, irrespective of $sum\{\mathbf{v}_i\}$.

6.3 | Relation frequency as a selection criterion

Up to now, the frequency of a relation in a given profile is the criterion used when truncating them using the hyperparameter λ. We select the λ-most frequent relations when we truncate a profile. The intuition being that the less frequent relations may be too noisy to characterize a type or an instance. However, the previous 2 sections have shown that this frequency may be less informative that initially thought, because they do not seem to influence the classification performance in a clear way. This may be due in part to the fact that the most frequent relations are also the most common among all profiles. In Section 4.2, we observe that the most frequent relations are *be*, *have*, *do*, and *become*, and this is something that is also present in the profiles.

A more discriminant criterion for a relation is its tf–idf score. For the type *philosopher*, for instance, the most frequent relations are *be*, *do*, *have*, *write*, and *say*, while the relations with the best tf–idf scores are *refute*, *analyze*, *ask*, *deny*, and *criticize*. We therefore reran the classifiers presented in Section 5.4 with a λ that now keeps the relations with the λ best tf–idf scores. Unfortunately, none of the algorithms performed better than what is presented in Table 5.

7 | A MANUAL EVALUATION

At 37% of classification error rate, our best algorithm still misclassifies a little over 1 instance of 3, which is significant. When faced with such an apparently difficult problem, it is useful to measure

how a human fares on the task, or at least on a similar task. This gives a sense of the difficulty of the task and contributes to setting a possible upper bound on the classification performance.

To do this, a collaborator was asked to classify the instances in our test set. He only had access to the list of 50 entity types and, for each instance, to their relation profile P_i. He did *not* have access to the relation profiles P_t of the 50 types, only to their labels. Our goal was to measure whether consulting a given P_i is sufficiently informative to assign it a type and *not* whether a human can mentally compare a P_t and a P_i. It turns out that the human annotator found the task almost impossible to complete, the instance profiles being too little informative (at least for a human). It does seem that this classification task is quite hard for a human.

Nonetheless, we resorted to a simplified task to obtain at least a human assessment for the task at hand. Instead of having to choose among 50 entity types for a given instance, the human judge was presented with a shuffled list of 3 types: 2 distractor types randomly selected from the types not present in the reference labels and the actual label. His performance was still a disappointing 48% error rate. However, when we asked our best algorithm (**kl** with $\lambda = 100$) to pick a solution from the same set, its error rate was very similar, at 51%. Randomly picking a solution would have given a 66% error rate.

This allows us to draw 2 conclusions. First, **kl** seems to work equally unsatisfactorily as a human for this particular task. This indicates that the task is quite difficult. Second, there seems to be little place for improvement for the algorithm, with a gap of only 3% between human and machine. The fact that the performance improves to 37% when using the full 50 types for **kl** (as shown in Table 5) is due to the fact that the algorithm can pick among many more types, many of which can be one of the correct labels.

The human judge also made an interesting remark: The simplifications we made to the relations (described in Section 3.1) may have been too aggressive when it comes to the passive voice. Since we remove the auxiliary verb *être—be* from verb phrases like *être sorti—have exited* to obtain the simpler *sortir—exit*, we also convert passive voices to active ones. For instance, *be demonstrated* becomes *demonstrate*. This latter case exemplifies the misleading nature of the simplification step. While *be demonstrated* can be applied to a *study*, for example, it is not applicable to a *philosopher*. In hindsight, this is something that could very well have influenced the results negatively and should be rectified in the pipeline.

The annotator made 2 additional observations. First, he felt that the frequency of relations was not a good indicator of the kind of type an instance belonged to. Often, he would have to consult the least frequent relations to perform the classification. This is consistent with what we have said so far: the most frequent relations are often the most ubiquitous, and the less discriminant.

Second, he indicated that the absence of the *arg2* rendered the task extremely difficult. Clearly (*x*, *give*, *lesson*) and (*x*, *give*, *birth*) suggest 2 different types for *x*. This is a remark that comes as no surprise: the richer the context of a relation, the clearer its meaning and the entity type involved. In this study, we initially thought the relation profile sufficient to unambiguously describe a type or an instance (and we succeeded up to a point), but exploring the potential for discrimination of *arg2*s is a natural extension to this work. It should be pointed out, however, that the more context is added to a relation, the less frequent this cooccurrence is bound to be, resulting in profiles rich in low-frequency elements. How this would affect the classifier presented remains to be seen.

8 | DISCUSSION

One of the goals of this paper was to attempt OIE in French and assess the difficulties encountered while doing so. The adaptation of REVERB went smoothly, partly because there are drop-in French

replacements for the POS and chunking statistical models the software uses and partly because its API is expertly written. We also opted not to adapt all of REVERB's filters to French, because we favoured recall over precision during the extraction step, since the following steps engendered a lot of filtering. However, we feel that implementing the rest would be straightforward should we need it in the future.

Like most OIE approaches, the problem of uninformative and ambiguous triples is significant. We lose a third of the extracted triples to pronominal anaphora alone, which amounts to 10 M triples for the corpus Wiki. This highlights the need for a robust anaphora resolver. For French, a recent study on a commercial-grade grammar checker[21] shows that 70% of these anaphora could be resolved, a possible addition of 7 M triples of information in our case. Naturally, the figure of 10 M triples lost is a minimum, since it does not take into account other types of anaphora (eg, *his father*). However, our system behaved reasonably well in the face of these latter problems, thanks to simple frequency thresholds akin to idf (inverse document frequency), reasoning that ubiquitous instances are bound to be nonspecific and uninformative.

The second goal of this paper was to explore whether it was possible to extract information from a generic corpus (Wiki) and use it to infer new knowledge in a different, domain-specific corpus (Erudit) through the analysis of OIE's resulting triples. We showed that it is indeed possible to identify and characterize entity types by the relations their respective instances are associated with. It then becomes possible to put these profiles to good use and classify instances extracted from the other corpus, for two-thirds of these instances. To our knowledge, in this context, this approach is original. It does suffer however from the fact that the instances to classify must be relatively frequent (to gather enough information on them). The system described here would be hard-pressed to associate a hapax instance to an entity type, for instance. Moreover, establishing "relation profiles" proves sensitive to systematic extraction errors, notably those committed during part-of-speech tagging. A tagging error that mislabels an adjective for a verb in a specific context (like *gothic* preceded by *author*) is bound to create significant artefacts in relation profiles, since the latter are designed to gather just such systematic specificities, whether they are linguistically motivated or the result of an extraction problem.

There is room for improvement when considering the figure of 37% of classification error reported here for the algorithm based on the Kullback–Leibler divergence. We identify some possible solutions above. Extensive error analysis, including a manual evaluation, reveals the difficulty of the task and uncovers some aspects of our work that could be improved.

It does seem that the task is quite complex, because in part of the extraction complications outlined above. However, a more fundamental problem lies in the insidious semantic divergences between Wikipedia and Érudit: The former tends to describe types with relations that can differ significantly from those of corresponding instances extracted from the latter, because both corpora are so different in scope, style, and purpose. Wikipedia is generic and has a large coverage, while Érudit makes use of instances in a very scholarly and narrow perspective, focusing on the specific aspects of those instances that are of interest to the scientific community. In other words, *domain adaptation* is a challenging aspect of the task we proposed here. This problem also reveals itself during the manual evaluation we propose, where the human annotator fails to produce the correct label 50% of the time when looking at the instance relations.

Another question is the ambiguous place of relation frequencies in profiles. On the one hand, the most frequent relations are often the least discriminant because ubiquitous, but on the other hand, the tfidf scores and tfidf algorithms designed to compensate for that fail to produce convincing results. To further complicate the matter, the error analysis we presented shows that there is no discernible correlation between the sparseness of a profile's relations and the difficulty in using it when classifying. What is certain, however, is that the relation *distributions* these frequencies shape are useful when

comparing types and instances, because a KL divergence comparing these distributions is the best classifier we devised here.

The error analyses also hint at more technical problems in our implementations. We may well have been too aggressive when simplifying relation phrases, stripping them of elements essential to their meaning, eg, the verb auxiliary indicating a passive voice. The relations by themselves may also be simply insufficient for the task at hand and could be enriched with their *arg2*s, although doing so would reduce their frequencies and render their probability distributions sparser.

Whatever the difficulties encountered, we hope to have shown in this work that there is definite potential in the idea of exploiting the knowledge derived by OIE from a generic corpus and then applying it to a stylistically and thematically different collection of texts.

ACKNOWLEDGMENT

The research for this paper was financially supported by the Nuance Foundation. We also wish to thank the reviewers for their helpful comments.

REFERENCES

1. Sowa JF. The challenge of knowledge soup. In: *epiSTEME-1 Conference*, December 16, 2014; Goa, India, pp. 55-90.
2. Banko M, Cafarella MJ, Soderland S, Broadhead M, Etzioni O. Open information extraction from the web. In: *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, January 6–12, 2007; Hyderadad, India, pp. 2670-2676.
3. Ferrucci DA. IBM's Watson/DeepQA. In: *Proceedings of the 38th Annual International Symposium on Computer Architecture*, ACM, 2011.
4. Poon H, Quirk C, DeZiel C, Heckerman D. Literome: PubMed-scale genomic knowledge base in the cloud. *Bioinformatics*. 2014;30:2840-2842.
5. Bollacker K, Evans C, Paritosh P, Sturge T, Taylor J. Freebase: a collaboratively created graph database for structuring human knowledge. In: *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*; June 9–12, 2008; Vancouver, Canada, pp. 1247-1250.
6. Suchanek FM, Kasneci G, Weikum G. Yago: a core of semantic knowledge. In: *WWW 2007 Proceedings of the 16th International World Wide Web Conference*; May 8–12, 2007; Banff, AB, Canada, pp. 697-706.
7. Lehmann J, Isele R, Jakob M, et al. DBpedia—a large-scale, multilingual knowledge base extracted from Wikipedia. *Semant Web J*. 2015;6:167-195.
8. Bizer C, Heath T, Berners-Lee T. Linked data—the story so far. *Int J Semant Web Inform Syst*. 2009;5:1-22.
9. Carlson A, Betteridge J, Kisiel B, Settles B Jr, Hruschka ER, Mitchell TM. Toward an architecture for never-ending language learning. In: *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010)*; July 11–15, 2010; Atlanta, Georgia, pp. 1306-1313.
10. Navigli R, Ponzetto SP. BabelNet: the automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artif Intell*. 2012;193:217-250.
11. Liu H, Singh P. ConceptNet—a practical common sense reasoning tool-kit. *BT Tech J*. 2004;22:211-226.
12. Fader A, Soderland S, Etzioni O. Identifying relations for open information extraction. In: *Empirical Methods in Natural Language Processing*; July 27–31, 2011; Edinburgh, United Kingdom, pp. 1535-1545.
13. Wu F, Weld DS. Open information extraction using Wikipedia. In: *48th Annual Meeting of the Association for Computational Linguistics*; July 11–16, 2010; Uppsala, Sweden, pp. 118-127.

14. Mausam, Schmitz M, Bart R, Soderland S, Etzioni O. Open language learning for information extraction. In: *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*; July 12–14, 2012; Jeju Island, Korea, pp. 523-534.

15. Zhila A, Gelbukh A. Comparison of open information extraction for English and Spanish. *Comput Ling Intell Tech*. 2013;12:714-722.

16. Gamallo P, Garcia M. Multilingual open information extraction. In: Pereira F, Machado P, Costa E, Cardoso A, eds. *Progress in Artificial Intelligence. Lecture Notes in Computer Science*, vol. 9273. Cham, Switzerland: Springer; 2015:711-722.

17. Falke T, Stanovsky G, Gurevych I, Dagan I. Porting an open information extraction system from English to German. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*; November 1–5, 2016, Austin, TX, pp. 892-898.

18. Stanovsky G, Ficler J, Dagan I, Goldberg Y. Getting more out of syntax with PropS. arXiv preprint 2016; arXiv:1603.01648.

19. Downey D, Etzioni O, Soderland S. A probabilistic model of redundancy in information extraction. DTIC Document; *Proceedings of the 19th international joint conference on Artificial intelligence*; July 30–August 05, 2005, 2006; Edinburgh, Scotland, pp. 1034-1041.

20. Hernandez N, Boudin F. Construction automatique d'un large corpus libre annoté morpho-syntaxiquement en français [in French]. In: *Traitement Automatique des Langues Naturelles (TALN)*; Jun 2013; Sables d'Olonne, France. https://hal.archives-ouvertes.fr/hal-00816350.

21. Pironneau M, Brunelle É, Charest S. Pronoun anaphora resolution for automatic correction of grammatical errors (correction automatique par résolution d'anaphores pronominales) [in French]. In: *Proceedings of TALN 2014 (Volume 1: Long Papers). 1*; July 2014; Marseille, France: Association pour le Traitement Automatique des Langues, pp. 113–124. http://www.aclweb.org/anthology/F14-1011.

How to cite this article: Gotti F, Langlais P. From French Wikipedia to Erudit: A test case for cross-domain open information extraction. *Computational Intelligence*. 2017;1–20. https://doi.org/10.1111/coin.12120