ReNoun: Fact Extraction for Nominal Attributes

Mohamed Yahya[*]
Max Planck Institute for Informatics
myahya@mpi-inf.mpg.de

Steven Euijong Whang, Rahul Gupta, Alon Halevy
Google Research
{swhang,grahul,halevy}@google.com

Abstract

Search engines are increasingly relying on large knowledge bases of facts to provide direct answers to users' queries. However, the construction of these knowledge bases is largely manual and does not scale to the long and heavy tail of facts. Open information extraction tries to address this challenge, but typically assumes that facts are expressed with verb phrases, and therefore has had difficulty extracting facts for noun-based relations.

We describe ReNoun, an open information extraction system that complements previous efforts by focusing on nominal attributes and on the long tail. ReNoun's approach is based on leveraging a large ontology of noun attributes mined from a text corpus and from user queries. ReNoun creates a seed set of training data by using specialized patterns and requiring that the facts mention an attribute in the ontology. ReNoun then generalizes from this seed set to produce a much larger set of extractions that are then scored. We describe experiments that show that we extract facts with high precision and for attributes that cannot be extracted with verb-based techniques.

1 Introduction

One of the major themes driving the current evolution of search engines is to make the search experience more efficient and mobile friendly for users by providing them concrete answers to queries. These answers, that apply to queries about entities that the search engine knows about (e.g., famous individuals, organizations or locations) complement the links that the search engine typically returns (Sawant and Chakrabati, 2013; Singhal, 2012; Yahya et al., 2012). To support such answers, the search engine maintains a knowledge base that describes various attributes of an entity (e.g., `(Nicolas Sarkozy, wife, Carla Bruni)`). Upon receiving a query, the search engine tries to recognize whether the answer is in its knowledge base.

For the most part, the aforementioned knowledge bases are constructed using manual techniques and carefully supervised information extraction algorithms. As a result, they obtain high coverage on head attributes, but low coverage on tail ones, such as those shown in Table 1. For example, they may have the answer for the query *"Sarkozy's wife"*, but not for *"Hollande's ex-girlfriend"* or *"Google's philanthropic arm"*. In addition to broadening the scope of query answering, extending the coverage of the knowledge base to long tail attributes can also facilitate providing *Web answers* to the user. Specifically, the search engine can use lower-confidence facts to *corroborate* an answer that appears in text in one of the top Web results and highlight them to the user.

This paper describes *ReNoun*, an open-information extraction system that focuses on extracting facts for long tail attributes. The observation underlying our approach is that attributes from the long tail are typically expressed as *nouns*, whereas most previous work on open-information extraction (e.g., Mausam et al. (2012)) extend techniques for extracting attributes expressed in *verb* form. Hence, the main contribution of our work is to develop an extraction system that complements previous efforts, focuses on nominal attributes and is effective for the long tail. To that end, ReNoun begins with a large but imperfect ontology of nominal attributes that is extracted from text and the query stream (Gupta et al., 2014). ReNoun proceeds by using a small set of high-precision extractors that exploit the nominal na-

[*] Work done during an internship at Google Research.

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 325–335,
October 25-29, 2014, Doha, Qatar. ©2014 Association for Computational Linguistics

Attribute	Fact	Phrase	Verb form seen
`legal affairs correspondent`	`(NPR, legal affairs correspondent, Nina Totenberg)`	NPR welcomed Nina Totenberg as its new legal affairs correspondent.	✗
`economist`	`(Princeton, economist, Paul Krugman)`	Princeton economist Paul Krugman was awarded the Nobel prize in 2008.	✗
`ex-boyfriend`	`(Trierweiler, ex-boyfriend, Hollande`	Trierweiler did not have any children with her ex-boyfriend Hollande.	✓
`staff writer`	`(The New Yorker, staff writer, Adam Gopnik)`	Adam Gopnik is one of The New Yorker's best staff writers.	✓

Table 1: Examples of noun phrases as attributes, none which are part of a verb phrase. Additionally, the first two attributes do not occur within a verb phrase in a large corpus (see § 2 for details) in a setting where they can be associated with a triple.

ture of the attributes to obtain a training set, and then generalizes from the training set via distant supervision to find a much larger set of extraction patterns. Finally, ReNoun scores extracted facts by considering how frequently their patterns extract triples and the *coherence* of these patterns, i.e., whether they extract triples for semantically similar attributes. Our experiments demonstrate that ReNoun extracts a large body of high precision facts, and that these facts are not extracted with techniques based on verb phrases.

2 Preliminaries

The goal of ReNoun is to extract triples of the form `(S,A,O)`, where `S` is *subject*, `A` is the *attribute*, and `O` is the *object*. In our setting, the attribute is always a noun phrase. We refer to the subject and object as the *arguments* of the attribute.

ReNoun takes as input a set of attributes, which can be collected using the methods described in Gupta el al. (2014), Lee et al. (2012), and Pasca and van Durme (2007). In this work, we use Biperpedia (Gupta et al., 2014), which is an ontology of nominal attributes automatically extracted from Web text and user queries. For every attribute, Biperpedia supplies the Freebase (Bollacker et al., 2008) domain type (e.g., whether the attribute applies to people, organizations or hotels). Since the attributes themselves are the result of an extraction algorithm, they may include false positives (i.e., attributes that do not make sense).

The focus of ReNoun is on attributes whose values are concrete objects (e.g., `wife`, `protege`, `chief-economist`). Other classes of attributes that we do not consider in this work are (1) numeric (e.g., `population`, `GDP`) that are better extracted from Web tables (Cafarella et al., 2008), and (2) vague (e.g., `culture`, `economy`) whose value is a narrative that would not fit the current mode of query answering on search engines.

We make the distinction between the *fat head* and *long tail* of attributes. To define these two sets, we ordered the attributes in decreasing order of the number of occurrences in the corpus [1]. We defined the fat head to be the attributes until the point N in the ordering such that the sum of the total number of occurrences of attributes before N equaled the number of total occurrences of the attributes after N. In our news corpus, the fat head included 218 attributes (i.e., $N = 218$) and the long tail included 60K attributes. Table 2 shows examples from both.

Fat head	daughter, headquarters president, spokesperson,
Long tail	chief economist, defender, philanthropic arm, protege

Table 2: Examples of fat head and long tail attributes.

The output of ReNoun is a set of *facts*, where each fact could be generated by multiple extractions. We store the provenance of each extraction and the number of times each fact was extracted.

Noun versus verb attributes

ReNoun's goal is to extract facts for attributes expressed as noun phrases. A natural question is whether we can exploit prior work on open information extraction, which focused on extracting relations expressed as verbs. For example, if we can extract facts for the attribute `advised` or `is advisor of`, we can populate the noun attribute `advisor` with the same facts. In Section 7.2 we demonstrate that this approach is limited for several reasons.

First, attributes in knowledge bases are typically expressed as noun phrases. Table 3 shows that

[1] The occurrences were weighted by the number of semantic classes they occur with in the ontology because many classes overlap.

Knowledge Base	%Nouns	%Verbs
Freebase	97	3
DBpedia	96	4

Table 3: Percentage of attributes expressed as nouns phrases among the 100 attributes with the most facts.

the vast majority of the attributes in both Freebase and DBpedia (Auer et al., 2007) are expressed as nouns even for the fat head (and even more so for the long tail). Hence, if we extract the verb form of attributes we would need to translate them into noun form, which would require us to solve the paraphrasing problem and introduce more sources of error (Madnani and Dorr, 2010). Second, as we dig deeper into the long tail, attributes tend to be expressed in text more in noun form rather than verb form. One of the reasons is that the attribute names tend to get longer and therefore unnatural to express as verbs (e.g. *chief privacy officer*, *automotive division*). Finally, there is often a subtle difference in meaning between verb forms and noun forms of attributes. For example, it is common to see the phrase *"Obama advised Merkel on saving the Euro,"* but that would not necessarily mean we want to say that *Obama* is an advisor of *Angela Merkel*, in the common sense of `advisor`.

Processed document corpus

ReNoun extracts facts from a large corpus of 400M news articles. We exploit rich synactic and linguistic cues, by processing these documents with a natural language processing pipeline comprising of – dependency parsing, noun phrase chunking, named entity recognition, coreference resolution, and entity resolution to Freebase. The chunker identifies nominal mentions in the text that include our attributes of interest. As discussed later in the paper, we exploit the dependency parse, coreference and entity resolution heavily during various stages of our pipeline.

3 Overview of ReNoun

Since ReNoun aims at extracting triples for attributes not present in head-heavy knowledge bases, one key challenge is that we do not have any labeled data (i.e. known facts) for such attributes, especially in the long tail. Therefore ReNoun has an initial seed fact extraction step that automatically generates a small corpus of relatively precise seed facts for all attributes, so that distant supervision can be employed. The second big challenge is to filter the noise from the resulting extractions.

ReNoun's extraction pipeline, shown in Figure 1, is composed of four stages.

Seed fact extraction: We begin by extracting a small number of high-precision facts for our attributes. For this step, we rely on manually specified lexical patterns that are specifically tailored for noun phrases, but are general enough to be independent of any specific attributes. When applying such patterns, we exploit coreference to make the generated seed facts more precise by requiring the attribute and object noun phrases of a seed fact to refer to the same real-world entity. This is elaborated further in Section 4.

Extraction pattern generation: Utilizing the seed facts, we use distant supervision (Mintz et al., 2009) to learn a set of dependency parse patterns that are used to extract a lot more facts from the text corpus.

Candidate generation: We apply the learned dependency parse patterns from the previous stage to generate a much larger set of extractions. We aggregate all the extractions that give rise to the same fact and store with them the provenance of the extraction. The extractions generated here are called candidates because they are assigned scores that determine how they are used. The application consuming an extraction can decide whether to discard an extraction or use it, and in this case the manner in which it is used, based on the scores we attach to it and the application's precision requirements.

Scoring: In the final stage, we score the facts, reflecting our confidence in their correctness. Intuitively, we give a pattern a high score if it extracts many facts that have semantically similar attributes, and then propagate this score to the facts extracted by the pattern (Section 6).

4 Seed fact extraction

Since we do not have facts, but only attributes, the first phase of ReNoun's pipeline is to extract a set of high-precision seed facts that are used to train more general extraction patterns. ReNoun extracts seed facts using a manually crafted set of extraction rules (see Table 4). However, the extraction rules and the application of these rules are tailored to our task of extracting noun-based attributes.

Specifically, when we apply an extraction rule to generate a triple `(S,A,O)`, we require that (1) `A` is an attribute in our ontology, and (2) the value of `A` and the object `O` corefer to the same real-world

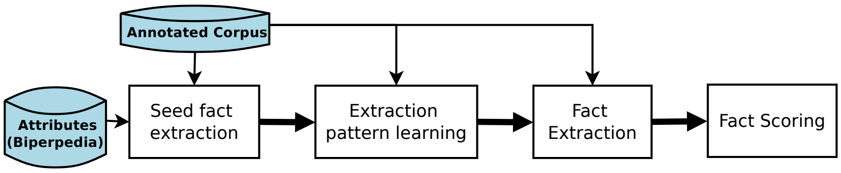


Figure 1: Extraction Pipeline: we begin with a set of high-precision extractors and use distant supervision to train other extractors. We then apply the new extractors and score the resulting triples based on the frequency and coherence of the patterns that produce them.

1. *the* A *of* S, O – the CEO of Google, Larry Page
2. *the* A *of* S *is* O – the CEO of Google is Larry Page
3. O, S A – Larry Page, Google CEO
4. O, S*'s* A – Larry Page, Google's CEO
5. O, [*the*] A *of* S – Larry Page, [the] CEO of Google
6. SAO – Google CEO Larry Page
7. S A, O – Google CEO, Larry Page
8. S*'s* A, O – Google's CEO, Larry Page

Table 4: High precision patterns used for seed fact extraction along with an example of each. Here, the object (O) and the attribute (A) corefer and the subject (S) is in close proximity. In all examples, the resulting fact is `(Google, CEO, Larry Page)`. Patterns are not attribute specific.

entity. For example, in Figure 2, `CEO` is in our ontology and we can use a coreference resolver to infer that `CEO` and `Larry Page` refer to the same entity. The use of coreference follows from the simple observation that objects will often be referred to by nominals, many of which are our attributes of interest. Since the sentence matches our sixth extraction rule, ReNoun extracts the triple `(Google, CEO, Larry Page)`.

Document:
"[Google]$_1$ [CEO]$_2$ [Larry Page]$_2$ started his term in 2011, when [he]$_2$ succeeded [Eric Schmidt]$_3$. [Schmidt]$_3$ has since assumed the role of executive chairman of [the company]$_1$."

(a)

Coreference clusters:

#	Phrases	Freebase ID
1	**Google**, the company	`/m/045c7b`
2	**Larry Page**, CEO, he	`/m/0gjpq`
3	**Eric Schmidt**, Schmidt	`/m/01gqf4`

(b)

Figure 2: Coreference clusters: (a) a document annotated with coreference clusters; (b) a table showing each cluster with the representative phrases in bold and the Freebase ID to which each cluster maps.

We rely on a coreference resolver in the spirit of Haghighi and Klein (2009). The resolver clusters the mentions of entities in a document so the references in each cluster are assumed to refer to the same real-world entity. The resolver also chooses for each cluster a *representative phrase*, which is a proper noun or proper adjective (e.g., *Canadian*). Other phrases in the same cluster can be other proper nouns or adjectives, common nouns like *CEO* or pronouns like *he* in the example. Each cluster is possibly linked by an entity resolver to a Freebase entity using a unique Freebase ID. Figure 2(b) shows the coreference clusters from the sample document, with representative phrases in bold, along with the Freebase ID of each cluster. Note that in our example the phrase *executive chairman*, which is also in our ontology of attributes, is not part of any coreference cluster. Therefore, the fact centered around this attribute in the example will not be part of the seed extractions, but could be extracted in the next phase. The resulting facts use Freebase IDs for the subject and object (for readability, we will use entity names in the rest of this work). In summary, our seed extraction proceeds in two steps. First, we find sentences with candidate attribute-object pairs that corefer and in which the attribute is in our ontology. Second, we match these sentences against our hand-crafted rules to generate the extractions. In Section 7 we show that the precision of our seed facts is 65% for fat head attributes and 80% for long tail ones.

5 Pattern and candidate fact generation

In this section we describe how ReNoun uses the seed facts to learn a much broader set of extraction patterns. ReNoun uses the learned patterns to extract many more candidate facts that are then assigned scores reflecting their quality.

5.1 Dependency patterns

We use the seed facts to learn patterns over dependency parses of text sentences. A dependency parse of a sentence is a directed graph whose vertices correspond to tokens labeled with the word and the POS tag, and the edges are syntactic relations between the corresponding tokens (de Marneffe et al., 2006). A *dependency pattern* is a subgraph of a dependency parse where some words have been replaced by variables, but the POS tags have been retained (called *delexicalization*). A dependency pattern enables us to extract sentences with the same dependency parse as the sentence that generated the pattern, modulo the delexicalized words. We note that one big benefit of using dependency patterns is that they generalize well, as they ignore extra tokens in the sentence that do not belong to the dependency subgraph of interest.

5.2 Generating dependency patterns

The procedure for dependency pattern generation is shown in Algorithm 1, and Figure 3 shows an example of its application. The input to the algorithm is the ontology of attributes, the seed facts (Section 4), and our processed text corpus (Section 2).

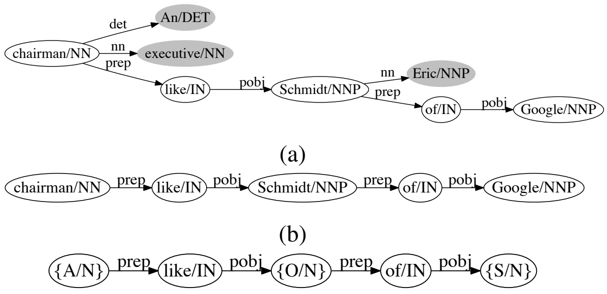
Algorithm 1: Dependency pattern generation

```
input    : Set of attributes $\mathcal{A}$, Seed facts $I$, Corpus $D$.
$\mathcal{P}$ := An empty set of dependency pattern-attribute pairs.
foreach sentence $s \in D$ do
    foreach triple $t = (S, A, O)$ found in $s$ do
        if $t \in I$ then
            $G(s)$ = dependency parse of $s$
            $P'$ = minimal subgraph of $G(s)$
            containing the head tokens of $S$, $A$ and $O$
            $P$ = Delexicalize($P'$, $S$, $A$, $O$)
            $\mathcal{P} = \mathcal{P} \cup \{\langle P, A\rangle\}$
return $\mathcal{P}$
```

Attributes: $\mathcal{A}$ ={executive chairman}

Seed fact: I = {(Google, executive chairman, Eric Schmidt)}

Sentence: s = *"An executive chairman, like Eric Schmidt of Google, wields influence over company operations."*



(a)

(b)

(c)

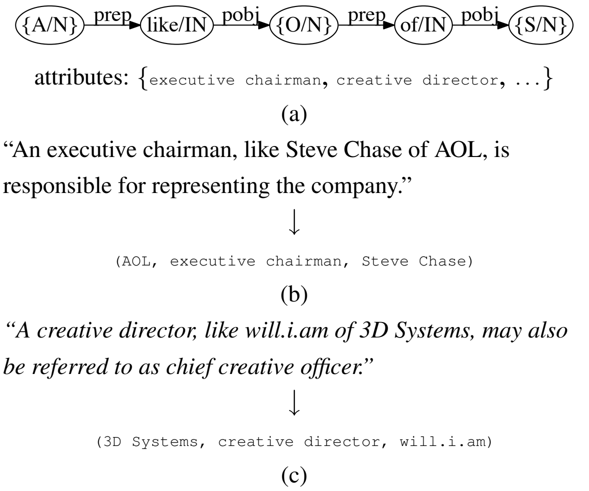
Figure 3: Dependency pattern generation using seed facts, corresponding to Algorithm 1: (a) shows the input to the procedure (dependency parse partially shown); (b) P'; (c) P.

The procedure iterates over the sentences in the corpus, looking for matches between a sentence s and a seed fact f. A sentence s matches f if s contains (i) the attribute in f, and (ii) phrases in coreference clusters that map to the same Freebase IDs as the subject and object of f. When a match is found, we generate a pattern as follows.

We denote by P' the minimal subgraph of the dependency parse of s containing the head tokens of the subject, attribute and object (Figure 3 (b)). We delexicalize the three vertices corresponding to the head tokens of the subject, attribute and object by variables indicating their roles. The POS tag associated with the attribute token is always a noun. The subject and object are additionally allowed to have pronouns and adjectives associated with their tokens. All POS tags corresponding to nouns are lifted to N, in order to match the various types of nouns. We denote the resulting dependency pattern by P and add it to our output, associated with the matched attribute. We note that in principle, the vertex corresponding to the head of the attribute does not need to be delexicalized. However, we do this to improve the efficiency of pattern-matching, since we will often have patterns for different attributes differing only at the attribute vertex.

It is important to note that because of the manner in which the roles of subject and object were assigned during seed fact extraction, the patterns ReNoun generates clearly show which argument will take the role of the subject, and which will take the role of the object. This is in contrast to previous work such as Ollie (Mausam at al., 2012), where the assignment depends on the order in which the arguments are expressed in the sentence from which the fact is being extracted. For example, from the sentence *"Opel was described as GM's most successful subsidiary."* and the seed fact (GM, subsidiary, Opel), the pattern that ReNoun generates will consistently extract facts like (BMW, subsidiary, Rolls-Royce), and not the incorrect inverse, regardless of the relative ordering of the two entities in the sentence.

At this point we have dependency patterns capable of generating more extractions for their seed fact attributes. For efficient matching, we use the output of Algorithm 1 to generate a map from dependency patterns to their attributes with entries like that shown in Figure 4(a). This way, a pattern match can be propagated to all its mapped attributes in one shot, as we explain in Section 5.3. Finally, we discard patterns that do not pass a support threshold, where support is the number of distinct seed facts from which a pattern could be generated.



attributes: {executive chairman, creative director, ...}

(a)

"An executive chairman, like Steve Chase of AOL, is responsible for representing the company."

↓

(AOL, executive chairman, Steve Chase)

(b)

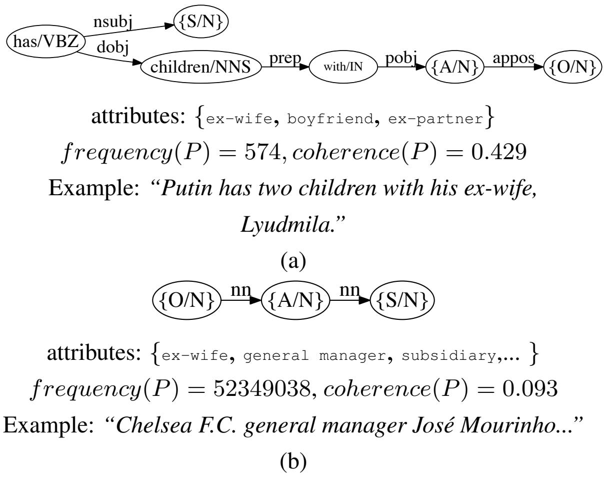
"A creative director, like will.i.am of 3D Systems, may also be referred to as chief creative officer."

↓

(3D Systems, creative director, will.i.am)

(c)

Figure 4: A dependency pattern and its use in extraction: (a) the pattern in our running example and the set of attributes to which it applies; (b) and (c) sentences matching the pattern and the resulting extractions.



attributes: {ex-wife, boyfriend, ex-partner}

$frequency(P) = 574, coherence(P) = 0.429$

Example: *"Putin has two children with his ex-wife, Lyudmila."*

(a)

attributes: {ex-wife, general manager, subsidiary,... }

$frequency(P) = 52349038, coherence(P) = 0.093$

Example: *"Chelsea F.C. general manager José Mourinho..."*

(b)

Figure 5: (a) a coherent pattern extracting facts for semantically similar attributes and (b) an incoherent pattern.

5.3 Applying the dependency patterns

Given the learned patterns, we can now generate new extractions. Each match of a pattern against the corpus will indicate the heads of the potential subject, attribute and object. The noun phrase headed by the token matching the {A/N} vertex is checked against the set of attributes to which the pattern is mapped. If the noun phrase is found among these attributes, then a triple (S, A, O) is constructed from the attribute and the Freebase entities to which the tokens corresponding to the S and O nodes in the pattern are resolved. This triple is then emitted as an extraction along with the pattern that generated it. Figure 4(b) and (c) show two sentences that match the dependency pattern in our running example and the resulting extractions.

Finally, we aggregate our extractions by their generated facts. For each fact f, we save the distinct dependency patterns that yielded f and the total number of times it was found in the corpus.

6 Scoring extracted facts

In this section we describe how we score the candidate facts extracted by applying the dependency patterns in Section 5. Recall that a fact may be obtained from multiple extractions, and assigning scores to each fact (rather than each extraction) enables us to consider all extractions of a fact in aggregate.

We score facts based on the patterns which extract them. Our scheme balances two characteristics of a pattern: its frequency and coherence. *Pattern frequency* is defined as the number of extractions produced by the pattern. Our first observation is that patterns with a large number of extractions are always able to produce correct extractions (in addition to incorrect ones). We also observe that generic patterns produce more erroneous facts compared to more targeted ones. To capture this, we introduce *pattern coherence*, which reflects how targeted a pattern is based on the attributes to which it applies. For example, we observed that if an extraction pattern yields facts for the coherent set of attributes ex-wife, boyfriend, and ex-partner, then its output is consistently good. On the other hand, a pattern that yields facts for a less coherent set of attributes ex-wife, general manager, and subsidiary is more likely to produce noisy extractions. Generic, more incoherent patterns are more sensitive to noise in the linguistic annotation of a document. Figure 5 shows an example pattern for each case, along with its frequency and coherence.

We capture coherence of attributes using word-vector representations of attributes that are created over large text corpora (Mikolov et al., 2013). The word-vector representation $v(w)$ for a word w (multi-word attributes can be preprocessed into single words) is computed in two steps. First, the algorithm counts the number of occurrences of a word w_1 that occurs within the text window centered at w (typically a window of size 10), producing an intermediate vector that potentially has a non-zero value for every word in the corpus. The intermediate vector is then mapped to a much smaller dimension (typically less than 1000) to produce $v(w)$. As shown in (Mikolov et al., 2013), two words w_1 and w_2 for which the cosine distance between $v(w_1)$ and $v(w_2)$ is small tend to be semantically similar. Therefore, a pattern is coherent if it applies to attributes deemed similar as per their word vectors.

Given an extraction pattern P that extracts facts for a set of attributes $\mathcal{A}$, we define the coherence of P to be the average pairwise coherence of all attributes in $\mathcal{A}$, where the pairwise coherence of two attributes a_1 and a_2 is the cosine distance between $v(a_1)$ and $v(a_2)$.

Finally, we compute the score of a fact f by summing the product of frequency and coherence for each pattern of f as shown in Equation 1.

$$S(f) = \sum_{P \in Pat(f)} frequency(P) \times coherence(P) \quad (1)$$

7 Experimental Evaluation

We describe a set of experiments that validate the contributions of ReNoun. In Sections 7.2 and 7.3 we validate our noun-centric approach: we show that extractions based on verb phrases cannot yield the results of ReNoun and that NomBank, the resource used by state of the art in semantic role-labeling for nouns, will not suffice either. In Sections 7.4-7.6 we evaluate the different components of ReNoun and its overall quality, and in Section 7.7 we discuss the cases in which ReNoun was unable to extract any facts.

7.1 Setting

We used the fat head (FH) and long tail (LT) attributes and annotated news corpus described in Section 2. When evaluating facts, we used majority voting among three human judges, unless otherwise noted. The judges were instructed to consider facts with inverted subjects and objects as incorrect. For example, while `(GM, subsidiary, Opel)` is correct, its inverse is incorrect.

7.2 Verb phrases are not enough

State-of-art open information extraction systems like Ollie (Mausam at al., 2012) assume that a relation worth extracting is expressed somewhere in verb form. We show this is not the case and justify our noun-centric approach. In this experiment we compare ReNoun to a custom implementation of Ollie that uses the same corpus as ReNoun and supports multi-word attributes. While Ollie does try to find relations expressed as nouns, its seed facts are relations expressed as verbs.

We randomly sampled each of FH and LT for 100 attributes for which ReNoun extracts facts and asked a judge to find potentially equivalent Ollie relations. Note that we did not require the judge to find exactly the same triple (thereby biasing the experiment towards finding more attribute matches). Furthermore, the judge was instructed that a verb phrase like *advised by* should be considered a match to the ReNoun attribute `advisor`. However, looking at the data, most facts involving the relation *advised* are not synonymous with the *advisor* relation as we think of it (e.g., *"Obama advised Merkel on saving the Euro"*). This observation suggests that there is an even more subtle difference between the meaning of verb expressions and noun-based expressions in text. This experiment, therefore, gives an upper bound on the number of ReNoun attributes that Ollie can cover.

ReNoun	Ollie
flagship company	-
railway minister	-
legal affairs correspondent	-
spokesperson	be spokesperson of
president-elect	be president elect of
co-founder	be co-founder of

Table 5: ReNoun attributes with and without a corresponding Ollie relation.

For FH, not surprisingly, we could find matches for 99 of the 100 attributes. However, for LT, only 31 of the 100 attributes could be found, even under our permissive setting. Most attributes that could not be matched were multi-word noun phrases. While in principle, one could use the Ollie patterns that apply to the head of a multi-word attribute, we found that we generate more interesting patterns for specific multi-word attributes. Table 5 shows examples of attributes with and without verb mappings in Ollie.

We also compare in the other direction and estimate the portion of Ollie relations centered around nouns for which ReNoun fails to extract facts. For this experiment, we randomly sampled 100 Ollie relations that contained common nouns whose objects are concrete values, and looked for equivalent attributes in ReNoun extractions. ReNoun extracts facts for 48 of the Ollie relations. Among the 52 relations with no facts, 25 are not in Biperpedia (which means that ReNoun cannot extract facts for them no matter what). For the other 27 relations, ReNoun did not extract facts for the following reasons. First, some relations expressed actions, which cannot be expressed using nouns only, and are not considered attributes describing the subject entity (e.g., `citation of` in *"Obama's citation* *of the Bible"*). Second, some relations have the object (a common noun) embedded within them (e.g., `have microphone in`) and do not have corresponding attributes that can be expressed using nouns only. The remaining relations either have meaningless extractions or use common noun phrases as arguments. ReNoun only uses proper nouns (i.e., entities) for arguments because facts with common noun arguments are rarely interesting without more context. We note that the majority of the 25 Ollie relations without corresponding Biperpedia attributes also fall into one of the three categories above.

7.3 Comparison against NomBank

In principle, the task of extracting noun-mediated relations can be compared to that of semantic role labeling (SRL) for nouns. The task in SRL is to identify a relation, expressed either through a verb or a noun, map it to a semantic frame, and map the arguments of the relation to the various roles within the frame. State of the art SRL systems, such as that of Johansson and Nugues (2008), are trained on NomBank (Meyers et al., 2004) for handling nominal relations, which also means that they are limited by the knowledge it has. We asked a judge to manually search NomBank for 100 attributes randomly drawn from each of FH and LT for which ReNoun extracts facts. For multi-word attributes, we declare a match if its head word was found. We were able to find 80 matches for the FH attributes and 42 for LT ones. For example, we could not find entries for the noun attributes `coach` or `linebacker` (of a football team). This result is easy to explain by the fact that NomBank only has 4700 attributes.

In addition, for some nouns, the associated frames do not allow for the extraction of triples. For example, all frames for the noun `member` specify one argument only, so in the sentence *"John became a member of ACM"*, the output relation is `(ACM, member)` instead of the desired triple `(ACM, member, John)`.

As we did with Ollie, we also looked at nouns from NomBank for which ReNoun does not extract facts. Out of a random sample of 100 NomBank nouns, ReNoun did not extract facts for 29 nouns (four of which are not in Biperpedia). The majority of the missed nouns cannot be used by ReNoun because they either take single arguments (instead of two) or take either prepositional phrases or common nouns (instead of proper nouns correponding to entities) as one their arguments.

7.4 Quality of seed facts

In Section 4, we described our method for extracting seed facts for our attributes. Applying the method to our corpus resulted in 139M extractions, which boiled down to about 680K unique facts covering 11319 attributes. We sampled 100 random facts from each of FH and LT, and obtained 65% precision for FH seed facts and 80% precision for LT ones. This leads us to two observations.

First, the precision of seed facts for LT attributes is high, which makes them suitable for use as a building block in a distant supervision scheme to learn dependency parse patterns. We are primarily interested in LT attributes, which earlier approaches cannot deal with satisfactorily as we demonstrated above.

Second, LT attributes have higher precision than FH attributes. One reason is that multi-word attributes (which tend to be in LT) are sometimes incorrectly chunked, and only their head words are recognized as attributes (which are more likely to be in FH). For example, in the phrase *"America's German coach, Klinsmann"*, the correct attribute is `German coach` (LT), but bad chunking may produce the attribute `coach` (FH) with `Germany` as the subject. Another reason is that FH attributes are likely to occur in speculative contexts where the presence of the attribute is not always an assertion of a fact. (While both FH and LT attributes can be subject to speculative contexts, we observe this more for FH than LT in our data.) For example, before a person is a `railway minister` of a country, there is little mention of her along with the attribute. However, before a person is elected `president`, there is more media about her candidacy. Speculative contexts, combined with incorrect linguistic analysis of sentences, can result in incorrect seed facts (e.g., from *"Republican favorite for US president, Mitt Romney, visited Ohio"*, we extract the incorrect seed fact `(US, president, Mitt Romney)`).

7.5 Candidate generation

Using the seed facts, we ran our candidate generation algorithm (Section 5). In the first step of the algorithm we produced a total of about 2 million unique dependency patterns. A third of these patterns could extract values for exactly one attribute. Manual inspection of these long tail patterns showed that they were either noise, or do not generalize. We kept patterns supported by at least 10 seed facts, yielding more than 30K patterns.

We then applied the patterns to the corpus. The result was over 460M extractions, aggregated into about 40M unique facts. Of these, about 22M facts were for LT attributes, and 18M for FH. We now evaluate the quality of these facts.

7.6 Scoring extracted facts

In Section 6, we presented a scheme for scoring facts using pattern frequency and coherence. To show its effectiveness we (i) compare it against other scoring schemes, and (ii) show the quality of the top-k facts produced using this scheme, for various k. To compute coherence, we generated attribute word vectors using the word2vec[2] tool trained on a dump of Wikipedia.

First, we compare the quality of our scoring scheme (FREQ_COH) with three other schemes as shown in Table 6. The scheme FREQ is identical to FREQ_COH except that all coherences are set to 1. PATTERN counts the number of distinct patterns that extract the fact while PATTERN_COH sums the pattern coherences. We generated a random sample of 252 FH and LT nouns with no entity disambiguation errors by the underlying natural language processing pipeline. The justification is that none of the schemes we consider here capture such errors. Accounting for such errors requires elaborate signals from the entity linking system, which we leave for future work. For each scoring scheme, we computed the Spearman's rank correlation coefficient ρ between the scores and manual judgments (by three judges). A larger ρ indicates more correlation, and computing ρ was statistically significant (p-value<0.01) for all schemes.

Scheme	**Spearman's ρ**
FREQ	0.486
FREQ_COH	0.495
PATTERN	0.265
PATTERN_COH	0.257

Table 6: Scoring schemes

FREQ and FREQ_COH dominate, which shows that considering the frequency with which patterns perform extraction helps. The two schemes, however, are very close to each other. We observed that adding coherence helps when two facts have similar frequencies, but this effect is tempered when considering a large number of facts.

	FH		**LT**	
k	Precision	#Attr	Precision	#Attr
10^2	1.00	8	1.00	50
10^3	0.98	36	1.00	294
10^4	0.96	78	0.98	1548
10^5	0.82	106	0.96	5093
10^6	0.74	124	0.70	7821
All	0.18	141	0.26	11178

Table 7: Precision of random samples of the top-k scoring facts, along with the attribute yield.

Second, we evaluate the scoring of facts generated by ReNoun by the precision of top-k results for several values of k. In this evaluation, facts with disambiguation errors are counted as wrong. The particular context in which ReNoun is applied will determine where in the ordering to set the threshold of facts to consider. We compute precision based on a sample of 50 randomly chosen facts for each k. Table 7 shows the precision results, along with the number of distinct attributes (#Attr) for which values are extracted at each k.

As we can see, ReNoun is capable of generating a large number of high quality facts ($\geq$70% precise at 1M), which our scoring method manages to successfully surface to the top. The major sources of error were (i) incorrect dependency parsing mainly due to errors in boilerplate text removal from news documents, (ii) incorrect coreference resolution of pronouns, (iii) incorrect entity resolution against Freebase, and (iv) cases where a triple is not sufficient (e.g., `ambassador` where both arguments are countries.)

7.7 Missed extractions

We analyze why ReNoun does not extract facts for certain attributes. For FH, we investigate all the 77 attributes for which ReNoun is missing facts. For LT, there are about 50K attributes without corresponding facts, and we use a random sample of 100 of those attributes.

Cause	**FH**	**LT**	**Example**
Vague	23	37	`culture`
Numeric	4	26	`rainfall`
Object not KB entity	11	6	`email`
Plural	30	15	`member firms`
Bad attribute / misspell	3	4	`newsies`
Value expected	6	12	`nationality`
Total	77	100	

Table 8: Analysis of attributes with no extractions.

[2] https://code.google.com/p/word2vec/

Table 8 shows the categorization of the missed attributes. The first three categories are cases that are currently outside the scope of ReNoun: vague attributes whose values are long narratives, numeric attributes, and typed attributes (e.g., email) whose values are not modeled as Freebase entities. The next two categories are due to limitations of the ontology, e.g., plural forms of attributes are not always synonymized with singular forms and some attributes are bad. Finally, the "Value expected" category contains the attributes for which ReNoun *should* have extracted values. One reason for missing values is that the corpus itself does not contain values of all attributes. Another reason is that some attributes are not verbalized in text. For example, attributes like `nationality` are usually not explicitly stated when expressed in text.

8 Related Work

Open information extraction (OIE) was introduced by Banko et al. (2007). For a pair of noun phrases, their system, TEXTRUNNER, looks for the attribute (or more generally the relation) in the text between them and uses a classifier to judge the trustworthiness of an extraction. WOE^{parse} (Wu and Weld, 2010) extends this by using dependency parsing to connect the subject and object. Both systems assume that the attribute is between its two arguments, an assumption that ReNoun drops since it is not suitable for nominal attributes.

Closest to our work are ReVerb (Fader et al., 2011) and Ollie (Mausam at al., 2012). ReVerb uses POS tag patterns to locate verb relations and then looks at noun phrases to the left and right for arguments. Ollie uses the ReVerb extractions as its seeds to train patterns that can further extract triples. While Ollie's patterns themselves are not limited to verb relations (they also support noun relations), the ReVerb seeds are limited to verbs, which makes Ollie's coverage on noun relations also limited. In comparison, ReNoun take a noun-centric approach and extracts many facts that do not exist in Ollie.clo

ClausIE (Del Corro and Gemulla, 2013) is an OIE framework that exploits knowledge about the grammar of the English language to find clauses in a sentence using its dependency parse. The clauses are subsequently used to generate extractions at multiple granularities, possibly with more than triples. While ClausIE comes with a predefined set of rules on how to extract facts from a dependency parse, ReNoun learns such rules from its seed facts.

Finally, Nakashole et al. (2014) and Mintz et al. (2009) find additional facts for attributes that already have facts in a knowledge base. In contrast, ReNoun is an OIE framework whose goal is to find facts for attributes without existing facts.

9 Conclusions

We described ReNoun, an open information extraction system for nominal attributes that focuses on the long tail. The key to our approach is to start from a large ontology of nominal attributes and apply noun-specific manual patterns on a large preprocessed corpus (via standard NLP components) to extract precise seed facts. We then learn a set of dependency patterns, which are used to generate a much larger set of candidate facts. We proposed a scoring function for filtering candidate facts based on pattern frequency and coherence. We demonstrated that the majority of long tail attributes in ReNoun do not have corresponding verbs in Ollie. Finally, our experiments show that our scoring function is effective in filtering candidate facts (top-1M facts are $\geq$70% precise).

In the future, we plan to extend ReNoun to extract triples whose components are not limited to Freebase IDs. As an example, extending ReNoun to handle numerical or typed attributes would involve extending our extraction pattern learning to accommodate units (e.g., kilograms) and other special data formats (e.g., addresses).

Acknowledgments

We would like to thank Luna Dong, Anjuli Kannan, Tara McIntosh, and Fei Wu for many discussions about the paper.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives 2007. DBpedia: A Nucleus for a Web of Open Data. In *Proceedings of the International Semantic Web Conference*.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni 2007. Open Information Extraction from the Web. In *Proceedings of the International Joint Conference on Artificial Intelligence* .

Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the International Conference on Management of Data*.

Michael J. Cafarella, Alon Y. Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang 2008. WebTables: Exploring the Power of Tables on the Web. In *Proceedings of the VLDB Endowment*.

Luciano Del Corro and Rainer Gemulla. 2013. ClausIE: Clause-based Open Information Extraction. In *Proceedings of the International World Wide Web Conference*.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of Language Resources and Evaluation*.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying Relations for Open Information Extraction. In *Proceedings of Empirical Methods in Natural Language Processing*.

Rahul Gupta, Alon Halevy, Xuezhi Wang, Steven Whang, and Fei Wu. 2014. Biperpedia: An Ontology for Search Applications. In *Proceedings of the VLDB Endowment*.

Aria Haghighi and Dan Klein 2009. Simple Coreference Resolution with Rich Syntactic and Semantic Features. In *Proceedings of Empirical Methods in Natural Language Processing*.

Richard Johansson and Pierre Nugues 2008. The Effect of Syntactic Representation on Semantic Role Labeling. In *Proceedings of the International Conference on Computational Linguistics*.

Taesung Lee, Zhongyuan Wang, Haixun Wang, and Seung-won Hwang 2013. Attribute Extraction and Scoring: A Probabilistic Approach. In *Proceedings of the International Conference on Data Engineering* .

Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. 2012. Open Language Learning for Information Extraction. In *Proceedings of Empirical Methods in Natural Language Processing.*

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv*.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant Supervision for Relation Extraction Without Labeled Data. In *Proceedings of the Association for Computational Linguistics*.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. Annotating Noun Argument Structure for NomBank. In *Proceedings of Language Resources and Evaluation*.

Nitin Madnani and Bonnie J. Dorr. 2010. Generating Phrasal and Sentential Paraphrases: A Survey of Data-Driven Methods. In *Computational Linguistics 36(3)*.

Ndapandula Nakashole, Martin Theobald, and Gerhard Weikum. 2011. Scalable Knowledge Harvesting with High Precision and High Recall. In *Proceedings of Web Search and Data Mining*.

Marius Pasca. 2014. Acquisition of Open-domain Classes via Intersective Semantics. In *Proceedings of the International World Wide Web Conference*.

Marius Pasca and Benjamin Van Durme. 2007. What You Seek Is What You Get: Extraction of Class Attributes from Query Logs. In *Proceedings of the International Joint Conference on Artificial Intelligence* .

Uma Sawant and Soumen Chakrabarti. 2013. Learning Joint Query Interpretation and Response Ranking. In *Proceedings of the International World Wide Web Conference*.

Amit Singhal. 2012 *Introducing the Knowledge Graph: things, not strings* `http://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html`

Fei Wu and Daniel S. Weld. 2010. Open Information Extraction Using Wikipedia. In *Proceedings of the the Association for Computational Linguistics*.

Mohamed Yahya, Klaus Berberich, Shady Elbassuoni, Maya Ramanath, Volker Tresp, and Gerhard Weikum. 2012. Natural Language Questions for the Web of Data. In *Proceedings of Empirical Methods in Natural Language Processing*.