Report

Building A Fact-Checking Engine

Group: SW 9000

https://github.com/sibarsoumi/factchecker

Data Science Group, Paderborn University, Paderborn, Germany

Sibar Soumi

sibar@mail.uni-paderborn.de

Edgard Hois

ehois@mail.uni-paderborn.de

Table of Contents

1 Introduction

In this report an approach for a fact-checking engine using a knowledge base will be introduced and evaluated.

The input is a knowledge base containing statements in Turtle triple form (subject, predicate, object). The output is a file containing the truth value for each statement of a given input file.

This takes place in two main steps:

1) Reading and Preprocessing
2) Calculating truth value for one statement

which will be described in the following.

2 Approach

2.1 Reading and Preprocessing

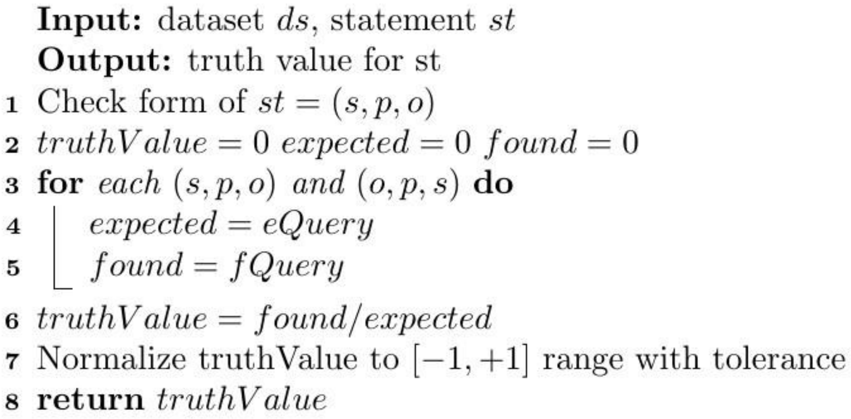
To start, the input datasets need to be read and transformed into a format that is fast to process. To do so, the data is read and converted to a TDB dataset. Now it is possible to iterate over the dataset containing the list of statements to be evaluated.

2.2 Calculating truth value for one statement

The approach to calculate the truth value for each statement is described in algorithm 1:

The input for the algorithm is the dataset and a single statement [consisting of a triple (subject, predicate, object)]. The output is a truth value for the given statement.

Algorithm 1:



```
Input: dataset $ds$, statement $st$
Output: truth value for st
Check form of $st = (s, p, o)$
$truthValue = 0$ $expected = 0$ $found = 0$
for $each$ $(s, p, o)$ $and$ $(o, p, s)$ do
    $expected = eQuery$
    $found = fQuery$
$truthValue = found/expected$
Normalize truthValue to $[-1, +1]$ range with tolerance
return $truthValue$
```

2.2.1 Check form

To guarantee that a valid fact in Turtle is used, a form check is performed first (line 1). Therefore, a fact in the form of (subject, predicate, object) can be assumed as given hereafter.

2.2.2 Initializations

The truth value is initialized as 0 to be modified at the end of the calculations (line 2). The #expected and #found are also initialized as 0, in the following #expected will count the number of predicates found with the queries (line 2).

2.2.3 Queries for direction (Subject Predicate Object)

To explain the next steps, the following triple will be used as an example

:AngelaMerkel :leaderOf :Germany.

The next step is to find entities other than the current subject (here: :AngelaMerkel) and object (:Germany) from the knowledge base that are linked through the same predicate (:leaderOf).

For example:

:EmmanuelMacron :leaderOf :France.

:DonaldTrump :leaderOf :USA.

:TheresaMay :leaderOf :UK.

:RecepTayyipErdogan :leaderOf :Turkey.

For each of those subject/object pairs (e.g. *:EmmanuelMacron :France*), relations other than *:leaderOf* that link these entities are found and the number of unique predicates of those counted as #expected.

The query for these steps to calculate #expected (eQuery, line 4) is the following:

```
SELECT (COUNT (DISTINCT ?otherExpectedPredicate) AS ?countOfExpected) \r\n" +
            "WHERE { \r\n" +
            "?anotherSubject <"+predicate+"> ?anotherObject .\r\n"+
            "?anotherSubject ?otherExpectedPredicate ?anotherObject.\r\n"+
            "FILTER(?otherExpectedPredicate != <"+predicate+">).\r\n"+
            "}
```

Next, the #found value is calculated by checking how many of these predicates also link the subject (*:AngelaMerkel*) and object (*:Germany*) of the statement.

The query (fQuery, line 5) for that is:

```
"SELECT (COUNT (DISTINCT ?otherExpectedPredicate) AS ?countOfExpectedAndExisting) \r\n" +
          "WHERE { \r\n" +
          "?anotherSubject <"+predicate+"> ?anotherObject .\r\n"+
          "?anotherSubject ?otherExpectedPredicate ?anotherObject.\r\n"+
          "FILTER(?otherExpectedPredicate != <"+predicate+">).\r\n"+
          "<"+subject+"> ?otherExpectedPredicate <"+object+">.\r\n"+
                   "}
```

2.2.4 Queries for direction (Object Predicate Subject)

Adding the predicates for the other direction has been proven do be advantageous. The other direction is analogous to 2.2.3 with subject and object swapped and the unique new predicates added to #expected or #found respectively.

2.2.5 Compute truth value

The counted values for #found and #expected are now used to compute the truth value.

The truth values are computed as the ratio: #found / #expected. The result will have a range of 0 to 1 (line 6), therefore it needs to be converted to the required range of -1 to 1.

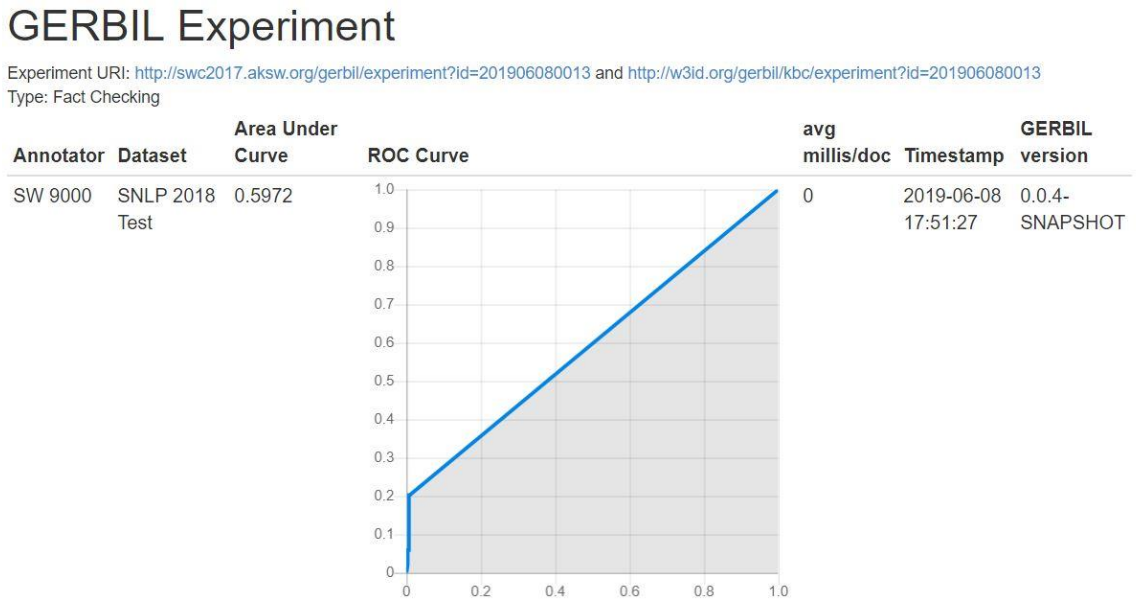
To avoid overfitting, a tolerance margin is allowed. Thus, the result is converted to [-1, +1.7] instead of [-1,+1] and values > +1.0 are rounded down to +1.0 (line 7).

The result for each statement is saved in the "result.ttl" file.

3 Evaluation

The system returns the file "result.ttl".

Uploading the file at *http://swc2017.aksw.org/gerbil/config* yields a result of 0.5972 (rounded).



GERBIL Experiment

Experiment URI: http://swc2017.aksw.org/gerbil/experiment?id=201906080013 and http://w3id.org/gerbil/kbc/experiment?id=201906080013
Type: Fact Checking

Annotator	Dataset	Area Under Curve	ROC Curve	avg millis/doc	Timestamp	GERBIL version
SW 9000	SNLP 2018 Test	0.5972		0	2019-06-08 17:51:27	0.0.4-SNAPSHOT

The random guesser had a result of 0.5211 and is therefore exceeded.

4 Weak points/Ways to improve

In the following 10 examples will be shown (5 false and 5 true facts) that are not part of the training or test dataset and that the system is not able to handle.

4.1 Examples of high truth values for wrong facts

The following 5 examples are considered as true by the system, even though they are false:

dbr:Canada dbo:officialLanguage dbr:Slavey_language .
dbr:Norway dbo:officialLanguage dbr:Kven_language .
dbr:Ecuador dbo:officialLanguage dbr:Shuar_language .
dbr:Argentina dbo:officialLanguage dbr:Toba_Qom_language .
dbr:Ecuador dbo:officialLanguage dbr:Shuar_language .

The truth value for the following triple will be evaluated in detail:

<http://dbpedia.org/resource/Canada><http://dbpedia.org/ontology/officialLanguage> <http://dbpedia.org/resource/Slavey_language>.

Using the approach of detecting pairs of subject and object with the same predicate re turns 559 pairs in total containing

<http://dbpedia.org/ontology/officialLanguage>

as the predicate.

An example for such a pair forming a triple with the same predicate is the following:

<http://dbpedia.org/resource/Sweden><http://dbpedia.org/ontology/officialLanguage> <http://dbpedia.org/resource/Swedish_language>.

The other relations connecting those 559 pairs are:

<http://dbpedia.org/ontology/language>
<http://dbpedia.org/ontology/regionalLanguage>
<http://dbpedia.org/ontology/ethnicGroup>
<http://www.w3.org/2000/01/rdf-schema#seeAlso>

This results in a value of 4 for #expected. 2 of those are also found between

<http://dbpedia.org/resource/Canada>

and

<http://dbpedia.org/resource/Slavey_language>

the other 2 are not.

#found / #expected = 2 / 4 = 0.5
Using the normalization with tolerance margin yields a value of 0.62 and so the statement is considered true, although it is false.
The calculations for the other 4 examples are analogously.

4.2 Examples of low truth values for correct facts

The following 5 examples are considered as false by the system, even though they are true:

dbr:Club_Atlas dbo:owner dbr:Grupo_Salinas .
dbr:Club_Atlas dbo:chairman dbr:Gustavo_Guzmán .

dbr:United_States dbo:governmentType dbr:Federalism .
dbr:Chris_Brokaw dbo:birthPlace dbr:New_York_City .
dbr:DB_Schenker dbo:locationCity dbr:Frankfurt .

The truth value for the following triple will be evaluated in detail:

<http://dbpedia.org/resource/Club_Atlas> <http://dbpedia.org/ontology/owner>
<http://dbpedia.org/resource/Grupo_Salinas>.

There are 60346 triples containing

<http://dbpedia.org/ontology/owner>

as predicate, the subject/object pairs of these triples contain 66 relations other than *<http://dbpedia.org/ontology/owner>*, for example:

<http://dbpedia.org/ontology/operator>
<http://dbpedia.org/ontology/tenant>
<http://dbpedia.org/ontology/location>

None of these relations exists for

<http://dbpedia.org/resource/Club_Atlas>

and

<http://dbpedia.org/resource/Grupo_Salinas>.

Therefore, #found / #expected = 0 / 66 = 0
Using the normalization with tolerance margin yields a value of -1 and so the statement is considered false, even though it is true.
The calculations for the other 4 examples are analogously.

4.3 Room for improvement

A way to improve the algorithm might be to add a weight function that assigns a weight to the predicates added to #found by counting the number of appearances in relation to the number of tested subject/object pairs. This might help to balance out predicates added to #found that only appear rarely but influence the truth value immensely.