FactCheck: Validating RDF Triples Using Textual Evidence

Zafar Habeeb Syed
Data Science Group, Paderborn University
Paderborn, Germany
zsyed@mail.uni-paderborn.de

Michael Röder
Data Science Group, Paderborn University
Paderborn, Germany
michael.roeder@upb.de

Axel-Cyrille Ngonga Ngomo
Data Science Group, Paderborn University
Paderborn, Germany
axel.ngonga@upb.de

ABSTRACT

With the increasing uptake of knowledge graphs comes an increasing need for validating the knowledge contained in these graphs. However, the sheer size and number of knowledge bases used in real-world applications makes manual fact checking impractical. In this paper, we employ sentence coherence features gathered from trustworthy source documents to outperform the state of the art in fact checking. Our approach, FactCheck, uses this information to score how likely a fact is to be true and provides the user the evidence used to validate the input facts. We evaluated our approach on two different benchmark datasets and two different corpora. Our results show that FactCheck outperforms the state of the art by up to 13.3% in F-measure and 19.3% AUC. FactCheck is open-source and is available at https://github.com/dice-group/FactCheck.

KEYWORDS

Fact Checking; Knowledge Graphs; Dependency parsing

ACM Reference Format:
Zafar Habeeb Syed, Michael Röder, and Axel-Cyrille Ngonga Ngomo. 2018. FactCheck: Validating RDF Triples Using Textual Evidence. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3269206.3269308

1 INTRODUCTION

With the increasing use of knowledge graphs (e.g., DBpedia [6], Google Knowledge Vault [2], Yago [12]) in commercial applications comes an increasing need to ensure that the facts contained in knowledge graphs reflect reality. While knowledge base curation has been performed manually in the past [4, 12], the growth of knowledge bases in both size and number makes this approach increasingly impractical. Hence, automatic approaches that support validity checks for facts in a knowledge base have been devised over the last few years. Fact validation systems (e.g., DeFacto [4]) often rely on word proximity analysis combined with string search in a Web corpus to determine how likely a fact is to be true.

In this paper, we go beyond the state of the art in fact checking by combining deep sentence parsing with topic coherence on static corpora to determine how likely a fact is to be true. Our approach is implemented in the novel automated approach to validate RDF triples dubbed FactCheck.

We evaluate our approach by using two static text corpora: the English Wikipedia and ClueWeb12.[1] We evaluate and report the results of DeFacto and FactCheck using the existing benchmark dataset FactBench [4] and a newly created dataset, which addresses some of the known weaknesses of FactBench. Our experiments show that FactCheck outperforms DeFacto on all datasets with both reference corpora by up to 13.3% F-measure and 19.3% AUC.

2 RELATED WORK

There are several approaches for validating a set of given facts based on knowledge gathered from web searches or a reference corpus. Most of these *fact finders* work in a similar way: First, a bipartite graph of facts and sources is generated. A source is connected to a fact if this source gives evidence for this particular fact. The scores—trustworthiness of sources and truth values of facts—are calculated based on this graph. [9] gives a good overview of several implementation of this principle. The state of the art for fact checking in RDF datasets is DeFacto[4]. In contrast to other approaches, DeFacto focuses on evaluating single facts and uses a combination of textual evidence and machine learning to this end. However, it relies on string matching and string similarities and does not take sentence structures into consideration. We address this drawback with FactCheck.

3 FACTCHECK

The architecture underlying FactCheck is shown in Figure 1. The input to the framework is an RDF triple, e.g., `<Albert_Einstein, :award, Nobel_Prize_for_Physics>`. This input triple is first verbalised, i.e., transformed into natural language. To this end, we use surface form libraries for the resources (e.g., from [4, 5]) to generate possible verbalizations of the input triple. These verbalizations of the triple are used to search through a static corpus and gather documents containing sentences similar to the verbalizations (we use Elasticsearch[2]). All sentences of the retrieve document which are similar to a verbalization of the input triple are used as evidence for the given fact. For every evidence, FactCheck extracts evidence features and for every document, trust worthiness features are further extracted. These features are used as input to a trained machine learning model, which returns a confidence value for the triple between 0 and 1. In the following, we focus on the features that ensure FactCheck outperforms DeFacto—namely dependency parsing and word coherence. The other features, which FactCheck shared with DeFacto, can be found in Section 3.3.

[1] https://lemurproject.org/clueweb12/
[2] https://www.elastic.co/products/elasticsearch

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3269308

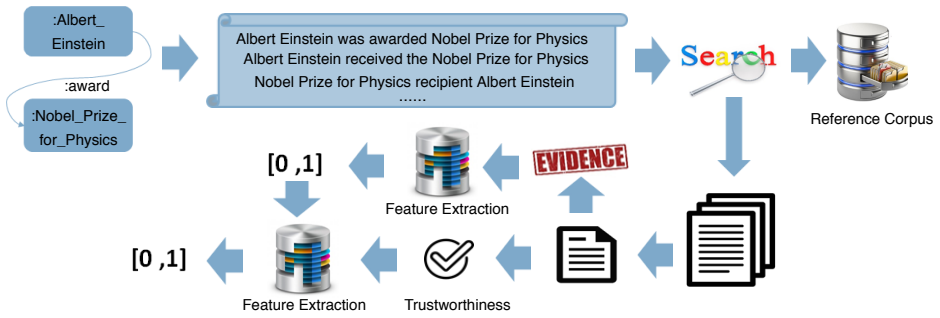

Figure 1: Overview of the architecture of FactCheck

3.1 Dependency Parse Features

To compute this feature, we begin by applying co-reference resolution to the documents returned by the search engine. We then extract sentence(s) that contain at least one verbalization of the predicate of the input triple. We then construct the dependency trees of these sentences and identify enhanced dependencies between the words of these sentences [11]. These enhanced dependencies are analyzed to detect the presence of direct dependencies involving the *subject*, *object* and *predicate* of the input triple. For example, consider the following input triple:

```
<Albert_Einstein, received, Nobel_Prize_in_Physics>
```

The textual evidences extracted for the input triple from two different source documents are shown below.

Albert Einstein received the 1921 Nobel Prize in Physics "for his services to theoretical physics, and especially for his discovery of the law of the photoelectric effect", a pivotal step in the evolution of quantum theory.

In 1945, after having been nominated by Albert Einstein, Pauli received the Nobel Prize in Physics for his "decisive contribution through his discovery of a new law of Nature, the exclusion principle or Pauli principle".

For the first proof phrase, we find a dependency where the *subject* and *object* depend directly on the *predicate*. This sentence will be assigned the feature value 1. If no direct dependencies can be found between the *subject*, *object* and *predicate* (see, e.g., Figure 2), the feature is assigned the value 0.

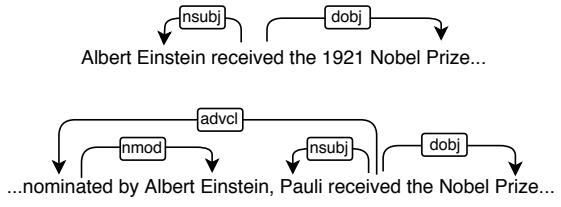

Figure 2: Dependency parse trees of the two example evidences.

3.2 Topic terms based on word coherence

In FactCheck, we use a hybrid approach by combining word set coherences [10] and topic indicators [7] to calculate trustworthiness features of a document. A list of pages related to the input triple are obtained by querying Wikipedia with the *subject* and *object* labels separately. A frequency analysis is performed on the retrieved pages and top n terms with highest frequencies are selected (excluding stop words).[3] We then calculate coherence values for these terms by forming word sets with *subject* and *object* labels respectively. All the terms whose coherence value is greater than a certain threshold are selected and used as topic terms.[4] Finally these terms are used to calculate the trustworthiness features as described in [4].

For our experiments we use the normalized pointwise mutual information (NPMI) coherence [1] defined as the average NPMI value of all word pairs of the given word set. The NPMI value of a word pair w_i and w_j is determined by Equation 1, where $\epsilon = 10^{-12}$ and $P(w_i)$,$P(w_j)$ as well as $P(w_i, w_j)$ are probabilities gathered from the English Wikipedia using a sliding window of 10 words [10].

$$NPMI(w_i, w_j) = \left(\frac{\log\left(\frac{P(w_i,w_j)+\epsilon}{P(w_i)P(w_j)}\right)}{\log\left(P(w_i,w_j)+\epsilon\right)} \right) \quad (1)$$

3.3 Other Features

The following features are shared with DeFacto.
Evidence Classification: BOA pattern score, token distance, Wordnet expansion, total occurrence, page title, end of sentence.
Trustworthiness: topic coverage, topic majority in search results, topic majority in the web, domain and range verification, statistical triple evidence. In addition, we also use combinations of features from both categories. An exact description of the features can be found in [4].

4 EXPERIMENTAL SETUP AND DATASETS

4.1 Reference corpora

We used two reference corpora—ClueWeb and the English Wikipedia. ClueWeb was generated by removing all HTML tags from the orginal files of ClueWeb12 and extracting relevant information such as pageid, text, URL and the page title. In addition, we also used the PageRank information provided by the ClueWeb page rank[5] to configure our search engine. Our final corpus contained approx.

[3] For our experiments, we are using $n = 50$.
[4] Based on our observation for a sample of topics chosen randomly, we are using 0.2 as threshold for the coherence value.
[5] http://www.lemurproject.org/clueweb12/PageRank.php

420 million plain text documents. The second corpus was created by extracting plain text from all articles of the English Wikipedia and comprises 5.4 million plain text documents.[6]

4.2 Datasets

In our experiments, we used two different datasets for training and evaluating the classifier underlying FactCheck: FactBench from [4] and the BPDP dataset created by us. FactBench is a multilingual benchmark dataset for the evaluation of fact validation algorithms[4].[7] FactBench is divided into train and test sets both containing positive and negative facts. The positive facts are generated by issuing SPARQL and MQL queries and selecting the top 150 results for each relation. The negative facts are derived by modifying positive facts. Given a KB K, let S and O be the set of all subjects and objects respectively in K. Let P be set of all properties.[8] Let triple $X = (s, p, o)$ represent a positive example in K and r a function which randomly selects an element from the given set. The following negative facts sets are generated corresponding to X while still following domain and range restrictions.

Domain: $\{(s', p, o) |\ s' = r(S), (s', p, o) \notin K\}$
Range: $\{(s, p, o') |\ o' = r(O), (s, p, o') \notin K\}$
Domain-Range: $\{(s', p, o') |\ s' = r(S), o' = r(O), (s', p, o') \notin K\}$
Property: $\{(s, p', o) |\ p' = r(P), (s, p', o) \notin K\}$
Random: $\{(s', p', o') |\ s' = r(S), p' = r(P), o' = r(O), (s', p', o') \notin K\}$
Mix: This set is built by randomly selecting 20% of the triples of each of the above sets

FactBench comprises 1,500 positive facts which are divided into a training and a test set of equal size. For each of the six categories of wrong facts, FactBench contains 1500 facts that are also separated into two halves for training and testing.

Our second evaluation dataset, the Birth Place/Death Place (BPDP) dataset, was created based on the observation that some fact checking approaches only check if subject and object have a relation to each other. However, the type of the relation, i.e., if it matches the property of the given fact, is not always taken into account. We hence created a more difficult dataset by generating false facts involving pairs of entities that are part of a triple. To this end, we queried DBpedia for persons which have their birth and death places in two different countries. We sorted this list in descending order based on PageRank scores as implemented in [8]. Two researchers checked this list to make sure that the countries were indeed different and not simply renamed because of historical events. The first 103 correct persons the two researchers agreed on were chosen. For each person, four facts were added to the dataset. Two facts have the correct birth and death places while the other two facts are generated by swapping birth and death place. Overall, the BPDP dataset comprises 412 facts separated in a train and a test set each containing an equal number of positive and negative facts.[9]

[6] We used the dump from the 1st October 2017. All documents of ClueWeb and Wikipedia were indexed separately using Elasticsearch on a cluster of 7 nodes with a combined storage capacity of 15TB and computing capacity of 32GB RAM per node.
[7] https://github.com/SmartDataAnalytics/FactBench
[8] FactBench contains 10 properties from the DBpedia ontology: award, birth, death, foundationPlace, leader, nbateam, publicationDate, spouse, starring and subsidiary.
[9] The dataset is available at https://hobbitdata.informatik.uni-leipzig.de/FactCheck/.

Table 1: Classification results of DeFacto and FactCheck on the Mix subset.

	Wikipedia					ClueWeb				
	P	R	F1	AUC	RMSE	P	R	F1	AUC	RMSE
DeFacto										
J48	**0.809**	**0.802**	**0.803**	0.827	**0.392**	**0.832**	**0.830**	**0.830**	0.829	0.384
Simple Logistics	0.795	0.784	0.784	**0.847**	0.394	0.819	0.818	0.818	**0.871**	**0.378**
Naive Bayes	0.726	0.600	0.554	0.820	0.620	0.762	0.740	0.740	0.863	0.480
SMO	0.787	0.777	0.777	0.782	0.471	0.821	0.820	0.820	0.816	0.424
FactCheck										
J48	**0.840**	**0.837**	**0.837**	0.825	0.391	0.891	**0.885**	**0.885**	0.925	0.299
Simple Logistics	0.836	0.835	0.835	0.883	**0.368**	**0.893**	0.883	0.883	**0.955**	**0.288**
Naive Bayes	0.768	0.684	0.659	**0.894**	0.555	0.808	0.760	0.760	0.869	0.475
SMO	0.826	0.825	0.825	0.825	0.418	0.885	0.871	0.871	0.877	0.358

4.3 Preparations

For the experiments, DeFacto and FactCheck were trained on the train part of the datasets used in the experiment. This includes the generation of feature vectors for training classification models. We use four different classifiers—J48, Simple Logistics, Naive Bayes and sequential minimal optimization (SMO)—of the Weka machine learning toolkit [3]. Once the final models were created, we generated the feature vectors for facts from the test set before classifying each single fact to be either true or false. We carried out experiments for DeFacto and FactCheck using all subsets of FactBench as well as ClueWeb and Wikipedia as reference corpora. For the BPDP dataset, only experiments with Wikipedia as reference were carried out. Note that for a fair comparison, we extended DeFacto to enable it to use our ClueWeb and Wikipedia as reference corpora.

The classification results were evaluated using precision, recall, F1-measure, ROC-AUC as well as root mean square error (RMSE). Owing to limited space, only a selected subset of the experimental results will be presented and discussed in the following section.[10]

5 RESULTS AND DISCUSSION

The aim of our experiments is to answer the following questions.

5.1 Q1: Impact of Corpus Size on Classification

To answer **Q1**, we compare the results of both systems on the **Mix** subset of FactBench using ClueWeb and Wikipedia as reference corpora. Table 1 shows the results of DeFacto and FactCheck. The F-Measures of both systems are higher for all four classification algorithms when using the larger ClueWeb corpus. For the best classifiers, the F1-score increases by 0.027 for DeFacto and 0.048 for FactCheck, respectively. The reason for this improvement is that the number of source documents found in ClueWeb is higher compared to Wikipedia for the majority of positive facts. In particular, no source documents for facts corresponding to the relations *foundationPlace* and *subsidiary* are found in Wikipedia.

5.2 Q2: Comparison of FactCheck and DeFacto

Table 1 already showed that FactCheck performs better than DeFacto on the **Mix** subset for both reference corpora. FactCheck's

[10] All results are available in our appendix which can be accessed at https://hobbitdata.informatik.uni-leipzig.de/FactCheck/cikm2018-appendix.pdf.

Table 2: Classification performance of FactCheck and DeFacto on the Property and BPDP dataset.

	FactCheck					DeFacto				
	P	R	F1	AUC	RMSE	P	R	F1	AUC	RMSE
Property subset										
J48	**0.819**	**0.819**	**0.819**	**0.865**	**0.374**	0.713	0.703	0.700	0.758	0.452
Simple Logistics	0.806	0.805	0.805	0.860	0.381	0.709	0.700	0.700	**0.781**	**0.441**
Naive Bayes	0.677	0.660	0.660	0.747	0.551	0.661	0.654	0.650	0.724	0.519
SMO	0.812	0.812	0.812	0.812	0.434	**0.720**	**0.710**	**0.710**	0.719	0.530
BPDP dataset										
J48	**0.814**	**0.811**	**0.811**	0.878	0.368	**0.691**	**0.678**	**0.678**	0.715	0.494
Simple Logistics	0.792	0.783	0.782	**0.922**	**0.348**	0.626	0.623	0.623	**0.729**	**0.487**
Naive Bayes	0.800	0.760	0.760	0.869	0.475	0.623	0.623	0.622	0.68	0.579
SMO	0.811	0.811	0.811	0.811	0.434	0.682	0.673	0.673	0.673	0.497

F1-score is 0.034 and 0.055 points higher compared to DeFacto when using Wikipedia and ClueWeb, respectively.

According to [4], the **Property** subset is the most difficult part of FactBench. Table 2 shows the results of both systems for this subset using the ClueWeb reference corpus. Again, FactCheck clearly outperforms DeFacto by 0.109 in terms of F1-measure and 0.084 in terms of AUC. A comparison with other FactBench subsets shows that while DeFacto is stable at classifying positive and negative examples on the other sets, its ability to identify false facts of the **Property** subset is limited. This is due to the fact that the classification features in DeFacto merely check for the presence of *property* patterns in the proof phrases and fail to consider their relation with the *subject* and *object* label of the input triple. However, with the Dependency Parse feature of FactCheck we pay attention to the relations between *subject*, *property* and *object* of the input triple while confirming facts from sentences. This enables FactCheck to detect false facts more accurately.

This observation is supported by the comparison of the system results for the BPDP dataset on the Wikipedia corpus. Although the dataset has been created to be even more challenging than the **Property** subset of FactBench, FactCheck still reaches a high F1-measure which is 0.133 above the best F1-measure reached by DeFacto as shown in Table 2. This supports FactCheck's advantage over similar systems. Note that, the Wilcoxon signed-rank test for the scores of both systems for a sample of 150 examples indicates the results are significant at $p \leq 0.01$.

5.3 Q3: Feature subset analysis

Our third aim was to check which of the features used by FactCheck are helpful for the classification of facts. To this end, we used subset evaluation techniques offered by Weka.[11] These evaluation techniques are used to search for the best feature combination for the **Mix** subset since it offers different types of false facts. We used the ClueWeb corpus and the J48 and Simple Logistics classifiers owing to their performance on the selected set. The results of the experiment show that J48 and Simple Logistics achieve F-measures of up to **0.898** and **0.892**, respectively, when an optimized set of features is used. In addition to search-related features (PageRank, number of hits), both classifiers used the topic-related features and the features based on dependency parsing presented in Section 3. The features *domain & range verification* and *statistical triple evidence* are not used by any of the classifier. This is probably an artifact of the use of FactBenct, since the creators of the dataset made sure that the domain and range of the properties are not violated.

[11] We used the evaluators *WrapperSubsetEval*, *ClassifierSubsetEval*, *FilteredSubsetEval*, *CfsSubsetEval* and *ConsistencySubsetEval*.

6 CONCLUSION AND FUTURE WORK

In this paper, we presented FactCheck—a fact validation approach based on machine learning and textual evidences gathered from a reference corpus. We showed that FactCheck outperforms the state of the art by up to 13.3% F1-measure and 19.3% AUC. Additionally, we analyzed the influence of the reference corpus size and the importance of the features used for the classification of facts. In the future, we want to use this advantage to create fact checking web services to enable access FactChecks functionality directly and to encourage other researchers to compare their own approaches with FactCheck easily. Another interesting future work is to extend FactCheck with more features and apply an ensemble learning instead of a single classifier.

ACKNOWLEDGEMENTS

This work has been supported by the BMVI projects LIMBO (project no. 19F2029I) and OPAL (project no. 19F2028A).

REFERENCES

[1] Nikolaos Aletras and Mark Stevenson. 2013. Evaluating topic coherence using distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)–Long Papers*. 13–22.

[2] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 601–610.

[3] Eibe Frank, Mark A. Hall, , and Ian H. Witten. 2016. The WEKA Workbench. In *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

[4] Daniel Gerber, Diego Esteves, Jens Lehmann, Lorenz Bühmann, Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, and René Speck. 2015. Defacto–temporal and multilingual deep fact validation. *Web Semantics: Science, Services and Agents on the World Wide Web* 35 (2015), 85–101.

[5] Daniel Gerber and Axel-Cyrille Ngonga Ngomo. 2011. Bootstrapping the Linked Data Web. In *1st Workshop on Web Scale Knowledge Extraction @ ISWC 2011*.

[6] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6, 2 (2015), 167–195.

[7] Satoshi Nakamura, Shinji Konishi, Adam Jatowt, Hiroaki Ohshima, Hiroyuki Kondo, Taro Tezuka, Satoshi Oyama, and Katsumi Tanaka. 2007. Trustworthiness analysis of web search results. *Research and advanced technology for digital libraries* (2007), 38–49.

[8] Axel-Cyrille Ngonga Ngomo, Michael Hoffmann, Ricardo Usbeck, and Kunal Jha. 2017. Holistic and Scalable Ranking of RDF Data. In *2017 IEEE International Conference on Big Data*. 10. https://svn.aksw.org/papers/2017/ESWC_HARE/public.pdf

[9] Jeff Pasternack and Dan Roth. 2010. Knowing What to Believe (when You Already Know Something). In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 877–885. http://dl.acm.org/citation.cfm?id=1873781.1873880

[10] Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*. ACM, 399–408.

[11] Sebastian Schuster and Christopher D Manning. 2016. Enhanced English Universal Dependencies: An Improved Representation for Natural Language Understanding Tasks.. In *LREC*.

[12] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 697–706.