ISSN 2007-9737

Probabilistic Error Detection Model for Knowledge Graph Refinement

Manuela Nayantara Jeyaraj, Srinath Perera,
Malith Jayasinghe, Nadheesh Jihan

WSO2, Mountain View,
United States

manuela.n.jeyaraj@mytudublin.ie

Abstract. Knowledge graphs are widely used in information queries. They are built using triples from knowledge bases, which are extracted with varying accuracy levels. Accuracy plays a key role in a knowledge graph, and knowledge graph construction uses several techniques to refine and remove any inaccurate triples. There are many algorithms that have been employed to refine triples while constructing knowledge graphs. These techniques use the information about triples and their connections to identify erroneous triples. However, these techniques lack in effective correspondence to human evaluations. Hence, this paper proposes a machine learning approach to identify inaccurate triples that correspond to actual human evaluations by injecting supervision through a subset of crowd-sourced human evaluation of triples. Our model uses the probabilistic soft logic's soft truth values and an empirical feature, the fact strength, that we derived based on the triples. We evaluated the model using the NELL and YAGO datasets and observed an improvement of 12.56% and 5.39% in their respective precision. In addition, we achieved an average improvement of 4.44% with the F1 scores, representing a better prediction accuracy. The inclusion of the fact strength augmented the modeling precision by an average of 2.13% and provided a higher calibration. Hence, the primary contribution of this paper is the proposal of a model that effectively identifies erroneous triples, aligning with high correspondence to actual human judgment.

Keywords. Information extraction, knowledge graph, machine learning, probabilistic soft logic.

1 Introduction

A knowledge extraction pipeline takes in data, converts it to a knowledge base, and finally provides the outcome of knowledge extraction as a Knowledge Graph. A single link or an edge in a knowledge graph is the relationship that connects a subject to its object.

The subject, object and the relationship together are known as the triple. Knowledge base is a collection of triples while knowledge graph adds missing connections and confidence measurements to those connections. We extract triples from various sources such as free text, database and knowledge bases, using NLP techniques such as part-of-speech tagging, tokenizing, stemming, and so on.

These extracted triples have different levels of accuracy. If inaccurate information is incorporated into the knowledge graph, queries based on that graph can return erroneous responses. Hence, this is a current concern with regard to knowledge graphs.

In order to completely identify the accuracy of knowledge graphs, the most trivial method is to perform a complete manual check on all the facts used for the graph. Yet this is rather expensive and exhaustive. Hence, most of the existing solutions resolve to automated methods and pre-process the knowledge bases for erroneous triples [23].

These techniques known in literature, measure the accuracy of triples, considering their neighboring triples. They can be based on heuristics, building vector space models and computing word scores based on the tf-idf weights of vectors [20], or averaging the ontology coverage based on the frequency classes [29] (further explained in section 6). However, these automated methods have not satisfactorily addressed the

doi: 10.13053/CyS-26-3-4346

inaccurate facts and the lack of correspondence to actual human evaluations.

Therefore, this paper leverages the advantages of automated methods and the correspondence to human judgment from manual methods, to propose a semi-approach for evaluating the accuracy of a large set of triples based on the human evaluation of a subset of triples.

We use machine learning to verify the correctness of the triples based on a set of features; subject, object, predicate and the probabilistic soft truth confidence values. We further improve our technique based on a novel, empirical feature, which we term as the “fact strength”. We use human evaluated data as the target variable for training and use the model to gauge the accuracy of triples.

Accordingly, as the primary contribution of this paper, we identify machine learning as a suitable candidate for further refining the knowledge bases or knowledge graphs based on a partially evaluated dataset. We propose classification models to identify the erroneous triples that correspond to actual human evaluation. We evaluated the models using the Never-Ending-Language-Learner (NELL)[1] and YAGO[2] datasets, and observed a 12.56% and 5.39% of improvement in the precision, respectively.

In addition, we achieved an average improvement of 4.44% in the F1 scores, representing a better prediction accuracy. Introducing the fact strength as a feature, provided an average positive augmentation of 2.13% in the precision to achieve the above improvement. Thus, this model addresses the use-case of effectively removing erroneous triples from knowledge graphs.

This paper is outlined with further details on the background, proposed solution, evaluation, discussion of results and the related work, with regard to addressing our contribution.

2 Background

2.1 Knowledge Graphs

Since the age of the internet, the retrieval and storage of information has become a vital part of all activities that are being conducted both online as well as offline. As such, a massive amount of data is being generated by seconds. According to Forbes, 2.5 quintillion bytes of data are being generated every single day[3]. In the past, databases were considered a sufficient store for information [18].

Further, as the amount of content grew exponentially, data warehouses came into context [13]. Consequently, the multivariate nature of content being produced, called forth the need for more sophisticated methods to retrieve data from these content and create a more generalized storage unit, ergo the concept of knowledge bases emerged [11]. Knowledge bases are built on an ontology based storage of information or so-called ‘facts’ [10] and consists of 2 major components : the interface engine and the knowledge repository. The interface engine serves as a search engine to browse for information stored in the repository.

Searching for facts is enabled through classified ontologies. Based on the sense in which ontologies are used, their definition varies [15]. Considering the Knowledge Engineering domain, we adhere to Gruber’s definition of an ontology [14] : A representational identification of a vocabulary, for a particular domain. As the knowledge base learns its facts from various sources, it classifies the learned facts under ontologies.

Such knowledge bases that have been of primal use to mankind are explicated here. (1) Freebase [4], contained 1.9 billion triples or learned-facts[4], before being deprecated, as Meta-web, the developer of Freebase, sold Freebase to Google in 2010. It harnessed its data from the semantic web [3] and Wikipedia articles. (2) DBpedia [2] extracts information from Wikipedia and builds structured

[1] The NELL knowledge base : http://rtw.ml.cmu.edu/rtw/

[2] The YAGO dataset: https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/

[3] Amount of data generated on a daily basis: https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/\#6ef79f2760ba\

[4] Data Dumps of Free base : https://developers.google.com/freebase/

facts that can be queried upon. (3) YAGO [30] does not directly scour Wikipedia articles.

Alternatively, it leverages the category pages available in Wikipedia[5]. YAGO represents the learned facts in the form of the Web Ontology Language (OWL) [24]. OWL uses the Resource Description Framework Schema (RDFS) [1], which only describes the relationship between facts using unrefined semantics. Hence, YAGO was developed as a refinement of RDFS. (4) The Never Ending Language Learner (NELL), which was developed at the Carnegie Mellon University, adopts the key difference between human-learning and machine-learning, continuously learning facts since 2010. Due to this learning process, NELL accumulates both correct as well as incorrect facts/beliefs.

An example of an accumulated fact in NELL is: *”Astoria is a city that lies on the river Columbia (river)”*. In this case, “*Astoria*” is the subject that relates to the object, i.e., “*Columbia*”, through the predicate or relationship, “*lies on*”. Hence, this relationship connects 2 entities: the subject and object, to form a triple, which is a fact learned within the knowledge base. Currently NELL has accrued 50 million such candidate beliefs.

Notwithstanding, knowledge bases lack in the sense of inter-connectivity between entities. Though a single fact’s entities: the subject and object, are connected, the way in which the rest of the entities connect with one another cannot be directly observed in plain, flat knowledge bases, consequently giving rise to Knowledge Graphs.

“Knowledge Graph” was a term coined by Google as it introduced Google’s Knowledge Graph in 2012 [9]. Knowledge graphs are built of triples where, the relationships or predicates form the edges between various entities, assembling a massive network of interconnected entities thus, providing more context into how different entities interact and maintain a relationship. The edges, which are the relationships in knowledge graphs, are inferred based on statistical relational learning (SRL). Probabilistic models in SRL are used to compute confidences to justify how far these inferred relationship edges hold in the graph. One such probabilistic model, that we use in our solution, is the Probabilistic Soft Logic (PSL) [7].

2.2 Probabilistic Soft Logic

The probabilistic soft logic is a statistical relational learning framework that infers a soft truth value which serves as a confidence for each fact based on the joint probabilistic reasoning in its relational domain [17].

It uses first order logic and a weight learning of rules as it projects the most probable explanation for inferences in the form of a convex optimization [6] and estimates a soft truth value which ranges between [0,1] as opposed to a restricted 0 or 1. PSL is best resorted to, in computing how far a fact or belief holds. It defines a set of general rules such as transitive, commutative, associative, etc. PSL rules are of the following form:

$$w : R_1(A,B) \wedge R_2(B,C) \rightarrow R_3(A,C), \qquad (1)$$

where *R₁, R₂ and R₃* are the relationships, *A, B* and *C* are entities, and *w* is the weight of the rule. As we apply constants or real-world entities from the facts, onto the rule, they become ground rules. And this process is appropriately known as *”grounding”* as shown in (2):

$$0.9 : livesIn(Claire, Paris) \wedge spouse(Claire, Blake) \\ \rightarrow livesIn(Blake, Paris). \qquad (2)$$

This states that if *Clair* lives in *Paris* and *Clair* is the spouse of *Blake*, then it implies that *Blake* could also be living in *Paris*. This rule holds with a weight of 0.9 with the belief that spouses are more likely to live in the same place. Here, *livesIn(Claire, Paris)*, *spouse (Claire, Blake)* and *livesIn(Blake, Paris)* are considered as atoms, *x*.

In the Knowledge Engineering community, these atoms can be addressed as triples as well. Here, the implied atom, which is on the right side, becomes the head of the relationship and the ones on the left are the body. Some of these triples can be known triples, with previously observed soft

[5] Information regarding the detailing of a category, time-stamps and sort keys can be fetched by querying in the api

doi: 10.13053/CyS-26-3-4346

truth values and the others may be unknown triples whose soft truth values are previously unknown:

$$\alpha_x = \tau, x \in \mathbb{S}^+,$$
$$\alpha_x = p(x_e), x \in \mathbb{S}^-. \tag{3}$$

where α_x is the soft truth value of the atom *x*, $\mathbb{S}^+$ is the set of known triples, $\mathbb{S}^-$ is the set of unknown triples, τ is the confidence of the known triple *x*, and *p(x$_e$)* is the conditional probability of atom *x* for its embedded soft truth values.

When 2 entities *e$_1$* and *e$_2$* are in a relationship *r*, as an atom *x*, and their soft truth value is previously unobserved, then we derive their initial soft truth value using *p(x$_e$)* which is the conditional probability represented by *p(e$_1$—e$_2$)* as shown in (4):

$$p(e_1|e_2) = \frac{count_{facts}(e_1, e_2)}{\Sigma_{r\in R}\, count_{facts}(r, e_2)}. \tag{4}$$

Here, *R* is the set of all the relationships in the domain. This is how we derive the soft truth values for the previously unknown atoms/facts of a ground rule [22]. Once, the soft truth values for the atoms are inferred, the logical connectives: $\wedge$, $\vee$ and $\neg$ of ground rules need to be relaxed using a normalization by Lukasiewicz t-norms [7]:

$$a \overset{\sim}{\wedge} b = max\{0, a+b-1\},$$
$$a \overset{\sim}{\vee} b = min\{1, a+b\}, \tag{5}$$
$$\overset{\sim}{\neg} a = 1-a.$$

This normalization is performed to enumerate an aggregated soft truth value for the entire body of the atom. With a soft truth value for the body, *r$_{body}$*, deduced using the Lukasiewicz normalization in (5), and a soft truth value for the head, *r$_{head}$*, obtained using (4), PSL models a statistical relationship based inference using the following concept.

If r is a rule in PSL such that $r_{body} \rightarrow r_{head}$ and given an inference *I* that is grounded to *r*, *r* is satisfied only if $I(r_{body}) \leq I(r_{head})$. If the rule is satisfied, the distance to satisfaction (*d*), will be 0. On the other hand, if it fails to be satisfied, the degree to which it deviates from satisfaction will be captured by the distance to satisfaction for that Inference, $d(I)$, as shown in (6):

$$d(I) = \; max\{0, \; I(r_{body}) - I(r_{head})\}. \tag{6}$$

Furthermore, given the set of ground atoms, a distribution will be built as the probability density function, $f(I)$ [17]:

$$f(I) = \frac{1}{Z} exp[-\sum_{r\in R} \lambda_r (d(I))^p];$$
$$Z = \int_I exp[-\sum_{r\in R} \lambda_r (d(I))^p]. \tag{7}$$

In (7), R is the finite set of all defined rules, Z is the discrete Markov random field normalization constant for a continuous value, λ_r is the assigned weight for the rule r, $d(I)$ is the distance to satisfaction for the inference I, and p indicates the loss function with either ‘1’ which supports interpretations that fully satisfy a single rule, willing to suffer a higher distance to satisfaction for the other rules or ‘2’ which is a quadratic loss function that attempts to support all the rules to some extent.

Here, the optimal distribution will be the conclusively inferred, soft truth value of the implied fact, i.e, the head atom, based on the statistical relationship between entities. As such, all the facts are grounded as ground rules such that they become the implied relationship or head atom in the rule, and the optimal distribution's probability will be rendered as that fact's soft truth value or confidence from each distribution generated based on prior computations.

These soft truth values are mere indications of how far the system is confident in its relational inference. But we do not have the means to clearly classify a fact as true or false according to human judgment by solely using these confidences. Hence, PSL constructs knowledge graphs with an accumulation of all these facts, along with their inferred confidence scores. This can build a noisy graph with inaccurately inferred facts.

Since there have been a substantial amount of work conducted with regard to missing data (explained further in the Related Work), in this paper, we attempt to address the removal of false data from massive knowledge graphs using

PSL and a small subset of human evaluated fact truths. The core difference between previous models that relate to error detection in knowledge graphs and our model is that, the validity of the fact is modeled with correspondence to actual human judgment, instead of solely relying on system generated confidences. And we discover a pattern to restrictively label facts as true or false, using a machine learning approach. The next section explains our solution model that was developed to achieve the above same.

3 A Model to Address Erroneous Triples in Knowledge Graphs

In order to identify the erroneous triples, we propose a supervised machine learning approach based on classification techniques to predict the accuracy of a triple. Our initial experiments were based on a basic features-set; the triples' subject, predicate, object, PSL's soft truth values and the human evaluations.

Since the subject, predicate and object are words, we encode to obtain their tf-idf as the feature to the models. However, the PSL soft truth value is directly taken as a feature to incorporate the confidence of each fact.

Later we extend our models by adding the fact strength as a feature. As the dependent variable to the classifiers, we use the human-evaluated score for each fact during the training. For supervision, we need this human evaluation.

Buhrmester et al. have shown the use of crowd-sourcing to generate human evaluated task sets for assessing the performance of various knowledge graph identification tasks [8]. Hence, we propose using crowd-sourcing to evaluate a randomly selected subset of triples from the complete knowledge base, using those triples for training the models.

Table 1. Sample dataset of the triple fields

Fact id	ontology : subject	predicate	ontology : object
1	person:leonardHofstader	livesIn	place:california
2	person:sheldonCooper	isFriendOf	person:leonardHofstader
3	person:sheldonCooper	isSpouseOf	person:amyFarrahFowler
4	person:sheldonCooper	worksWith	person:leonardHofstader
5	person:amyFarrahFowler	livesIn	place:california
6	person:leonardHofstader	isFriendOf	person:AmyFarrahFowler

Moreover, we extend our solution model to introduce a novel empirical feature to quantify the importance of each triple. This feature was a derivative of the number of relationships or interaction between the subject-predicate-object triple. We term this as the ***fact strength***, φ, for referential purposes. Equation (8) shows how we compute φ:

$$\varphi = \frac{(\eta_{spd} \times \eta_{sd}) + (\eta_{opd} \times \eta_{od})}{\eta_{so}}. \tag{8}$$

For each fact or belief *b* in the dataset, such that, $b \in \mathbb{B}$, where $\mathbb{B}$ is the finite set of all beliefs within the domain, we compute η_x. The following explicates the variations of η_x used in equation (8):

η_{spd} → Number of times the `ontology of the subject` and the `predicate` appear in a fact within the dataset.
η_{sd} → Number of times the `subject` and the `predicate` appear in a fact within the dataset.
η_{opd} → Number of times the `ontology of the object` and the `predicate` appear in a fact within the dataset.
η_{od} → Number of times the `object` and the `predicate` appear in a fact within the dataset.
η_{so} → Number of times the `subject` and the `object` appear in a fact within the dataset.

For example, consider we have a dataset with 6 facts as shown in Table 1. However, in reality there are millions of facts.

According to the sample dataset in Table 1:
η_{spd} = 2, since the subject's ontology, *person*, and the predicate, `livesIn`, occur twice throughout the dataset.
η_{sd} = 1, since the subject, *leonardHofstader* and predicate, *livesIn*, occurs only once throughout the dataset.
η_{opd} = 2, since the object's ontology, `place`, and the predicate, *livesIn*, occur twice throughout the dataset.
η_{od} = 2, since the object, *california*, and the predicate, *livesIn*, occur twice throughout the dataset.
η_{so} = 2, since the subject, *leonardHofstader*,

and the object, *california*, occur twice throughout the dataset.

Thus, the fact strength will be computed as:

$$\varphi = \frac{(2 \times 1) + (2 \times 2)}{2}, \quad \varphi = 3. \tag{9}$$

Hence, the fact strength for the first fact in the dataset will be 3. This is an abstract example of how the derived, empirical feature φ is calculated for each and every fact. Nevertheless, in large datasets such as the ones we have used to train and test, the fact strength can range to a greater value such as 200 and above. However, the fact strength will always be a positive value as $\varphi \geq 1$.

We extend our models to adopt the fact strength (computed using the complete knowledge base) as a feature apart from the initial set of features.

During the preparation of the dataset, we computed the PSL-stv and the fact strength for the dataset, keeping the crowd-sourced predictions as the target variable. The inclusion of the fact strength showed an improvement in the precision, F1 and recall of the predictions, varying based on the type of classifier used.

4 Evaluation

For the training process, we used the NELL dataset which is a universal standard when it comes to knowledge base and knowledge graph experiments. The main intent of the NELL dataset is to expand its knowledge base by iteratively learning facts. It uses the facts that it had learned through previous experience to generate newly learned candidate beliefs/facts. Also, our selection of NELL was based on the consideration that NELL has been constantly learning facts and, therefore is up-to-date on the facts that it had learned, without being deprecated. With any other dataset, the time-value of information will have to be considered for the temporal validity of information. NELL also accumulates its facts from a wide array of sources, with confidences based on the sources' reliability as understood from previously learned facts. We extracted a subset of NELL that consists of around 2000 triples, along with their crowd-sourced evaluations.

Another dataset that we opted for evaluation purposes, is the YAGO dataset which is often used as the test set along with the NELL test dataset. YAGO fetches its facts from Wikipedia as well as WordNet. The classified labeling of the entities and the properties of YAGO facts, enables the easy access to its ontologies. In addition to that, scouring Wikipedia articles through their category pages allows YAGO to incorporate an extensive knowledge of a domain, based on its sub domains. Being triple based, YAGO bears reference to the time and location of the source from which its facts were derived. Hence, we adhere to the NELL and YAGO datasets based on their aforementioned advantageous features. As such, we extract around 1400 triples from the YAGO dataset for our experiments.

The crowd-sourced evaluations for both the datasets were obtained using the Amazon Mechanical Turk[6]. Consequently, the accumulated crowd-sourced outcomes are rendered in the form of 1 and 0 for each fact, respectively indicating a fact being true or false according to human evaluators. These results were more sought after as the evaluators were of diverse demographics [8].

As performed in [12] and [19], we used the 70:30 split on the NELL dataset to proceed with the training.

Since this prediction is a classification problem, we chose the Support Vector Classifier, Stochastic Gradient Descent and Random Forest Classifier to identify the optimal classifier.

We use the rbf kernel as a default for the classifiers as it is a stable kernel that is invariable to translations. In the case of evaluating $M(a, b)$, the rbf kernel will compute the same value, that is, $M(a, b)$ for a translation such as $M(a + x, b + x)$, where x is a constant vector for the dimension, such that it aligns with the inputs. The rbf kernel achieves this invariability by observing the difference between the computed vectors. Further, we maintain the default hyper parameters across all the classifiers.

[6] Amazon Mechanical Turk: https://www.mturk.com/

4.1 Experiments

Initially we investigate the effect of the basic features (tf-idf of subject, predicate and object, PSL soft truth values, and the human evaluations) to identify the erroneous facts with the aforementioned classifiers. The evaluation was performed on both the NELL and YAGO datasets. We have presented these observations in the first part of this section.

4.1.1 Experiment 1

After training the model on the training dataset from NELL, we evaluated the test dataset of NELL using the trained model. These results are depicted in Figure 1.

The Baseline here, is the performance index values for the triples, based on solely their soft truth values. For example, if we assume that all the triples with a soft truth value less than or equal to 0.1 are false, and all those with a soft truth value greater than 0.1 are true, and evaluate our prediction accuracy against actual human evaluations, then the performance index values obtained for the precision, recall and f1 scores will be considered the baseline for the threshold value of 0.1.

According to [26], we chose the threshold of $\theta \geq 0$ for the PSL soft truth values θ, based on the consideration that this is the threshold that maximized the F1 score for our PSL evaluations. Hence, our baseline for the precision is 0.828, recall is 0.910 and f1 is 0.867.Comparing against this baseline, we evaluated to see if the classifiers showed a proven improvement.

Please note that the performance index values displayed in the table in Figure 1 are rounded off to 3 decimal points, while the graph adheres to the original values.

According to Figure 1, all the classifiers show an improvement in the precision, whereas the Random Forest Classifier obtains the optimal precision of 0.927, as opposed to the baseline of 0.828. This indicates a 11.96% of improvement in the precision index for the classifier. However, the test accuracy was evaluated in terms of the improvement achieved for the F1 score. From a baseline of 0.867 to an augmented 0.920, the Random Forest Classifier displays a 6.11% improvement in the prediction accuracy for the NELL dataset. However, the recall drops for the support vector classifier and the stochastic gradient descent compared to the baseline. The Random Forest Classifier achieves slightly better recall and significantly outperforms the baseline in terms of the F1 score.

4.1.2 Experiment 2

We applied the trained models on the YAGO dataset. The performance index values rendered for this experiment are shown in Figure 2.

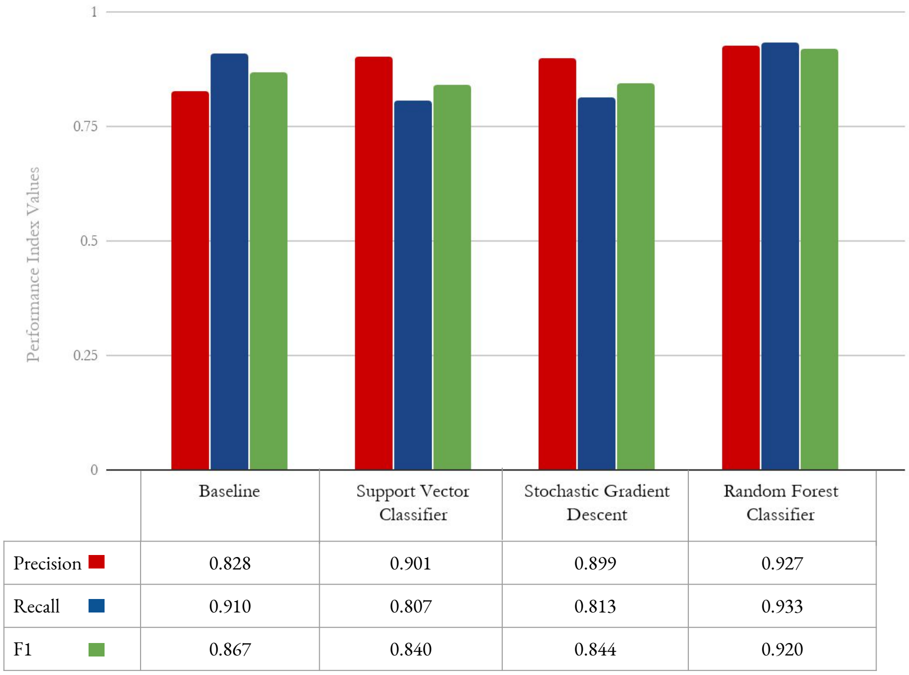
According to this, the Stochastic Gradient Descent holds a slightly higher precision of 0.945, compared to the the random forest classifier that showed a precision of 0.943. Hence, the stochastic gradient descent and the random forest classifier showed respective precision improvements of 1.94% and 1.73%.

Here, we achieve an F1 score of 0.957 from a baseline of 0.944, with the random forest classifier which renders a 1.38% improvement in the test accuracy, in terms of the F1 score.

Considering the above two experiments, with the sole feature-set of the triples (subject, predicate and object), the PSL confidences, and the human evaluations, we were able to achieve an average precision improvement of 6.85%. Consequently, we were able to arrive at an observation that the random forest classifier displayed an enhanced test accuracy with an average improvement of 3.75%.

During these experiments, the Random Forest Classifier performed with the optimal precision and descent recall values.

Furthermore, we extend the experiments to assess the accuracy gain of the models with the feature that we introduce, the fact strength. The following part of this section will illustrate the observed performance of the models after incorporating the fact strength in assessing erroneous triples.



	Baseline	Support Vector Classifier	Stochastic Gradient Descent	Random Forest Classifier
Precision 🟥	0.828	0.901	0.899	0.927
Recall 🟦	0.910	0.807	0.813	0.933
F1 🟩	0.867	0.840	0.844	0.920

Fig. 1. Performance Index Values for the NELL dataset(experiment 1)

4.1.3 Experiment 3

Here, we added the derived feature, that is, the fact strength, during the training process and performed the above same for the NELL dataset, now with an extended feature-set (subject, predicate, object, PSL soft truth values, fact strength and the human evaluations). This is shown in Figure 3.

Compared to the results obtained in Figure 1, the new models, shown in Figure 3, perform better, with the Random Forest Classifier achieving a precision of 0.932. Comparing this against our baseline showed that the precision had increased by 12.56%.

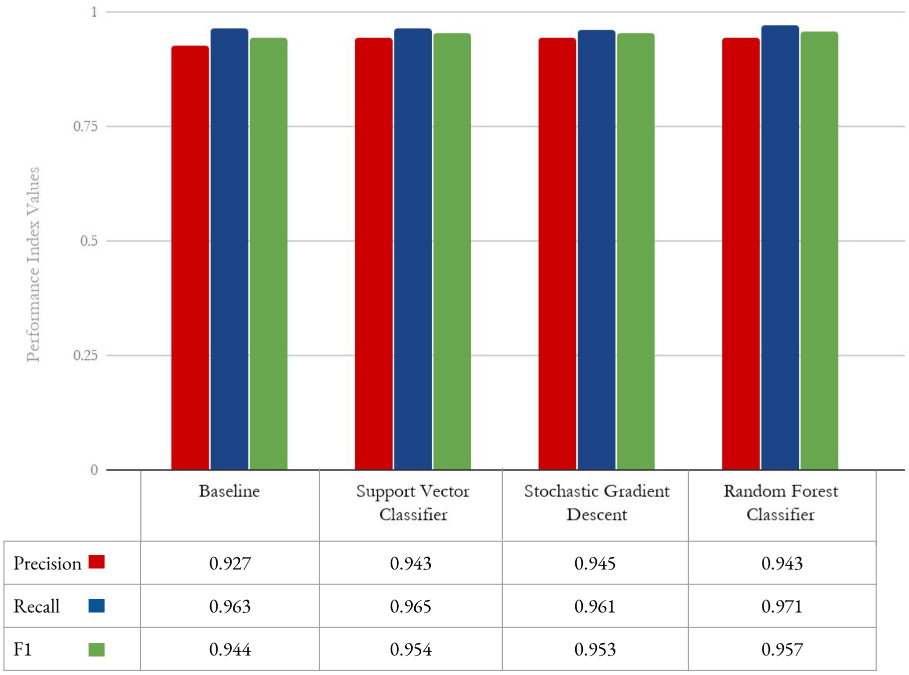
This indicates that the derived fact strength positively influences the classification based on the precision and provides a better recall of 0.936 as well.

Subsequently, we analyzed the F1 score for the datasets in order to garner a complete picture of the classifiers' accuracy. As such, with the introduction of the fact strength, the random forest classifier produced an F1 score of 0.922 from its baseline of 0.867, giving a test accuracy improvement of 6.34%.

Solely considering the F1 score to verify the impact of the fact strength, we observed only a trivial raise of 0.23%.

However, viewed together with the enhancements in the precision and recall, we can safely derive that the random forest classifier displays a better test accuracy on the whole. Furthermore, we observed a significant drop in the stochastic gradient descent, where its F1 score plummeted to 0.300.

With the precision at a recognizable 0.908, the recall of 0.256 suggested that the stochastic gradient descent was biased towards the false negatives and misinterprets more of the the false facts. Hence, as a result, the F1 depreciated as well.



	Baseline	Support Vector Classifier	Stochastic Gradient Descent	Random Forest Classifier
Precision 🟥	0.927	0.943	0.945	0.943
Recall 🟦	0.963	0.965	0.961	0.971
F1 🟩	0.944	0.954	0.953	0.957

Fig. 2. Performance Index Values for the YAGO dataset (experiment 2)

4.1.4 Experiment 4

We conducted the above same experiment on the YAGO dataset with the extended set of features, including the computed fact strength, and the results are depicted in Figure 4.

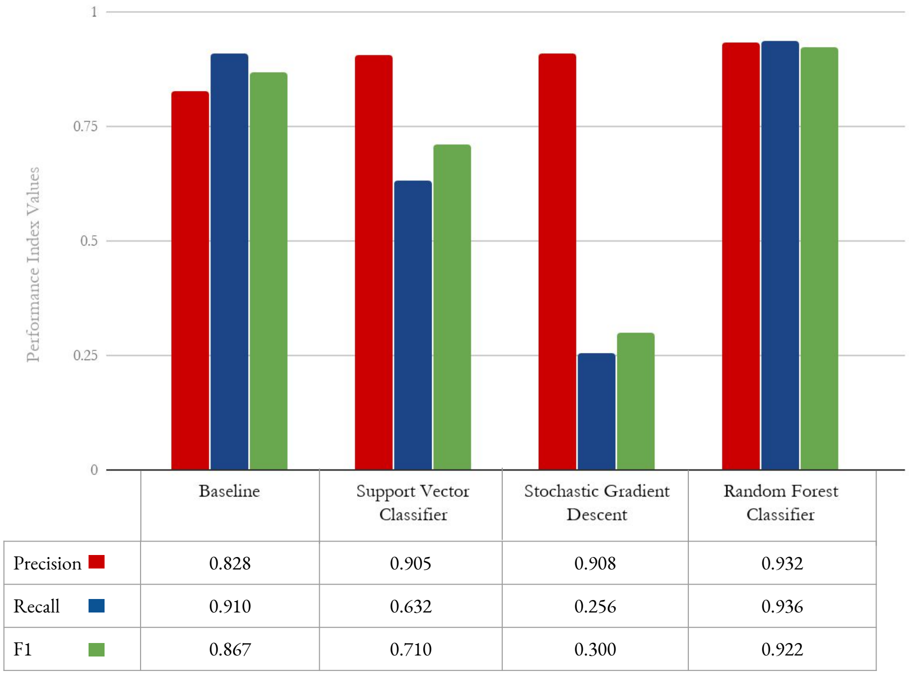
Experimenting the same on the YAGO dataset had a much better response to the fact strength feature as it augmented the precision by 3.61%. Also, the trivial improvement in the recall suggested that the true positive rate had increased. With most of the facts in this dataset being evaluated as true facts through human judgment, we were able to obtain an improved high F1 score of 0.968 as opposed to the baseline 0.944, through the Random forest classifier. All the classifiers gain an improvement over the precision baseline of 0.927 and F1 baseline of 0.944. This illustrated a test accuracy improvement of 2.54% for the random forest classifier.

Hence, with the introduction of the fact strength, we were able to garner an average prediction accuracy improvement of 4.44% for the random forest classifier. This test also proved that the random forest classifier produces a high precision and recall, deriving a reasonable F1 score as identified in the previous experiments as well.

4.2 Classifier Calibration

In addition to the above performance index metrics, we proceeded to evaluate the calibration of the classifiers[7]. The classifier calibration models the classifiers' alignment as opposed to the best or ideal calibration for the actual predictions. The predicted truth value against the fraction of positives, plots the graph in the form of $y = mx + c$. As such, Figure 5 was rendered and is shown here.

[7] Calibration of Classifiers: https://scikit-learn.org/stable/modules/calibration.html



	Baseline	Support Vector Classifier	Stochastic Gradient Descent	Random Forest Classifier
Precision 🟥	0.828	0.905	0.908	0.932
Recall 🟦	0.910	0.632	0.256	0.936
F1 🟩	0.867	0.710	0.300	0.922

Fig. 3. Performance Index Values for the NELL dataset (experiment 3)

5 Discussion

Initially we present the discussion of the results on the NELL and YAGO datasets with only the subject, predicate, object, PSL soft truth values and the human evaluations, excluding the fact strength.

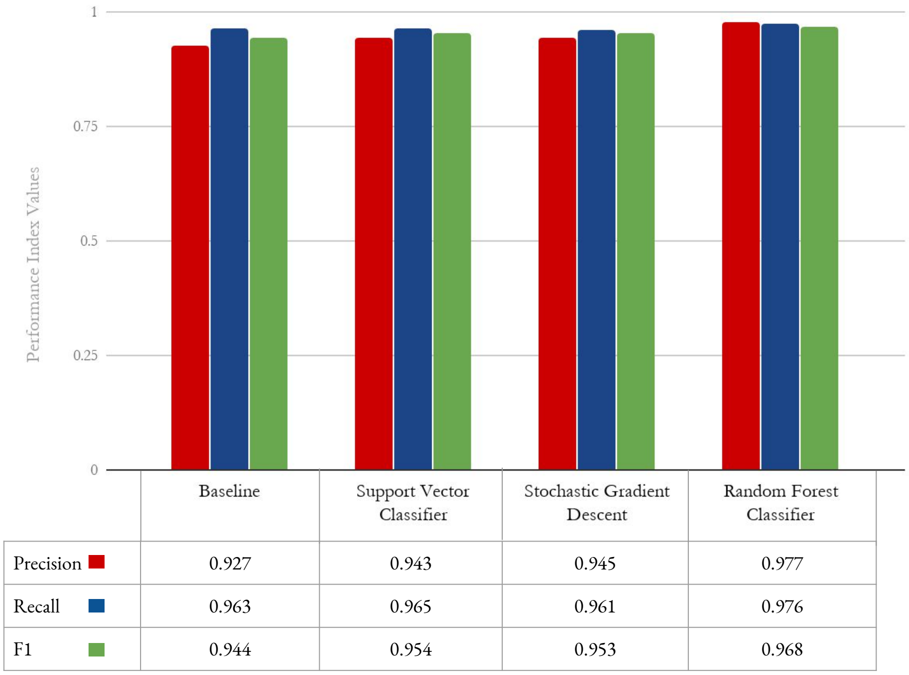
Based on the experiments, we support the claim that, our approach can be generalized to different types of dataset domains, by applying the NELL trained dataset on the YAGO dataset and still achieving remarkable Precision and F1 score improvements as proved in the second and fourth experiments.

Further, our evaluation proves that the addition of the empirical fact strength feature improves the models' performance index values as shown in the third and fourth experiments. Hence, we identify the fact strength as an enhanced feature in modelling the strength or weight of a fact in relation to its entire domain.

Reverting back to our contribution, adding a level of supervision in the form of actual human evaluations, increases the prediction accuracy in identifying the inaccurate triples as proven through the average Precision and F1 scores as evidenced in these experiments.

Secondly, the addition of a small set of human evaluated triples, to evaluate a larger unseen dataset performs effectively, especially for the random forest classifier. This claim of ours is supplemented throughout the experiment, with observed improvement in the precision, recall, and F1 scores for all the experiments, when compared against their respective baselines, where actual human evaluation based supervision is not included.

We were able to accompany our claims for all the experiments, with the Random forest classifier. Though the Stochastic Gradient Descent showed occasionally reasonable performance index values, they fluctuate throughout the



	Baseline	Support Vector Classifier	Stochastic Gradient Descent	Random Forest Classifier
Precision 🟥	0.927	0.943	0.945	0.977
Recall 🟦	0.963	0.965	0.961	0.976
F1 🟩	0.944	0.954	0.953	0.968

Fig. 4. Performance Index Values for the YAGO dataset (experiment 4)

experiments. Hence, we resorted to the random forest classifier that constantly illustrates the expected behaviour, with high performance index values and average improvements.

Solely based on these experiment results, we saw the best performance of the random forest classifier.

However, in order to verify its performance, in terms of calibration, we analyzed the classifier calibration as shown in Figure 5. Here, it is clearly observable that the Random Forest Classifier (RFC) has the optimal calibration with its actual prediction, as it lies as close to the ideal, perfectly calibrated $y = mx$ line, while the calibration for the Stochastic Gradient Descent (SGD) and Support Vector Classifier (SVC) deviate further away from the perfect calibration.

Hence, based on the performance index values (precision, recall and f1), and the classifier calibration, we safely arrive at the conclusion that the Random Forest Classifier, trained on a part of the NELL dataset, with a feature set of triples (subject, predicate, object), computed PSL soft truth values, derived fact strength, and crowd-sourced human evaluations, can be used to effectively predict the most probable human evaluation/truths for any larger datasets of triples.

Using these evaluations, we then propose on dropping the facts or triples, predicted as false by the classifier and building the knowledge graph with the facts/triples predicted to be true. This will be an effective pre-process in removing erroneous data from knowledge graphs.

Hence, the knowledge graph will be more pure and accurate in the sense of holding true facts that correspond to actual human judgment. The application of this model also proposes use cases in information extraction systems, where data can be represented in the form of triples.

A constraint that we identified with this model is the specificity of the feature set, in the form of triples, in order to compute the PSL soft truth

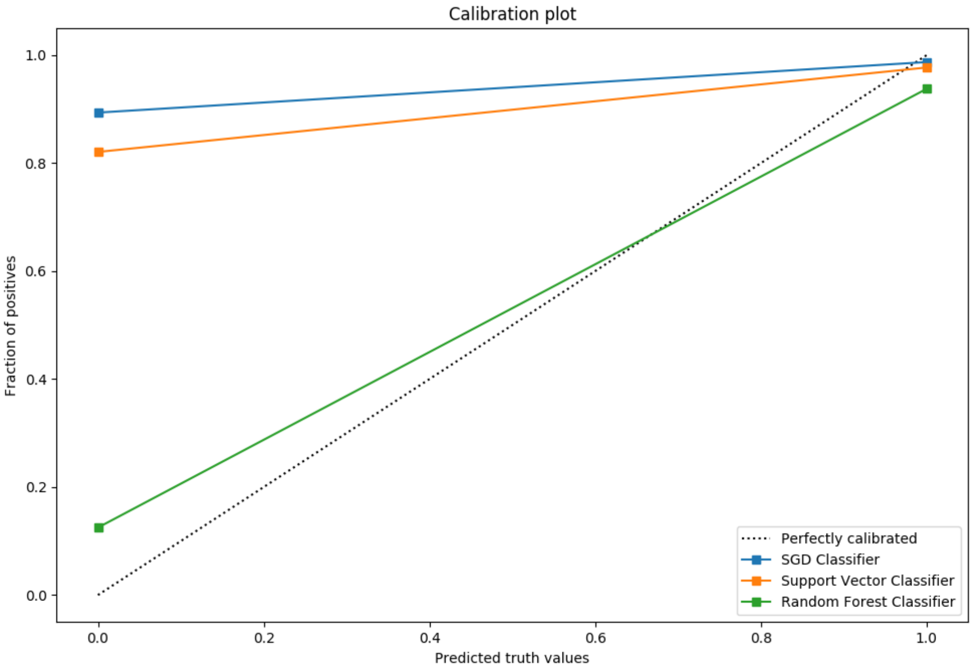


Fig. 5. Classifier Calibration Plot

values. Thus, our future research will be directed towards the generalization of this solution model such that it is applicable across omni-various fact/data formats.

6 Related Work

There are various forms of knowledge stores or representation methods as described in the Background section. We were able to observe that the evolution of these methods correlated to the amount and complexity of the data being generated with time.

Since we identify knowledge graphs as the optimal knowledge representation method, we looked into the constraints in knowledge graphs. As such, there are 2 major issues that need to be addressed [25]. The first concerns data completion, where certain information may be missing from knowledge graphs and may not be available when queried for. The second issue is the presence of erroneous data which will lead to the return of incorrect responses, or responses that do not correlate with human judgment, when the user queries the knowledge graph.

There has been extensive research with regard to data completion in knowledge graph whereas the removal of erroneous data is still an area with on-going research prospects. Previous work by Lin et al. [21] has discussed about handling knowledge graph completion by learning entities and relation embeddings.

This was set as an extension of the TransE [5] and TransH [31] models that compute entity and relational embeddings as a translation from one entity to the other in a relationship. This model was called TransR.

In the TransE and TransH models, vector embeddings are learned and encrusted within the same space implying that the entities and relationships are set in the same vector space $\mathbb{R}^k$. But entities have variations, and relations model for the variations of the entities.

Hence, the TransR model proposed by Lin et al. identifies separate vector spaces for the entities and relations for each triple (h, r, t). The entities will be embedded as $h, t \in \mathbb{R}^k$ and the relation will be embedded as $r \in \mathbb{R}^d$.

Another method in knowledge graph completion includes the Adaptive Sparse Transfer Matrix where each entity and relation is encoded numerically and triplet classification and link prediction tasks are performed to complete the graph [16]. Extending from the previous work on the TransE, TransH and TransR models, Ji et al. identified the heterogeneity of and imbalance in the entities and relations in knowledge graphs. Their SparseTrans(share) model resolves to sparse transfer matrices in place of transfer matrices.

When both ends of a relation, the entities, are set in the same transfer matrix, the relations connect a number of entity pairs, determining the degrees of the sparse transfer matrices. This addresses the heterogeneous nature of the entities and relations. The SparseTrans(separate) model deals with the imbalance, which occurs as the number of entities on either side of a relation vary in number. For this, it uses 2 separate sparse matrices, one for each entity and evaluates the degrees based on the number of entities in each space.

A project by Shi and Weninger, aimed on filling missing data in knowledge graphs by using a shared variable neural network model that learns joint embeddings of the knowledge graph's entities and edges with trivial changes to the standard loss function [28].The scalability of the model to massive knowledge graphs was a major concern that generated Shi et al.'s model, projE.

Also, the knowledge that the other models were found to use, were based on pre-trained embeddings. So projE considered the task of knowledge completion as a ranking task. Based on the ranking priority, the candidates are directed into 2 separate input embedding spaces using a combination bias as shown in (10):

$$e \oplus r = D_e e + D_r r + b_c. \tag{10}$$

Here, D_e and D_r are the entity and relation weights that are represented as $m \times m$ diagonal matrices and b_c is a the connective combination bias. This computes the embedding function using f and c which are activation functions as shown in equation (11). Here, $W^c \in R^{s \times k}$, such that s is the number of candidate entities:

$$h(e, r) = g(W^c f(e \oplus r) + b_p). \tag{11}$$

Considering the removal of erroneous data from knowledge graphs, Ryu et al. proposed an erroneous relation elimination method that removes the erroneous data from knowledge graphs. This rests on the concept that entities within a semantic relationship are represented by the same node [27]. Therefore, a single representative entity will be selected to represent each semantically similar relation. Consequently, error detection is performed based on relational weights and predefined limitations or conditions.

The Deep Fact Validation is yet another method that addresses this issue by providing users with brief extracts of web pages and a confidence score for the facts based on the sources from which they were retrieved [20]. This paved way to eliminate facts with relatively low confidence values, assuming a direct proportionality to their sources' accuracy.

The Probabilistic Soft Logic is a statistical relational framework that computes confidences in the form of soft truth values for facts or triples. As discussed in the Background section, we harness PSL, to infer confidence scores about the triples in order to use them as a feature. We garner the baseline performance index values for our experiments based on [26]. According to Pujara et al., the PSL threshold value that determines the baseline of the evaluations, is the threshold that gives the optimal f1 score. As such, the baseline performance index values for the precision, recall and f1 scores are 0.828, 0.910 and 0.867, respectively.

So far, all of these refinement techniques adhere to simply evaluating the validity of facts based on their sources, their semantic similarities, relational embeddings, all of which are completely automated and computed based on the dataset. However, the previous work does not incorporate human evaluation while measuring the accuracy of the triples in the knowledge graph. In contrast, we propose a machine learning based approach to incorporate human evaluation to achieve significant improvement over the baseline.

7 Conclusion

Knowledge Graphs are a sleek way to represent mass amount of data and model the relationship between various entities. However, they are not always complete or accurate. Hence, our solution model proposes a method to address the removal of inaccurate or erroneous facts from knowledge graphs. We consider the validity of the fact being accurate as a correspondence to actual human evaluations. Manually processing the knowledge graph facts can be expensive and extensive.

Thus, we use a machine learning approach, along with the probabilistic soft truth values computed using PSL, and an empirically derived feature, the fact strength, to train a model on a subset of human evaluated triples. Evaluating the trained model on unseen datasets rendered a precision improvement of 12.56% and 5.39% which achieved an average precision gain of 8.98%.

We were also able to achieve an average improvement of 4.44% for the prediction accuracy, in terms of the F1 score. Therefore, the inclusion of the fact strength as a training feature here, showed an average amplification of 2.13% in the precision as opposed to the model that did not use the fact strength.

Hence, the primary contribution of this paper is the proposal of a machine learning approach injected with a sufficient level of supervision through a subset of human evaluated fact truths and a probabilistic inference of fact confidences. These predictions can be used to eliminate inaccurate edges or relations in the knowledge graph, thus, refining it and addressing the erroneous data issue in knowledge graphs. Also, we intend on exploiting the capabilities of Bayesian statistics to compute the probabilistic confidences for the facts in our future work.

References

1. **Allemang, D., Hendler, J. (2011).** Semantic web for the working ontologist: effective modeling in RDFS and OWL.
2. **Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z. (2007).** Dbpedia: A nucleus for a web of open data. In The semantic web. Springer, pp. 722–735.
3. **Berners-Lee, T., Hendler, J., Lassila, O. (2001).** The semantic web. Scientific american, Vol. 284, No. 5, pp. 34–43.
4. **Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J. (2008).** Freebase: a collaboratively created graph database for structuring human knowledge. Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp. 1247–1250.
5. **Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O. (2013).** Translating embeddings for modeling multi-relational data. Advances in neural information processing systems, pp. 2787–2795.
6. **Boyd, S., Vandenberghe, L. (2004).** Convex optimization. Cambridge university press.
7. **Brocheler, M., Mihalkova, L., Getoor, L. (2012).** Probabilistic similarity logic. arXiv preprint arXiv:1203.3469.
8. **Buhrmester, M., Kwang, T., Gosling, S. D. (2011).** Amazon's mechanical Turk: A new source of inexpensive, yet high-quality, data? Perspectives on psychological science, Vol. 6, No. 1, pp. 3–5.
9. **Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., Zhang, W. (2014).** Knowledge vault: A web-scale approach to probabilistic knowledge fusion. Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp. 601–610.
10. **Fensel, D. (2001).** Ontologies. In Ontologies. Springer, pp. 11–18.
11. **Giaretta, P., Guarino, N. (1995).** Ontologies and knowledge bases towards a terminological clarification. Towards very large knowledge bases: knowledge building & knowledge sharing, Vol. 25, No. 32, pp. 307–317.
12. **Godbole, S., Sarawagi, S. (2004).** Discriminative methods for multi-labeled classification. Pacific-Asia conference on knowledge discovery and data mining, Springer, pp. 22–30.
13. **Golfarelli, M., Maio, D., Rizzi, S. (1998).** The dimensional fact model: A conceptual

model for data warehouses. International Journal of Cooperative Information Systems, Vol. 7, No. 02n03, pp. 215–247.

14. **Gruber, T. R. (1993).** A translation approach to portable ontology specifications. Knowledge acquisition, Vol. 5, No. 2, pp. 199–220.

15. **Guarino, N., Oberle, D., Staab, S. (2009).** What is an ontology? In Handbook on ontologies. Springer, pp. 1–17.

16. **Ji, G., Liu, K., He, S., Zhao, J. (2016).** Knowledge graph completion with adaptive sparse transfer matrix. AAAI, pp. 985–991.

17. **Kimmig, A., Bach, S., Broecheler, M., Huang, B., Getoor, L. (2012).** A short introduction to probabilistic soft logic. Proceedings of the NIPS Workshop on Probabilistic Programming: Foundations and Applications, pp. 1–4.

18. **Korfhage, R. R. (2008).** Information storage and retrieval.

19. **Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T. (2011).** Hmdb: a large video database for human motion recognition. Computer Vision (ICCV´11) IEEE International Conference on, IEEE, pp. 2556–2563.

20. **Lehmann, J., Gerber, D., Morsey, M., Ngomo, A.-C. N. (2012).** Defacto-deep fact validation. International Semantic Web Conference, Springer, pp. 312–327.

21. **Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X. (2015).** Learning entity and relation embeddings for knowledge graph completion. AAAI, volume 15, pp. 2181–2187.

22. **Liu, S., Liu, K., He, S., Zhao, J. (2016).** A probabilistic soft logic based approach to exploiting latent and global information in event classification. AAAI, pp. 2993–2999.

23. **Ma, Y., Gao, H., Wu, T., Qi, G. (2014).** Learning disjointness axioms with association rule mining and its application to inconsistency detection of linked data. Chinese Semantic Web and Web Science Conference, Springer, pp. 29–41.

24. **McGuinness, D. L., Van Harmelen, F. (2004).** Owl web ontology language overview. W3C recommendation, Vol. 10, No. 10, pp. 2004.

25. **Paulheim, H. (2017).** Knowledge graph refinement: A survey of approaches and evaluation methods. Semantic web, Vol. 8, No. 3, pp. 489–508.

26. **Pujara, J., Miao, H., Getoor, L., Cohen, W. (2013).** Knowledge graph identification. International Semantic Web Conference, Springer, pp. 542–557.

27. **Ryu, P. M., Jang, M. G., Kim, H., Hwang, Y., Lim, S., Heo, J., Lee, C. H., Oh, H. J., Lee, C., Choi, M., others (2013).** Apparatus and method for knowledge graph stabilization. US Patent 8,407,253.

28. **Shi, B., Weninger, T. (2017).** Proje: Embedding projection for knowledge graph completion. AAAI, volume 17, pp. 1236–1242.

29. **Spyns, P. (2005).** EvaLexon: Assessing triples mined from texts. STAR, Vol. 9, pp. 09.

30. **Suchanek, F. M., Kasneci, G., Weikum, G. (2007).** Yago: a core of semantic knowledge. Proceedings of the 16th international conference on World Wide Web, ACM, pp. 697–706.

31. **Wang, Z., Zhang, J., Feng, J., Chen, Z. (2014).** Knowledge graph embedding by translating on hyperplanes. Proceedings of the AAAI Conference on Artificial Intelligence, volume 28, pp. 1112–1119.

Article received on 27/02/2019; accepted on 17/01/2021.
Corresponding author is Manuela Nayantara Jeyaraj.