machine learning & knowledge extraction

MDPI

Article

Claim Consistency Checking Using Soft Logic

Nouf Bindris *, Nello Cristianini and Jonathan Lawry

Intelligent Systems Laboratory, University of Bristol, Bristol BS8 1UB, UK; nello.cristianini@bristol.ac.uk (N.C.); j.lawry@bristol.ac.uk (J.L.)

* Correspondence: nouf.bindris@bristol.ac.uk

Received: 12 June 2020; Accepted: 2 July 2020; Published: 6 July 2020

Abstract: Increasing concerns about the prevalence of false information and fake news has led to calls for automated fact-checking systems that are capable of verifying the truthfulness of statements, especially on the internet. Most previous automated fact-checking systems have focused on the use of grammar rules only for determining the properties of the language used in statements. Here, we demonstrate a novel approach to the fact-checking of natural language text, which uses a combination of all the following techniques: knowledge extraction to establish a knowledge base, logical inference for fact-checking of claims not explicitly mentioned in the text through the verification of the consistency of a set of beliefs with established trusted knowledge, and a re-querying approach that enables continuous learning. The approach that is presented here addresses the limitations of existing automated fact-checking systems via this novel procedure. This procedure is as follows: the approach investigates the consistency of presented facts or claims while using probabilistic soft logic and a Knowledge Base, which is continuously updated through continuous learning strategies. We demonstrate this approach by focusing on the task of checking facts about family-tree relationships against a corpus of web resources concerned with the UK Royal Family.

Keywords: fact checking; information extraction; probabilistic soft logic

1. Introduction

Increasingly information available on the internet is being claimed, rightly or wrongly, to be false. In fact, many people consider that false statements are crowding out true ones. Unfortunately, traditional, manual approaches to verifying the truth of statements are often no longer applicable, because of the sheer amount of information now being published on the internet. Recognition of this problem has prompted research into and the development of, automated computational approaches to the identification of potentially false information. Several automated fact-checking systems [1–5] have been developed and used for real-world applications. However, their operation has been limited to the use of grammar rules, such as linguistic features and other features to verify statements on the basis of the nature of language that they use. Our key observation is that it is often not possible to correctly check the truth of claims, but it may be possible to check their consistency with an established knowledge base. While other approaches use correlations with stylistic signals [5,6], we use consistency of beliefs. The underpinning idea behind this approach we use here, Facts Automatic Consistency Test (FACT), is to establish the relevant information that is required in order to verify a particular claim against that information. The underlying aim of FACT is to minimize inconsistency, as determined by inference using soft logic [7]. This requires information extraction, and then the application of logical inference to check the claim. This approach contrasts with those that study the subjectivity of language in order to detect such a thing as propaganda, satire, and hoaxes [5] or speculation [6].

This paper is a development of work presented in a previous study [8] and the culmination of [9]. Here, we demonstrate the method introduced in [8], but with the addition of a continuous learning process that is based on a re-querying strategy and adding more logical inference rules.

In this paper, we describe FACT as a system that builds a knowledge base using an information extraction tool for extracting constants, their properties, and their relations, with a given domain, and then fact-checking any given claims or statements by establishing their consistency with this knowledge base. Verification is achieved via logical inferences while using first-order logic rules, based on the information extracted from the domain. For example, if we were to say that it is raining in Bristol today, our approach would not be able to determine whether this claim was true or false. However, it would be able to check the claim is consistent with a knowledge base. If we have a claim and a trusted KB we could check if the given claim is consistent with the KB (true) or not (false) by trying to infer the negation of this claim from the KB. If we can then this suggests claim is false.

FACT adopt a logical approach to beliefs, so that a belief is the application of a truth value to a declarative statement. Here, a statements are ground atoms corresponding to instantiation of predicates and relations. The set of all such statements is referred to as the Herbrand Base (HB). An assignment of truth values to all the statements in a formal language is called an interpretation of that language. We can assign truth values to statements or claims directly or by observation from a given text using information extraction tools (IE). In addition, we can infer the truth value of other statements by checking whether there are interpretations that are consistent with these observations and a set of background knowledge constraints in the form of logical rules defining key concepts and relations. Because we cannot assume that the observations are entirely accurate; we adopt a notion of approximate consistency; we will attempt to make assignments of truth values to each element of a given set of statements by finding an interpretation, in relation to the HB, which is maximally consistent. For this, we will relax the definition of an interpretion to allow for truth values in the interval [0,1], and then apply Probabilistic Soft Logic (PSL) to identify maximally consistent truth value assignment.

Recent work using automated fact-checking methods is based on machine learning and deep learning techniques algorithms [10]. The main advantages of FACT are that it does not assess the stylistic or statistical properties of a text, and it does not need to compare a claim with a specific sentence in a given document. Instead, it is designed to assess the consistency of a claim with a set of other claims distributed across multiple sources. This work is more related, in spirit although not technically, to the use of knowledge-networks [11], since our approach is based on PSL and dynamic generation of the KB, which has advantages in handling inconsistencies and approximate inference, as well as efficiency.

The remainder of this paper is structured, as follows. Section 2 discusses related work with Section 3 giving the background to Probabilistic Soft Logic. In Section 4, we describe the software pipeline which has been developed for the purpose of fact-checking. This pipeline includes all of the processes involved: web querying, information extraction using a JAPE grammar, entity disambiguation, statement checking using PSL inference, and finally the re-querying necessary for enhancing the knowledge base. Section 5 presents our experimental work that involves a case study of the relationships between members of the UK royal family. Section 6 then gives a detailed evaluation of the pipeline, and finally, Section 7 gives some conclusion and discussion of possible future direction.

2. Related Work

Recent reference has been made to the problem of false published information. Hassan et al. [3] refer to half-truths, hyper-boles, and falsehoods, and Thorne and Vlachos [12] refer to the fact that false information can now be disseminated to millions of people. Mihaylova et al. [13] examined the issue of potentially false information, specifically in relation to community question answering forums. Mihaylova et al. [13] presented a system, which, similarly to our approach in [8], checked the validity of answers to particular questions, an issue that has been neglected thus far. The novel contribution of their work was a multi-faceted model that captured the diverse types of information that can be gleaned from the answers given in these forums, such as what is said, how, where, and by whom. A different

approach to fact-checking was taken by Rashkin et. al. [5], who determined the presence or otherwise of fake information by comparing the language of real news with the language of satire, propaganda, and hoaxes. Rashkin et al. [5] use a corpus from PolitiFact and argued that, while fact-checking remains an open topic for research, the truthfulness of a text can be determined by stylistic cues in the language. An approach similar to this, which also utilized the properties of the language itself, was by Nakashole and Mitchell [6], who used linguistic features, such as a subjectivity lexicon of strong and weak subjective words, sentiment lexicon of positive and negative words, Wikipedia derived bias lexicon and Part-of-speech (POS) tags were used to establish whether a source of information was objectively representing facts or was opinionated and/or speculative in nature.

In response to the fact that, over the last few years, there appears to have been an increase in false information published on the Internet. Mohtarami et al. [14] presented an automatic detection system while using end-to-end memory networks, which can check whether a particular statement agrees, disagrees, discusses, or is unrelated to, a target claim; the system uses convolutional and recurrent neural networks and a similarity matrix. The authors used the Fake News Challenge benchmark in order to test their proposed system, and found that their system, which can be considered feature light, performed as well as more complex systems. Automatic fact-checking was considered by Vlachos and Riedel [15] as a process that must include the identification of check-worthy statements. Hassan et al. [2,3] considered the creation of questions that are related to these statements, with a focus on finding information pertinent to the construction of a knowledge base related to these statements and then the inferring of the validity of such check-worthy statements. Thorne et al. [16] investigated automatic fact-checking whle using surface-level linguistic patterns. Specifically Thorne et al. [16] uses a hybrid convolutional neural network that integrates text with metadata, and that he argues improves text-only (rather than knowledge focused) deep learning. Popat et al. [17] produce CreadEey a user interface for fact-checking. The user enters a claim that needs to be checked and the output is the probability of it being true or false conditional on the information obtained from a web search for this claim. CreadEey also checks the trustworthiness of information sources and gives evidence in the form of a screenshot of the text containing relevant material.

Thorne and Vlachos [12] note that the interdisciplinary nature fact checking research has resulted in terminology inconsistency. In order to respond to this issue, Thorne and Vlachos [18] surveyed the various research efforts concerned with automated fact-checking that are based on natural language processing.

Hassan et al. [3] base their approach to fact-checking on the analysis of the meanings and characteristics of natural language sentences. They introduce ClaimBuster, a system that adopted this approach to fact-checking by analyzing natural language statement with a particular political discourse. Their system works by firstly, classifying and ranking sentences into non-factual, insignificantly factual, and check-worthy sentences; the latter are then labeled in an appropriate way and feature extraction is applied so as to provide a dataset for training [19]. However, the limitation of ClaimBuster is that there are discrepancies between the classifications made by the human checkers employed and those made by the machine system. Karadzhov et al. [20] presented a fact-checking system that is similar to ours, in that claims that are to be verified must be constructed or identified outside the system. The verification of claims is also the focus of the method used by Baly et al. [21], who developed an approach that involves determining the stances of documents in relation to a claim and then determining whether the claim is factual.

In response to the problems that are associated with using traditional methods for fact-checking the ever-increasing volume of information published online, Ciampaglia et al. [11] offered a computational approach that uses knowledge graphs that represent semantic proximity from transitive closure that is showing the smallest bathe between two nodes. Thus, their approach leverages existing bodies of expert knowledge to assess the truth of statements. This is similar to our approach, which is also based on known truths. Another related approach, Shi and Weninger [22], which uses knowledge graphs that incorporate predicate interactions and connectivity. In the present study, we use entities

and the relationships between them in fact-checking processes. Again, similarly, Ciampaglia et al. [11] and Shi and Weninger [22] both used statements of fact in the form of a subject, a predicate, and an object, whereby there is a stated relationship between them. A different approach to fact-checking has been proposed by Popat et al. 2016 [23] who, although using a domain setting to check the credibility of claims made in natural language, utilizes an inference based on a joint interaction between the kind of language employed by the claim and the reliability of the web source. The same authors extended their work [24] to include the claim's temporal footprint on the web; this was shown to be effective for the early detection of emerging claim.

In this, we propose an approach in which we search for an interpretion of the language which is consistent with a given KB. We focused on looking for interpretations that are highly consistent or are maximally consistent with KB given a set of rules. Verification of the consistency of statements is performed using PSL to conduct the necessary logical inference [8]. Our method starts with a process that extracts the relations and constants from texts using an information extraction tool, which can work across any domain to build a trusted knowledge base. Once this is built, we can check the consistency of given statements with the KB, in terms of pre-specified rules. In addition, there is a mechanism for re-querying which enables the expansion of the knowledge base, so that continuous learning can be achieved. This research uses PSL as a rule-based probabilistic framework in order to infer facts that are not explicitly mentioned in the text and check their consistency with a trusted knowledge base. This paper shows how PSL can be used to assess the consistency of one fact against others. In the next section (Section 3), we introduce PSL and its underlying method that is based on HL-MRF.

3. HL-MRFs and PSL

Probabilistic Soft Logic (PSL) [25] is a many-value logic that uses an inference mechanism that is based on Hinge-Loss Markov Random Field (HL-MRF) [7]. HL-MRF is a graphical model that is analogous to a discrete Markov Random Field (MRF) [26]; but, instead is applied to continuous variables in the unit interval [0,1]. The associated probability density is defined over the random variable Y as a condition of the variable X, as follows:

$$P(Y|X) = \frac{1}{Z(\lambda)} exp[f_\lambda(Y, X)] \tag{1}$$

Here $Z(\lambda)$ is a normalization item and $f_\lambda(Y, X)$ is a constrained Hinge-Loss feature function.

$$f_\lambda(Y, X) = \sum_{r \in R} \lambda_j \phi_j(y, x) \tag{2}$$

where $\lambda \in (\lambda_1, ..., \lambda_m)$ are the wights of the potential $r \in R$, which shows the importance of the respective potential (rules) in the model and R is the set of all the potentials in the model. $\phi_j(Y, X)$ are potentials represented by Hinge-Loss functions that make the model tractable.

In this paper, we use PSL as a framework for modeling rich relational domains and probabilistic models in order to specify the properties of the HL-MRF. PSL is a programming language that is easy to use and implement and that can be easily represented by HL-MRFs. Models can be specified via a first-order logic syntax and that can also be used to describe the features that define a HL- MRF. The following is an example of a PSL logical rule:

$$w : P(X, Y) \wedge Q(X, A) \Rightarrow R(Y, A)$$

Where w is the weight for the rule and indicates the significance of the rule in relation to all the rules in the ruleset. Additionally, P, Q, and R are predicates of the model, and the X, Y, and A are its variables. E.g., P could be $friend(X, Y)$, which represents the fact that the two variables, X and Y, are friends. For example,

$$1.0 : friend(X, Y) \wedge support(X, A) \Rightarrow support(Y, A)$$

The above means that a person will support the same team as his/her friend. Each of the predicates in the logical rule is an atom (e.g., support(X,A)). An atom becomes an instance or ground atom when it is variables are instantiated to constants (e.g, support("Tom","Manchester United")). Each ground atom is mapped to a truth value in the interval [0,1] and this mapping then constitutes an interpretation I of the language. PSL interval values between 0 and 1 relaxing the interpretation of the connectives using Lukasiewicz logic for the logical conjunction ($\wedge$), dis-junction ($\vee$), and negation ($\neg$) operators, as shown in the following: Here, a and b are truth value in the interval 0 and 1:

$$a \wedge b = max\{0, a + b - 1\} \tag{3}$$

$$a \vee b = min\{a + b, 1\} \tag{4}$$

$$\neg a = 1 - a \tag{5}$$

Equations (3), (4), and (5) show how Lukasiewicz logic is applied in order to relax the logical rules that are to be employed in the potential function. Using the above relaxations and the logical identity $p \Rightarrow q \equiv \neg p \vee q$, a ground instance of a rule r ($r_{body} \Rightarrow r_{head}$) is satisfied (i.e.,$I(r) = 1$) when ($I(r_{body}) \leq I(r_{head})$). PSL defines a probability distribution by quantifying a distance to satisfaction for each grounded instance of a rule. A rule's distance to satisfaction under interpretation I is calculated, as follows:

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

All of the rules in a PSL model induce a hinge-loss potential of the form (6), in which the loss function, I_r, is defined in term of the distance to satisfaction of the rule, as in (1). It follows that the goal of optimization is to minimize the weighted sum of the distance to satisfaction of each and every one of the rules. Because Equation (1) is a log-concave in Y, PSL provides a way to solve the problem of finding (or inferring) the Maximum A-Posteriori Probability (MAP) in relation to HL-MRFs, by using a convex optimization algorithm operating with continuous truth values. The problem of finding the MAP can be restated as the problem of finding the values for the free variables Y given observations X. The convex optimization problem is then solved while using the method in Bach et al.'s [7], applying an alternating direction method of multipliers (ADMM).

Figure 1 shows the general steps, whereby PSL assigns a truth value to a variable. PSL has as input a set of rules and some data (observations). First, PSL will ground the rules using the observations; this yields the ground rules. Next, we relax the ground rules while using Lukasiewicz logic to make it possible for PSL to assign an interval values iv [0,1]. After relaxation, each rule will give us a convex potential function. PSL will take the weighted average sum of all the rules giving an overall potential function. This is then maximized using the Alternating Direction Method of Multipliers (ADMM algorithm) [27].

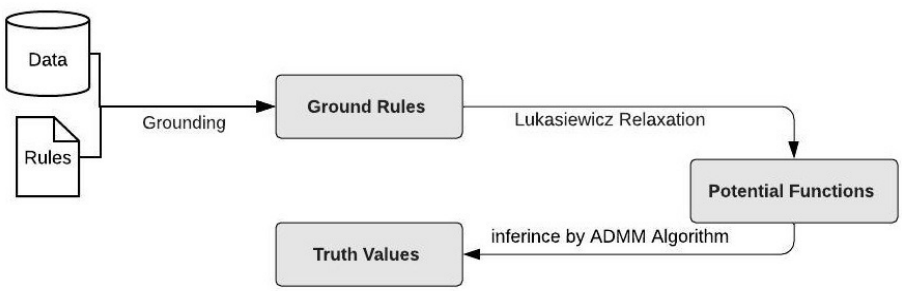


Figure 1. How PSL assigned truth values for the random variable.

4. Fact Inference and Consistency Checking

The primary purpose of the proposed methodology is to discover whether a given claim in natural language is consistent with a set of documents retrieved from the web specifically in relation to that claim. Our method builds a universe of discourse for any chosen domain by employing an information extraction tool in order to extract entities along with their properties and relationships to build a knowledge base (KB) that consists of constants and relations. Specifically, the constants we are interested in here are named entities from the texts and the relations of those entities. The relations we are interested in are those which are specified in a limited number of pre-written (using a JAPE grammar) pattern rules. These rules are used to extract matching 1-ary and 2-ary relations from the texts (NB, relations between constants are ground atoms). We will demonstrate these two extraction steps of the texts and extraction of the information from the text in the next section (Section 4.1). After building a KB according to the above method, PSL is employed as a logical inference tool to check claims by searching for interpretations that are maximally consistent with the KB and our rules.

The set of relations used in logical inference comprise the ground atoms that are used to create our HB elements. An HB consists of all the ground atoms and their negations. Each ground atoms in the HB must be assigned a truth value in order to generate an interpretation that is as consistent as possible with the KB and rules. The elements of the HB are linked together by rules forming a network and the connections of this network are created by the list of rules. The optimal solution is to find one interpretation that is consistent with all the rules and all facts. Our set of rules regarding the universe of discourse in question has been written in first-order logic (FOL) and these will be used in the logical inference tool, here PSL.

PSL search for an interpretation that is maximally consistent across the HB network. Moreover, in many cases such there is no entirely consistent interpretation. In these cases, PSL will then attempt to find the one interpretation that violating the fewest rules. In this interpretation, inconsistency will exist and cannot be entirely eliminated, only minimized. In such cases, PSL will use truth values in an interval of 0 to 1 and finds a maximally consistent interpretation for the HB by undertaking relaxation and approximation.

In the next sub-sections, we will describe the different steps in our pipeline and how our method operates by using information extraction and logical inference to check a given set of claims

4.1. FACT Pipeline

This sub-section describes the software pipeline that we have built for checking the consistency of claims with trusted facts being represented in the KB. The purpose of the pipeline is to facilitate the input of a claim about a relation between two entities and then to generate a corpus by querying the web specifically in relation to that claim. Existing information extraction tools are then used to extract named entities (constants) and the relations between them from the text; this process creates the KB and will enable us to check that the claim is consistent with respect to the KB, while using PSL. Here, we assume that our claims/queries are concerned with binary (Boolean) relations and are written in natural language (English), which is parsed using a Natural Language Processing tool. The results of this process constitute the relations that we need to parse for (across the web). Our experiments will be described in detail in the Experiment Section (Section 5). The pipeline was implemented employing pre-existing tools wherever possible. Figure 2 shows our approach pipeline. In order to clarify this further, we will present the different stages of our pipeline in detail. Each FACT stages were showed next in detail.

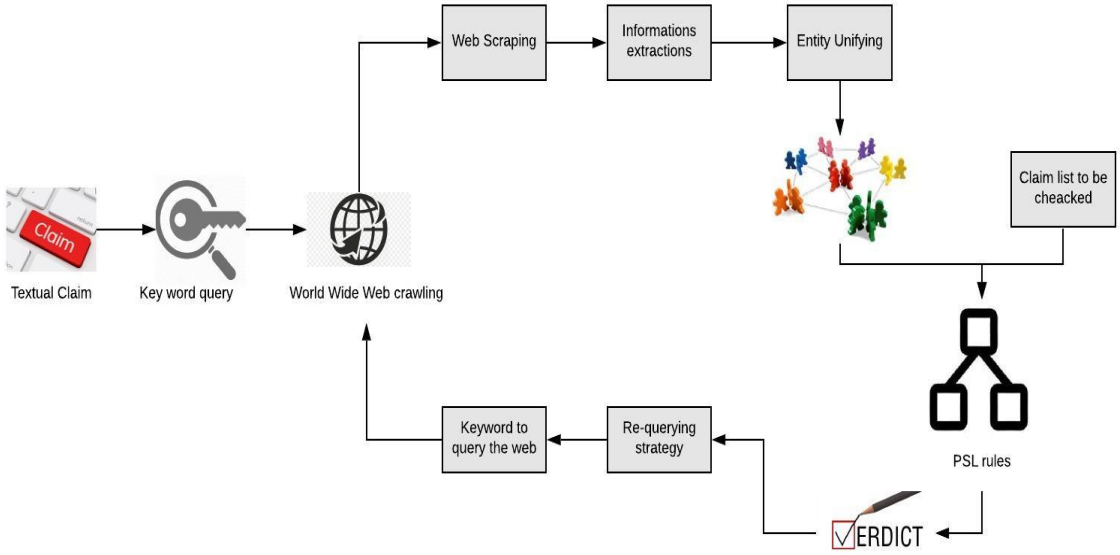


Figure 2. Facts Automatic Consistency Test (FACT) Pipeline.

4.1.1. Web Query for a Claim in Natural Language

Processing starts with the input of set of claims to be checked, and we randomly choose a claim from which, we generate a web query to build a text corpus, as follows:

- Generated Query: Given a claim specified as a natural language text or query we extract from it two named entities; this pair of entities then constitutes our search query. The two entities could be any name entities type.
- Query the Web: this part of the process starts by crawling the web for links related to the given search query; this is in order to build the corpus and knowledge base.
- Web scraping: for FACT, we built a web scraper, a tool that extracts relevant articles from the links found and that perform some pre-processing of the web page in order to remove noise and yield 'clean' articles.

This process is repeated for all the claims.

4.1.2. Constants and Relations from Text

In the next stage, we build our KB by extracting the constants and relations from the text corpus created in the previous stage. Our KB consists of constants and relations. The constants are entities without duplicates. Co-reference, anaphora resolution, and disambiguation are carried out as outlined below:

- Information Extraction (IE) processing: information extraction is performed on the scraped articles in a number of steps, as follows:
  - Pre-processing: pre-processing takes place on the data intended for the corpus. Before starting fact and relation extraction, our pipeline applies a tokenizer, a PoS tagger, and a syntactic parser to the data. For this pre-processing, we use ANNIE, 'A Nearly-New IE' system from GATE. ANNIE combines the resources of a sentence splitter, a tokenizer, a PoS tagger, a gazetteer, and a JAPE transducer[28]. In general, ANNIE adds annotations to the text in order to indicate the positions of the elements identified by these processing resources. ANNIE performs the role of a named entities recognition tool, extracting named entities, such as a person, organizations, etc.
  - Co-reference Resolution: the text of each article is processed for named entity co-reference resolution; this is the process that determines whether two distinct natural language

expressions, found in the text, actually refer to the same entity in the world [29]. By this process, we create a list of entities without duplication. The Orthomatcher module of the ANNIE Information extraction module of the GATE (General Architecture for Text Engineering) system distribution [29] is used to perform this task.
  - Anaphora Resolution: once the co-reference has been resolved the pronominal resolution module in ANNIE is employed to perform anaphora resolution. The system resolves pronouns that present themselves in all the forms that are recognized in GATE.
  - Grammar rule: we use the JAPE grammar processor to extract patterns. Its grammar processing is carried out over a set of phases, each of which employs a specific set of pattern rules. Each phase is executed sequentially and the whole constitutes a cascade of finite state transducers over annotations (these being obtained from ANNIE). In the grammar rules, the left-hand-sides (LHSs) consist of a description of an annotation pattern that must be found in the file for the rule to be triggered. The right-hand-side (RHS) of each rule consists of a set of annotation manipulation statements. Annotations (e.g., representing persons, organizations, etc.) that are matched by the LHS of a rule may be referred to on the RHS by means of labels that are attached to the pattern elements [30]. Once an LHS is matched, a new annotation may be added to the file by the RHS.
- Store in Relations DB: we count how many times each relation has been extracted from the trawled data in order to decide which relations are to be considered trusted and, therefore, can be added to our KB. We set a threshold for the number of times that a relation must be repeated for it to become trusted. This criterion is combined with restrictions relating to trusted websites in order to determine whether a relation is to be stored in the database along with the information regarding the article from which it was extracted.

4.1.3. Named Entities Unifying (Cross-Document Co-Referencing Resolution)

One of the most important stages in this process is named entity unification, which involves identifying whether ot not two entities that are encountered in different text sources are the same. For this, we apply co-referencing resolution across different sources. If two such entities are the same, a standard name will be used for it throughout, regardless of the name by which it is referred to in the text [31]. Many different studies have tackled the processing that is involved in this task [31–34]. Here, adopt the following methodology proposed in [35] for identifying the named entities encountered in different text sources and representing each instance by a standard name for the referred-to entity.

- For each text extract the co-referencing chains (using GATE) and set these as the local co-referencing chains.
- Extract features from the KB by apply the Ambiverse Natural Language Understanding(AmbiverseNLU) tool [36]. Additionally, features we extract using this are type, Wikipedia link, and name.
- The next process to be performed is similarity-based clustering for each entity so that the co-references of each entity and features from the KB can be placed in one cluster.
- Lastly, entities exhibiting high similarity will all be allocated to one cluster and we combine the entities using co-referencing chains, such that we can then use the standard name as the main name for this cluster group.

Once each instance of each entity is represented while using its standard name, no duplicate names will exist in the texts. This will eliminate any duplicate extracted relations.

4.1.4. Claim Consistency Checking Using a Logical Inference Tool

PSL was used for consistency checking as follows:

- PSL MAP inference is used here as the means by which the most consistent interpretation of each claim is found. In PSL programs, the required observations are defined to be the relations that we have previously stored in the KB. For more details regarding how PSL work is found in Section 3.

4.1.5. Re-Querying the Web (A Continuous Learning System)

The last stage in the procedure is to expand the KB in such a way that more is learned each time a claim is processed. This is achieved by re-querying, following the strategy described below:

- Re-querying: this step consists of querying the web with a new, specifically constructed, query in order to expand the KB. There are a number of different strategies that could inform this re-querying. The particular strategy that is followed here is that of obtaining information relating to the claim that has just been processed. This information will be focused on two entities which have been identified as a result of the claim input by the user; a fact about the relationship between these two entities that could not be answered by the above processing, or a statement that could not be interpreted due to there being insufficient information stored in the KB to be used as the basis for the re-querying. The web is re-queried in order to learn more and expand the KB[9].
- Search the web with the next query from the user and process using the information obtained from the re-querying strategy.

Here, we do not discuss the issue of the validation of the software because we cover this in the evaluation Section 6.

5. Experimental Case Study

5.1. Experiment Setup

This sub-section covers the set-up for carrying out our experiment. First, the JAPE grammar was written, specifying the textual forms of the relationships to be extracted. Next, the PSL model with all of its rules was constructed. The case study concerns the UK Royal Family's kinship relationships and line of succession from Queen Victoria to Prince Louis; this set-up enables users to check statements regarding the royal family's kinship and family-tree relationships. The target data for this experiment consisted of a number of claims to be checked, corresponding to statements made about family tree relationships between the identified persons.

Initially, the user inputs a query, from which we extract the keywords that correspond to the two entities or constants involved. We then search the web to extract articles that are relevant to these entities and create a corpus focused on them. From this corpus, we extract entities that are the relevant constants. In this case, these will all represent members of the royal family. We also extract the constant properties of these people, such as gender. In addition, we extract the parent relations and spouse relations. All of these properties and relations are extracted via text patterns that are specified in JAPE (one of the facilities offered by ANNIE). This processing is undertaken in order to build the KB.

From the extracted relations, we create the PSL model, which includes the predicates defining the observations and also the predicates defining the relations to be inferred. The predicates for the observations relate to the information we extract from the web in order to build the KB and the predicates for the inferred relations allow PSL to infer new relations, as will be shown later. In addition, we constructed a set of rules written in first-order logic (FOL), to encode the family-tree relationships while using the relations extracted from the text.

Our approach has been designed to check the consistency of the relative truth of a given set of statements in relation to our KB and the predefined rules. Initially, we assigned truth value of *zero* to all of the claims there by using negative priors [7] which adopts the initial belief of being false except the ones which have evidence to the contrary. If some claims remain with low truth values this does not necessarily mean that they are false or inconsistent but may mean only that there is no evidence

that they are true. We use negative priors since PSL maximization of truth values tends to mean that it will tend to allocate truth values of close to *one* to statements unless there are specific constraints that prevent this. This will tend to overestimate the true value over a claim. In other words, it will return the truth value of true if this it does not conflict with the KB. In contrast, using negative priors FACT will only return the result that the claim is consistent/true if there is direct evidence to support it. A consequence of this approach is that a truth value close to *zero* simply indicates the absence of any direct evidence for a claim being true. This might mean that it is false, but it could also be true and that the relevant evidence has not yet been found. In our experiment, the consistency of each statement or claim that we need to obtain relies on accepted definitions of family-tree relationships. We aim to find an interpretation that is consistent with all of the rules that we have defined for the approach. An interpretation that is entirely consistent may not exist because, essentially, there may be noise in the data. In these circumstances, PSL will attempt to find an interpretation that is as consistent as possible, but that may not be completely consistent.

In the next sub-section (Section 5.1.1), we will describe the steps that are followed from claim to web querying to corpus creation and, then, from the constructed corpus, the extraction of the constants and relations. Subsequently, the grammar that we use in our model to extract the relevant patterns, the relations, are shown next. Lastly, we demonstrate the checking of the consistency claims while using a logical inference tool (here, PSL). Finally, we show how we model the family-tree relationship problem in PSL.

5.1.1. Family Relationship JAPE Grammar

Our experimental approach extracts two types of patterns, those that represent Parent relations and those that represent Spouse relations, as shown in Table 1. The patterns may be amended or added to, manually, whenever new kinds of the pattern are encountered in the article texts. The JAPE grammars for extracting Parent and Spouse relations are shown in Table 1.

Table 1. JAPE grammar patterns.

Relation	Patterns
1. Parent relations	
	• $Person, \{Tokens\}, Token == (\text{“}child\text{”} \mid \text{“}son\text{”} \mid \text{“}daughter\text{”}), Token == \text{“}of\text{”}, Person$
	• $Person, \{Tokens\}, Token == (\text{“}child\text{”} \mid \text{“}son\text{”} \mid \text{“}daughter\text{”}), Token == \text{“}is\text{”}, Person$
	• $Person, \{Tokens\}, Token == \text{“}is\text{”}, Person, Token == (\text{“}child\text{”} \mid \text{“}son\text{”} \mid \text{“}daughter\text{”})$
2. Spouse relations	
	• $Person, \{Tokens\}, Token == (\text{“}married\text{”} \mid \text{“}wife\text{”} \mid \text{“}husband\text{”}), \{Tokens\}, Person$

There are three different patterns that trigger the extraction of a parent relation:

1. if a Person entity is followed by the word *child* or *son* or *daughter* and this is then followed by the word *of* followed by a Person entity, the second person is assumed to refer to a Parent entity;
2. if a Person entity is followed by the word *child* or *son* or *daughter*, and this is then followed by the word *is* followed by another Person entity, the first person is assumed to refers to a Parent entity; and,

3. if a person entity is followed by the word *is* and this is then followed by a Person entity followed by the word *child* or *son* or *daughter*, the second person is assumed to refer to a parent entity.

Thus, the approach annotates the relations found by the above patterns as Parent/child appropriately. {Tokens} are pronouns or stop words that can occur in between the words that are identified by these patterns.

Another set of patterns is used in a similar way to annotate Spouse relations: (1) if a *Personentity* is followed by the word *married* or $wife$ or *husband*, and this is then followed by another *Personentity* then each person entity is assumed to be the spouse of the other. These annotation patterns will be detected in the text, where they exist, and the information extracted from them will be used in order to build the KB.

The approach extracted such a relation and used it in the KB if (a) a large number of equivalent such relations were encountered in different texts; (2) the patterns were found in reliable sources; and, (3) these patterns were repeated across a number of texts from a number of different sources. In the next sub-section (Section 5.1.2), we will show how we modeled the family-tree relationship problem in PSL.

5.1.2. Modeling Family Network Relations in PSL

The PSL model is based on observations regarding the relations stored in the data, i.e., the relations which were extracted from the text. After extracting these relations, we are able to consider the situations common to questions regarding family-tree relationships. In our model, the constants are the people in the relations and these constants have some properties, such as the constant property gender. The predicates for our proposed family model are as follows: $Male(X)$ designates that X has the property of being male; $Female(X)$ designates that X has the property of being female; $Parent(X, Y)$ designates that X is the Parent of the child Y; and, $Spouse(X, Z)$ designates that there is a marital relation between X and Z, one is the husband and the other is the wife. The information extraction task extracted instances of these relations from the text and added them as observations to be processed by our PSL model.

From the above predicates are a set of conditional rules, we can also reason about unobserved relations. The predicates relating to unobserved relations that were used in our model are shown in Table 2, both gendered and ungendered family relations. Examples of a few of the logical rules we used to infer relations are discussed later.

We now propose rules, using these predicates, which encode the family-tree relations on the basis of which we will make inference and predictions. Some of the main family-tree relations rules are shown below, and the rest can be found in appendix A:

$$Parent\_Of(X, B) \wedge Parent\_Of(X, A) \wedge A \neq B \Rightarrow Sibling\_Of(A, B)$$
$$Parent\_Of(X, B) \wedge Parent\_Of(Y, A) \wedge Sibling\_Of(X, Y) \Rightarrow Cousin\_Of(A, B)$$
$$Parent\_Of(X, B) \wedge Sibling\_Of(X, Y) \wedge Female(Y) \Rightarrow Aunt\_Of(Y, B)$$
$$Parent\_Of(X, B) \wedge Sibling\_Of(X, Y) \wedge Male(Y) \Rightarrow Uncle\_Of(Y, B)$$
$$Parent\_Of(X, B) \wedge Sibling\_Of(X, Y) \wedge Male(B) \Rightarrow Nephew\_Of(B, Y)$$

The first of the above rules defines the relation of sibling; this rule states that, if X is the Parent of B and X is also the Parent of A and A and B are different people, then B and A should be considered to be Sibling. The second rule states that, if A and B are Cousins, then if X is the Parent of B, and Y is the Parent of A, then X and Y are sibling. The third rule states that if X is the Parent of B and X is the sibling of Y and Y is a female then Y is the Aunt of B. The fourth rule enable the inferring of the Uncle relation and the fifth infers the Nephew relation.

Table 2. Inferred predict.

Un_Gender Predict	Gender Predict
Ancestor_Of(X,Y)	—
Descendent_Of(X,Y)	—
Sibling_Of(X,Y)	Sister_Of(X,Y) Brother_Of(X,Y)
Parent_Of(X,Y)	Mother_Of(X,Y) Father_Of(X,Y)
Child_Of(X,Y)	Daughter_Of(X,Y) Son_Of(X,Y)
Spouse_Of(X,Y)	Wife_Of(X,Y) Husband_Of(X,Y)
Uncle_Of(X,Y)	—
Uncle_In_Law_Of(X,Y)	—
Aunt_Of(X,Y)	—
Aunt_In_Law_Of(X,Y)	—
Niece_Of(X,Y)	—
Niece_In_Law_Of(X,Y)	—
Nephew_Of(X,Y)	—
Nephew_In_Law_Of(X,Y)	—
Cousin_Of(X,Y)	—
Cousin_In_Law_Of(X,Y)	—
Child_In_Law_Of(X,Y)	Son_In_Law_Of(X,Y) Daughter_In_Law_Of(X,Y)
Parent_In_Law_Of(X,Y)	Mother-In_Law_Of(X,Y) Father-In_Law_Of(X,Y)
Sibling_In_Law_Of(X,Y)	Sister_In_Law_Of(X,Y) Brother_In_Law_Of(X,Y)
Grand_Child_Of(X,Y)	Grand_Daughter_Of(X,Y) Grand_Son_Of(X,Y)
Grand_Parent_Of(X,Y)	Grand_Mother_Of(X,Y) Grand_Father_Of(X,Y)

In our experiment, each logical rule is assigned at an equal weight of 1.0. Additionally, the negative priors (set of rules) are included in the model with low weight, 0.1, so it will be easily outweighed by the other rules. These priors presume the initial beliefs built into the model [7]. An example of negative priories rules:

$$Sibling\_Of(A, B) = 0$$
$$Cousin\_Of(A, B) = 0$$
$$Aunt\_Of(Y, B) = 0$$
$$Uncle\_Of(Y, B) = 0$$
$$Nephew\_Of(B, Y) = 0$$

5.2. Experimental Result

In this section, we present the experimental result for the royal family network relations. Experiments are initialized by querying and scraping the web in order to build a corpus. Web pages were filtered before scraping; the articles referred to by web page links are only scraped if they are referenced on trusted news article web sites, such as that of the BBC. We did not accept as trusted any social media (e.g., Twitter) postings. We started with the given claim 'Is 'Queen Elizabeth II' the mother of 'Prince Charles'? This generated a search query in the form of the two entities 'Prince Charles' and 'Queen Elizabeth II' and using this query a KB is built by crawling the web for related links of these entities. Links to relevant articles to scrape were then extracted, and these articles were then placed in the corpus. This process was repeated for all the claims used in the experiment, in what we refer to as the continuous learning process, so as to expand the corpus and hence to improve fact-checking accuracy. In Table 3, we show how many articles were scraped during each iteration of the continuous

learning process. Following this, from the retrieved and pre-processed articles, we then created a KB by extracting the relations and the constants contained in this documents. Here, we employed the GATE IE tool and a JAPE grammar to identify the relations, while using the pre-designated patterns, as shown in Experiment Setup Section (Section 5.1.1). Table 3 also shows the number of relations that were extracted from the article for both parent and spouse with its iterations and how it increases in each iteration. For illustration, we show the first three iterations and a full table found in Appendix B:

Table 3. The table shows the number of articles processed in each iteration and the number of relations extracted.

Search Keyword	# Articles	# Parent Relations	# Spouse Relations
"Diana Spencer" "Prince Edward"	134	18	0
"Prince William" "Prince Philip"	114	30	0
"Lady Sarah Chatto" "Mia Grace Tindall"	43	30	4

In Figure 3, we show how our KB starts and continues learning and how the KB expands via this process. This extracted information constitutes the observations used PSL, as discussed in Section 5.1.2. The two extracted relation types, the spouse and parent relations, can be used to visualize a family tree, as shown in Figure 4.

Figure 4 shows a part of the extracted royal family tree with Queen Elizabeth as root and her descendants as nodes connected to this. FACT can accept a text corpus as input and then generate a tree, as shown in Figure 4. One weakness of this lies in the entities' linkage shown in the tree, where there are nodes with different person names that nevertheless refer to the same entity, e.g., Princess Anne and Princess Royal all refer to the same person. This issue is mostly solved by the unifies entities stage in the pipeline. However, this does not entirely solve the problem because the information residing in the KB may not be sufficient for this to happen. In addition, Lady Frances Armstrong-Jones is shown, in the tree, as a daughter of Princess Anne, and this is not correct. Inevitably, in general, the KB will contain some error, but, because of the flexibility of PSL in allocating the real-valued truth assignment that maximizes consistency, a small number of errors should not have a noticeable effect.

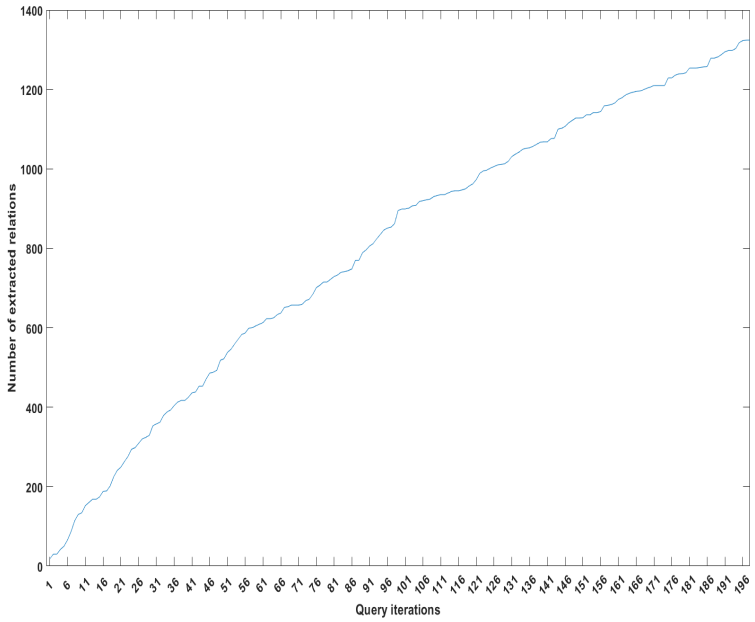


Figure 3. How the knowledge base (KB) increase with continuous learning. The vertical axis (y-axis) shows the number of facts FACT has learn. The horizontal axis (x-axis) shows the number of claims.

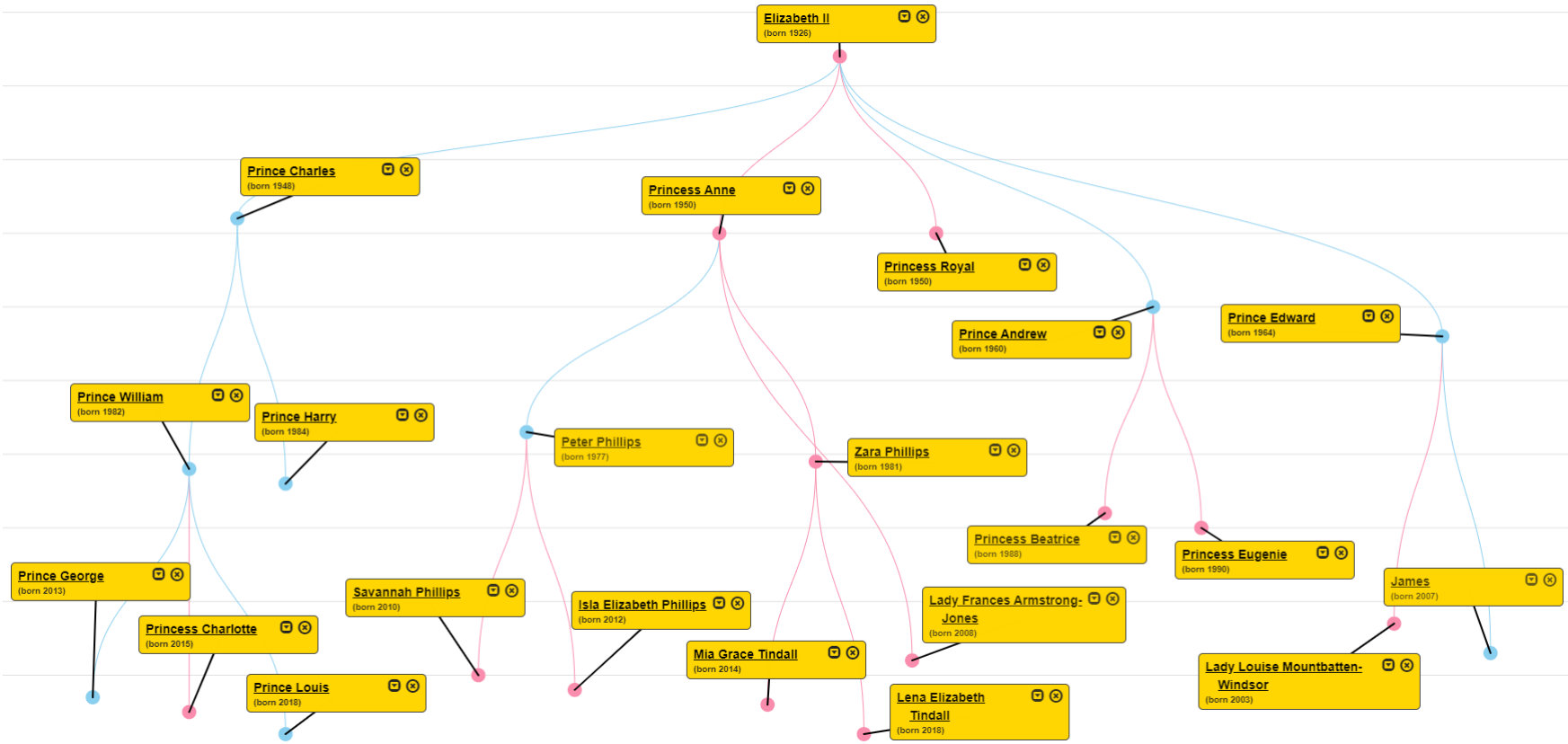


Figure 4. Family tree with Queen Elizabeth as root and her descendants.

Subsequent to the fact-checking, we ran the PSL model in order to infer further information, while using the model rules and observations. PSL will give each claim a truth value that indicates how confident PSL is concerning the truth of the best interpretation of the claim: i.e., how consistent the claim is with the KB. The PSL target set consisted of 99 true family relation claims and 99 false family relation claims; we will discuss the PSL result truth value next in the evaluation section (Section 6).

In the next section (Section 6), we will show how the continuous learning process will improve the performance, as quantified by the evaluation matrice (AUC) method, which can learn more information and is able therefore to more effectively process subsequent claims.

6. Evaluation

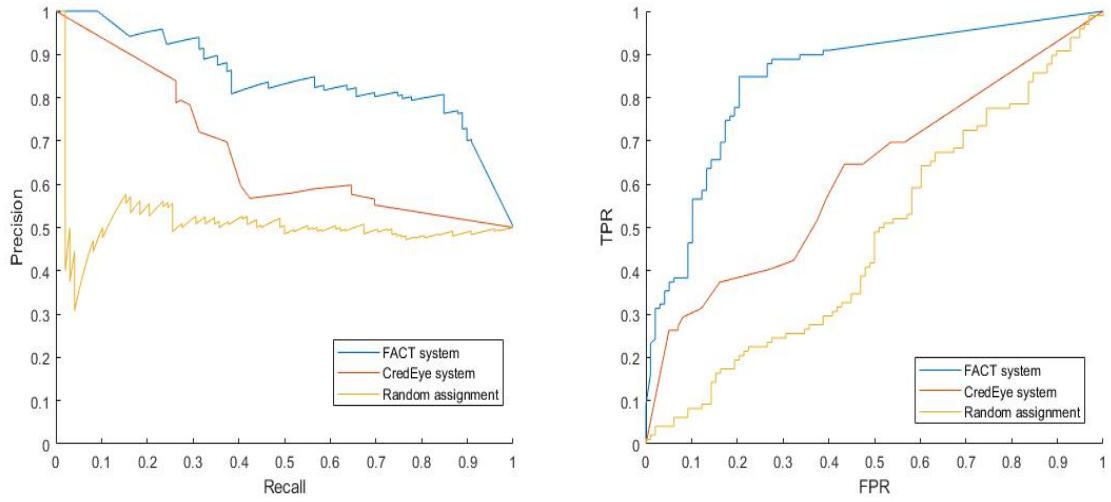
Estimating the effectiveness of this methodology is a difficult but important task. The ground truth set is a set of N claims about the royal family created from the UK Royal Family tree KB (Royal Family tree and line of succession: https://www.britroyals.com/royaltree.asp). It consists of 99 true claims about the UK Royal family's kinship relationships, extracted from the UK family tree, and 99 false claims created by randomly matching up a list of the UK Royal Family's personal names with a list of different family relations and, again, a random member of the Royal family's name. As example of a claim from the ground truth set:

Prince Edward is the child of ElizabethII. "True statement"
David Armstrong − Jones is the nephew of ElizabethII. "False statement"

The factual ground truth set is hierarchically structured, starting from the parent and leading to the great-grandchildren (Ground truth set could be found in Ground truth). The ground truth set was then added to the PSL model as targets. We then ran PSL to check how many of the factual claims would be returned as consistent with most of the rules and the KB in the model and how many false statements would be returned with truth values that are close to zero.

For comparison, we ran both our model and CredEye (A credibility lens for analyzing and explaining misinformation (https://gate.d5.mpi-inf.mpg.de/credeye/)) to fact-check the ground truth set. CredEye is a user interface for an automatic fact truthfulness checking system, which takes, as input, a claim or statement in natural language text and crawls the web to extract articles that are related to it. The method then checks the language style and stance of the articles along with the trustworthiness of the underlying sources in order to judge the truthfulness of the given claim and to provide some evidence for the validity of this judgment. It provides a value that indicates how probable it considers the input claim to be and, in contrast, the probability of it being false. We will use the truth probability returned by this tool to compare with the PSL truth value returned by our approach for the same claim. Due to our use of negative prior, all claims are initially allocated values of zero. After running FACT, the aim is for the truth value of false claim to remain close to zero, while those of true claim increase to close to one.

After running the ground truth set of 99 true and 99 false statements on both systems several times, we produced ROC curves in order to measure the performance of both systems in terms of fact-checking. Figure 5 shows the ROC curves for both methods. The ROC curve is a graph of the true positive rate (TPR) on the vertical axis (y-axis) and the false positive rate (FPR) on the horizontal axis (x-axis) at various threshold settings, which here are the different truth values given for all the ground truth set. The TPR is the number of claims correctly classified as true (TP) divided by the total number of actual true claims. On the other hand, the FPR is the number of false claims that were classified as true (FP) over the total number of claims that were actually false. Performance can be evaluated by the area under the ROC curve (AUC), which, for our method, was a proportion of 0.79 (Figure 5a FACT line) as compared with 0.62 for CredEye (Figure 5a CredEye line).



(a) ROC curves. **(b)** Precision and Recall curves.

Figure 5. Performance figures. In the graph, the blue line represents system FACT, the red line represents CredEye system, and the yellow line represents the random assignment.

Moreover, we can also assess this by a null hypothesis by generating a random truth value for all of the ground truth set, this will theoretically yield a rising diagonal line containing 0.50 of the area, as was confirmed empirically in Figure 5a random assignment line, where the area that we obtained with random assignment is 0.49. In order to show how far our method gives a true prediction that is significantly more often correct than the random method. We repeated the test of this null hypothesis 1000 times using both average and best performance results from each method. For all statements, the p-values were < 0.001, which demonstrates that our method performs statistically significantly better than chance/random assignment in predicting correct truth values. Therefore, taking the chance success rate into account, FACT has out-preformed CredEye, in more details FACT has scored 0.30 above the chance rate, CredEye, on the other hand, has scored only 0.13 above the chance rate. It is clear that FACT is twice as better than CredEye when compared to the chance rate. FACT has outperformed CredEye, because it was able to check claims using facts that were not explicitly stated in the text. For Credeye, in contrast, if the information is not explicitly presented in the retrieved text, such information cannot form the basis of any claim checking. On the other hand, FACT can infer information not explicitly presented in the text from different text source.

In addition, we also compared the two methods while using the metrics Precision and Recall [37,38]. Precision is the number of true identified claims (TP) divided by the total number of true and false identified claims (TP+FP); in other words, it is the percentage of the claims correctly identified as true among all of the claims classified as true. Recall, on the other hand, is the TPR, as defined above. We calculated the precision and recall of both systems and then plotted the precision and recall curves, as shown in Figure 5b for FACT line, and Figure 5b for CredEye line. This was done in order to show how both of the methods classified the claims made. Once again, our method comes out best. This can be seen from the fact the area under the line is much greater for FACT line in Figure 5b than for CredEye line in Figure 5b. Again, both may be compared with the result from random assignment of truth value to all ground truth set. That yields a theoretical horizontal straight line at 0.5 on the side scale, with 0.50 of the area beneath it, closely matched by our empirical calculation of the result with chance truth assignment, represented in random assignment line, as shown in Figure 5b. The Precision and Recall curve here for FACT is showing more accurate classification for the ground truth when comparing with CredEye.

In conclusion, we see that our method performs significantly better than both random truth assignment and CredEye in this case study.

7. Conclusion and Future Work

This paper has presented a novel fact-checking approach incorporating information extraction, soft-logic based reasoning, and a method whereby the system can 'learn' and so add to its knowledge-base on a continuous basis.

Information was extracted from a set of corpora of trusted sources; this information was then added to the system's knowledge base. A claim could then be checked for consistency with this knowledge base. An absolutely key affordance of the system was its ability to provide an 'assessment' regarding the validity of a claim, even where the knowledge base did not contain sufficient explicit knowledge to determine this entirely. The system achieved this via its use of soft-logic inference. Where there was insufficient information to determine whether, in absolute terms, the claim was entirely consistent, entirely inconsistent, or most likely consistent with the knowledge base, a guess was made as to its consistency by constructing the most consistent interpretation of the claim (as either being true or not true) which is an interval value in $[0, 1]$. In order to determine the 'assessment' to be output, the soft-logic system was applied to the claim in relation to the knowledge base. The second key affordance of the system was re-querying for new corpora of trusted sources in order to find out more information and increase the knowledge base.

The system, as it is now, has a number of limitations. One of these is that it can only deal with knowledge and claims relating to a quite limited universe of discourse—the family-tree relationships that exist among the members of the British royal family. Secondly, the 'assessment' it is able to provide is only as good as the information that it extracts from its corpora and the soft-logic rules that have been constructed for it.

Regarding the first point of course, it is possible to construct a system on the same basis, which deals with other topics or, indeed, multiple topics. Additionally, this would be a valid direction for future research.

In terms of its limitation as regards its corpora, future research could mitigate this in a number of ways. First, the information contained in the corpora can, of course, become out of date. Between January 2009 and January 2017, any reference in the corpora to 'the president of the united states' would almost certainly be to President Obama; from January 2017, such references would more likely be to the new incumbent, President Trump. Thus, it would be very useful to conduct research to consider ways and means of updating such time-based and knowledge.

The above is a list of limitations and directions for possible future research and so might be taken, overall, as a criticism of the system, but it is not. Previous systems have largely focused on narrow considerations of linguistic consistency. By using this three-pillar approach building a knowledge base from corpora, the use of soft logic inference, and continuous learning this method has a wide enough foundation to allow for indefinite future development. Some of the possible directions of this development have been discussed above.

Author Contributions: All authors contributed equally to the writing of this paper. Data curation, N.B.; Investigation, N.B.; Methodology, N.B., N.C. and J.L.; Supervision, N.C. and J.L.; Validation, N.B.; Visualization, N.B.; Writing—original draft, N.B.; Writing—review & editing, N.C. and J.L. All authors read and approved the final manuscript.

Funding: N.B. was supported by a grant from KSU, Saudi Arabia.

Conflicts of Interest: The authors declare that they have no competing interests.

Abbreviations

The following abbreviations are used in this manuscript:

FACT	Fact Automated Consistency Testing.
CredEye	A credibility lens for analyzing and explaining misinformation.
PSL	Probabilistic SOft Logic.
MAP	Markov Random Field.
HL-MRF	Hinge-Loss Markov Random Fields.
KB	Knowledge Base.
NLP	Natural Language Processing.
POS	Part Of Speech.
IE	Information Extraction.
ADMM	Alternating Direction Method of Multipliers.
GATE	General Architecture for Text Engineering.
ANNI	A Nearly-New IE system.
LHS	Lift Hand side in the grammar rule.
RHS	Right Hand side in the grammar rule.
ROC	Receiver operating characteristic.
AUC	Area Under the ROC Curve
TP	True Positive.
FP	False Negative.
FPR	False Positive Rate.
TPR	True Positive Rate.

Appendix A. List of logical rules

Here is a list of the logical rules we use in our PSL model:

Logical Rules
$Male(X) \Rightarrow \neg Female(X)$
$Parent\_Of(X,Y) \wedge X \neq Y \Rightarrow Ancestor\_Of(X,Y)$
$Ancestor\_Of(X,Y) \wedge Ancestor\_Of(Y,Z) \wedge Y \neq X \wedge Y \neq Z \Rightarrow Ancestor\_Of(X,Z)$
$Descendent\_Of(Y,X) \wedge Y \neq X \Rightarrow Ancestor\_Of(X,Y)$
$Parent\_Of(X,Y) \wedge Y \neq X \Rightarrow Descendent\_Of(Y,X)$
$Descendent\_Of(X,Y) \wedge Descendent\_Of(Y,Z) \wedge Y \neq X \wedge Y \neq Z \Rightarrow Descendent\_Of(X,Z)$
$Ancestor\_Of(X,Y) \wedge Y \neq X \Rightarrow Descendent\_Of(Y,X)$
$Parent\_Of(X,Y) \wedge Male(X) \wedge Y \neq X \Rightarrow Father\_Of(X,Y)$
$Parent\_Of(X,Y) \wedge Female(X) \wedge X \neq Y \Rightarrow Mother\_Of(X,Y)$
$Spouse\_Of(X,Y) \wedge Female(X) \wedge Y \neq X \Rightarrow Wife\_Of(X,Y)$
$Spouse\_Of(X,Y) \wedge Male(X) \wedge Y \neq X \Rightarrow Husband\_Of(X,Y)$
$Parent\_Of(X,Y) \wedge Y \neq X \Rightarrow Child\_Of(Y,X)$
$Parent\_Of(X,Y) \wedge Male(Y) \wedge Y \neq X \Rightarrow Son\_Of(Y,X)$
$Parent\_Of(X,Y) \wedge Female(Y) \wedge Y \neq X \Rightarrow Daughter\_Of(Y,X)$
$Parent\_Of(X,Y) \wedge Parent\_Of(X,Z) \wedge Y \neq Z \Rightarrow Sibling\_Of(Y,Z)$
$Sibling\_Of(Y,Z) \wedge Male(Y) \wedge (Y \neq Z) \Rightarrow Brother\_Of(Y,Z)$
$Sibling\_Of(Y,Z) \wedge Female(Y) \wedge (Y \neq Z) \Rightarrow Sister\_Of(Y,Z)$
$Parent\_Of(X,Z) \wedge Sibling\_Of(Y,X) \wedge Female(Y) \wedge X \neq Z \wedge Y \neq X \Rightarrow Aunt\_Of(Y,Z)$
$Uncle\_Of(X,Z) \wedge Spouse\_Of(Y,X) \wedge X \neq Z \wedge Y \neq X \Rightarrow AuntInLaw\_Of(Y,Z)$
$Aunt\_Of(X,Z) \wedge Spouse\_Of(Z,Y) \wedge X \neq Z \wedge Y \neq X \Rightarrow AuntInLaw\_Of(X,Y)$
$Parent\_Of(X,Z) \wedge Brother\_Of(Y,X) \wedge X \neq Z \wedge Y \neq X \Rightarrow Uncle\_Of(Y,Z)$
$Aunt\_Of(X,Z) \wedge Spouse\_Of(Y,X) \wedge X \neq Z \wedge Y \neq X \Rightarrow UncleInLaw\_Of(Y,Z)$
$Uncle\_Of(X,Z) \wedge Spouse\_Of(Z,Y) \wedge X \neq Z \wedge Y \neq X \Rightarrow UncleInLaw\_Of(X,Y)$
$Aunt\_Of(Z,X) \wedge Female(X) \wedge X \neq Z \Rightarrow Niece\_Of(X,Z)$
$Uncle\_Of(Z,X) \wedge Female(X) \wedge X \neq Z \Rightarrow Niece\_Of(X,Z)$
$UncleInLaw\_Of(Y,X) \wedge Female(X) \wedge X \neq Y \Rightarrow NieceInLaw\_Of(X,Y)$
$AuntInLaw\_Of(Y,X) \wedge Female(X) \wedge X \neq Y \Rightarrow NieceInLaw\_Of(X,Y)$

$$Uncle\_Of(Y,X) \wedge Male(X) \wedge X \neq Y \Rightarrow Nephew\_Of(X,Y)$$
$$Aunt\_Of(Y,X) \wedge Male(X) \wedge X \neq Y \Rightarrow Nephew\_Of(X,Y)$$
$$UncleInLaw\_Of(Y,X) \wedge Male(X) \wedge X \neq Y \Rightarrow NephewInLaw\_Of(X,Y)$$
$$AuntInLaw\_Of(Y,X) \wedge Male(X) \wedge X \neq Y \Rightarrow NephewInLaw\_Of(X,Y)$$
$$Parent\_Of(X,B) \wedge Parent\_Of(Z,A) \wedge A \neq B \wedge Sibling\_Of(X,Z) \wedge X \neq Z \Rightarrow Cousin\_Of(B,A)$$
$$Parent\_Of(X,B) \wedge Parent\_Of(Z,A) \wedge A \neq B \wedge SiblingInLaw\_Of(X,Z) \wedge X \neq Z \Rightarrow Cousin\_Of(B,A)$$
$$Cousin\_Of(Z,X) \wedge Spouse\_Of(X,Y) \wedge X \neq Z \wedge Y \neq X \Rightarrow CousinInLaw\_Of(Z,Y)$$
$$Parent\_Of(X,Z) \wedge Spouse\_Of(Y,Z) \wedge X \neq Z \wedge Y \neq Z \Rightarrow ChildInLaw\_Of(Y,X)$$
$$ChildInLaw\_Of(Y,X) \wedge Male(Y) \wedge Y \neq X \Rightarrow SonInLaw\_Of(Y,X)$$
$$ChildInLaw\_Of(Y,X) \wedge Female(Y) \wedge Y \neq X \Rightarrow DaughterInLaw\_Of(Y,X)$$
$$Spouse\_Of(X,Y) \wedge Parent\_Of(Z,X) \wedge Y \neq X \wedge Z \neq X \Rightarrow ParentInLaw\_Of(Z,Y)$$
$$ParentInLaw\_Of(Z,Y) \wedge Female(Z) \wedge Y \neq Z \Rightarrow MotherInLaw\_Of(Z,Y)$$
$$ParentInLaw\_Of(Z,Y) \wedge Male(Z) \wedge Y \neq Z \Rightarrow FatherInLaw\_Of(Z,Y)$$
$$Spouse\_Of(X,Y) \wedge Sibling\_Of(X,Z) \wedge Y \neq X \wedge X \neq Z \Rightarrow SiblingInLaw\_Of(Y,Z)$$
$$Spouse\_Of(X,Y) \wedge Sibling\_Of(Z,X) \wedge Y \neq X \wedge X \neq Z \Rightarrow SiblingInLaw\_Of(Z,Y)$$
$$SiblingInLaw\_Of(Z,Y) \wedge Female(Z) \wedge Z \neq Y \Rightarrow SisterInLaw\_Of(Z,Y)$$
$$SiblingInLaw\_Of(Z,Y) \wedge Male(Z) \wedge Z \neq Y \Rightarrow BrotherInLaw\_Of(Z,Y)$$
$$Parent\_Of(X,Y) \wedge Parent\_Of(Y,Z) \wedge X \neq Y \wedge Z \neq Y \Rightarrow GrandChild\_Of(Z,X)$$
$$GrandChild\_Of(Y,X) \wedge Male(Y) \wedge X \neq Y \Rightarrow Grandson\_Of(Y,X)$$
$$GrandChild\_Of(Y,X) \wedge Female(Y) \wedge X \neq Y \Rightarrow Granddaughter\_Of(Y,X)$$
$$Parent\_Of(X,Y) \wedge Parent\_Of(Y,A) \wedge X \neq Y \wedge A \neq Y \Rightarrow GrandParent\_Of(X,A)$$
$$GrandParent\_Of(X,Y) \wedge Male(X) \wedge X \neq Y \Rightarrow Grandfather\_Of(X,Y)$$
$$GrandParent\_Of(X,Y) \wedge Female(X) \wedge X \neq Y \Rightarrow Grandmother\_Of(X,Y)$$

Appendix B. Experiment Results

The table below shows the number of articles processed in each iteration and the number of relations extracted for both parent relation and spouse relation.

Search keywords	# articles	# parent relations	# spouse relations
"Diana Spencer" "Prince Edward"	134	18	0
"Prince William" "Prince Philip"	114	30	0
"Lady Sarah Chatto" "Mia Grace Tindall"	43	30	0
"Mia Grace Tindall" "Princess Eugenie"	77	38	4
"Sarah Fergie Ferguson" "Prince William"	100	46	4
"Princess Beatrice" "Prince Charles"	121	60	6
"Isla Elizabeth Phillips" "James"	111	74	13
"Elizabeth II" "Zara Phillips"	127	91	23
"Prince Philip" "Arthur Chatto"	111	103	27
"Prince Harry" "Prince William"	113	107	27
"Elizabeth Bowes-Lyon" "Elizabeth II"	144	122	30
"Princess Margaret" "Prince William"	121	128	32
"Autumn Phillips" "Princess Anne"	105	135	33
"Prince William" "Prince George"	101	135	33
"Prince William" "Prince Louis"	102	140	33
"Peter Phillips" "Sarah Fergie Ferguson"	85	150	38
"Prince Louis" "Princess Eugenie"	102	151	38
"Peter Phillips" "Princess Eugenie"	102	159	43
"Prince Charles" "David Armstrong-Jones"	141	175	50
"Diana Spencer" "Princess Anne"	125	191	50
"Zara Phillips" "Princess Beatrice"	111	206	57
"Arthur Chatto" "Princess Beatrice"	129	216	60
"Lady Sarah Chatto" "Princess Anne"	109	233	61
"Prince Harry" "Elizabeth II"	139	237	61
"Elizabeth II" "Diana Spencer"	127	247	62
"Samuel Chatto" "Lady Sarah Chatto"	133	249	71
"Arthur Chatto" "Princess Margaret"	110	251	73
"Autumn Phillips" "Prince Harry"	98	255	127
"Elizabeth II" "Antony Armstrong-Jones"	169	277	76
"Prince George" "Princess Beatrice"	109	282	76
"Prince Louis" "Prince Harry"	105	286	76
"Lady Sarah Chatto" "Lady Margarita Armstrong-Jones"	101	302	77
"Elizabeth II" "Prince Harry"	144	308	80
"Princess Margaret" "Princess Eugenie"	98	313	80
"Peter Phillips" "Prince Charles"	96	319	85
"Prince William" "Elizabeth II"	138	328	85
"Princess Charlotte" "Prince George"	117	332	85
"Princess Charlotte" "Prince Harry"	113	332	85
"Prince Charles" "Princess Anne"	131	340	85
"Isla Elizabeth Phillips" "Savannah Phillips"	111	346	90
"Prince Andrew" "Prince Charles"	108	348	90
"Princess Margaret" "Zara Phillips"	112	362	91
"Kate Middleton" "Prince William"	85	362	91
"Princess Anne" "Isla Elizabeth Phillips"	94	378	93
"Elizabeth II" "Prince Charles"	126	392	94
"Prince Harry" "Prince William"	117	394	94
"Princess Margaret" "Prince Charles"	117	397	96
"Prince Louis" "Lady Sarah Chatto"	93	419	99
"Princess Charlotte" "Prince Harry"	115	422	100
"David Armstrong-Jones" "Prince William"	139	437	101
"Prince George" "Prince Edward"	128	444	102
"Princess Beatrice" "Elizabeth II"	135	456	103
"Prince Edward" "Prince Charles"	125	468	103
"Elizabeth II" "Princess Margaret"	112	478	105
"Lady Sarah Chatto" "Charles Armstrong-Jones"	89	482	105
"Samuel Chatto" "Lady Sarah Chatto"	4137	488	112
"Lady Sarah Chatto" "Samuel Chatto"	130	488	112
"Arthur Chatto" "Prince Harry"	130	492	113
"Kate Middleton" "Prince William"	84	496	113
"Peter Phillips" "Zara Phillips"	129	500	113
"Zara Phillips" "Prince Charles"	85	508	115
"Prince Harry" "Princess Anne"	107	508	115
"Lady Louise Windsor" "Elizabeth II"	101	508	115
"Elizabeth II" "Princess Margaret"	113	516	117
"Elizabeth II" "Princess Anne"	132	520	117
"Princess Beatrice" "Peter Phillips"	108	530	121

"Princess Beatrice"	"Peter Phillips"	108	530	121
"Princess Anne"	"Elizabeth II"	140	532	121
"Meghan Markle"	"Lena Elizabeth Tindall"	95	534	123
"Prince Louis"	"Prince Charles"	109	534	123
"Lady Sarah Chatto"	"Samuel Chatto"	129	534	123
"Prince Louis"	"Prince Harry"	105	536	123
"Prince Edward"	"Peter Phillips"	115	544	124
"Savannah Phillips"	"Princess Anne"	100	547	125
"Prince Edward"	"Elizabeth II"	122	557	127
"Prince George"	"Prince Edward"	126	569	132
"Prince William"	"Prince Harry"	107	575	132
"Princess Eugenie"	"Princess Anne"	107	582	133
"Arthur Chatto"	"Lady Sarah Chatto"	127	582	133
"Lady Sarah Chatto"	"Princess Anne"	107	588	134
"Prince Harry"	"Isla Elizabeth Phillips"	125	590	139
"Lady Sarah Chatto"	"Princess Margaret"	110	594	139
"Prince Philip"	"Diana Spencer"	112	600	140
"Prince Charles"	"Elizabeth II"	124	600	141
"Prince Charles"	"Princess Beatrice"	128	603	141
"Princess Anne"	"Mia Grace Tindall"	63	603	145
"Diana Spencer"	"Elizabeth II"	138	617	153
"Prince Harry"	"Peter Phillips"	125	617	153
"Elizabeth II"	"Savannah Phillips"	118	629	167
"Cecilia Bowes-Lyon"	"Elizabeth Bowes-Lyon"	84	634	172
"Mia Grace Tindall"	"Lena Elizabeth Tindall"	83	640	172
"Isla Elizabeth Phillips"	"Prince William"	94	650	174
"Elizabeth II"	"Margaret Elphinstone"	105	660	175
"Arthur Chatto"	"David Armstrong-Jones"	94	668	178
"Princess Anne"	"Prince Andrew"	135	673	178
"Prince Louis"	"Prince Charles"	116	675	178
"Lady Sarah Chatto"	"Princess Margaret"	110	683	179
"Elizabeth Bowes-Lyon"	"Cecilia Bowes-Lyon"	83	714	181
"Princess Eugenie"	"Prince William"	106	718	181
"Prince William"	"Kate Middleton"	106	718	181
"Princess Beatrice"	"Prince Edward"	117	720	181
"Princess Beatrice"	"Prince Charles"	131	726	181
"Prince George"	"Prince Louis"	103	726	182
"Prince Andrew"	"Elizabeth II"	175	736	182
"Lady Louise Windsor"	"James"	98	738	182
"David Armstrong-Jones"	"Arthur Chatto"	99	740	182
"Prince William"	"Elizabeth II"	144	742	182
"Princess Anne"	"Lady Sarah Chatto"	95	748	182
"Prince Harry"	"Mia Grace Tindall"	68	749	184
"Zara Phillips"	"Peter Phillips"	130	751	184
"Lady Sarah Chatto"	"Elizabeth II"	120	751	184
"Daniel Chatto"	"David Armstrong-Jones"	92	755	184
"Mike Tindall"	"Peter Phillips"	107	759	184
"Princess Beatrice"	"Elizabeth II"	141	759	186
"Prince Harry"	"Kate Middleton"	102	759	186
"Prince Harry"	"Princess Eugenie"	114	761	186
"Lady Margarita Armstrong-Jones"	"Charles Armstrong-Jones"	116	763	187
"Prince Charles"	"Prince William"	128	769	188
"Peter Phillips"	"Princess Anne"	126	773	189
"Lady Margarita Armstrong-Jones"	"Charles Armstrong-Jones"	113	781	192
"Prince Edward"	"Prince William"	115	789	200
"James"	"Prince Harry"	144	795	200
"Princess Beatrice"	"Isla Elizabeth Phillips"	93	795	202
"Prince Louis"	"Meghan Markle"	106	799	203
"Capt Mark Phillips"	"Elizabeth II"	118	803	203
"Prince Charles"	"Prince William"	126	807	203
"Elizabeth II"	"Sarah Fergie Ferguson"	114	807	203
"David Armstrong-Jones"	"Serena Armstrong-Jones"	98	802	204
"Peter Phillips"	"Elizabeth II"	124	815	204
"Antony Armstrong-Jones"	"Elizabeth II"	155	827	204
"Prince Charles"	"Sarah Fergie Ferguson"	129	832	205
"Princess Beatrice"	"Princess Eugenie"	92	837	205
"Princess Anne"	"Prince Charles"	117	844	205
"Peter Phillips"	"Prince William"	111	847	205
"Prince Edward"	"Prince William"	117	847	206
"Elizabeth II"	"Princess Eugenie"	126	851	206
"Prince William"	"Elizabeth II"	135	855	207

"Capt Mark Phillips" "Prince William"	103	860	207
"Prince Harry" "Prince Charles"	117	860	207
"Kate Middleton" "Prince Harry"	85	860	208
"Mary Elphinstone" "Cecilia Bowes-Lyon"	79	866	210
"Prince Philip" "Prince William"	121	866	211
"Antony Armstrong-Jones" "Princess Margaret"	147	888	212
"Prince George" "Prince William"	113	890	212
"Peter Phillips" "Savannah Phillips"	120	894	213
"Prince Edward" "Diana Spencer"	133	902	214
"Prince William" "Princess Anne"	126	908	214
"Princess Anne" "Prince William"	129	914	214
"Elizabeth II" "Prince William"	128	914	214
"Prince Louis" "Prince William"	113	914	215
"Arthur Chatto" "Charles Armstrong-Jones"	86	920	216
"Savannah Phillips" "Prince William"	126	920	216
"Prince Philip" "Prince Harry"	111	926	216
"Prince Philip" "Prince Harry"	111	926	216
"Meghan Markle" "Prince William"	88	926	216
"Prince Charles" "Prince Andrew"	119	928	216
"Capt Mark Phillips" "Prince Charles"	123	938	221
"Princess Eugenie" "Prince Charles"	121	939	221
"Princess Margaret" "Princess Anne"	125	941	221
"Princess Eugenie" "Elizabeth II"	129	943	223
"Diana Spencer" "Prince Andrew"	118	951	224
"Prince Harry" "Princess Anne"	118	955	224
"Prince Louis" "Isla Elizabeth Phillips"	90	959	227
"Isla Elizabeth Phillips" "Princess Beatrice"	92	963	227
"Princess Anne" "Savannah Phillips"	109	963	230
"Prince Harry" "Princess Beatrice"	95	964	231
"Elizabeth II" "Prince Philip"	130	965	231
"Savannah Phillips" "Isla Elizabeth Phillips"	108	967	232
"Prince George" "Lady Sarah Chatto"	111	971	232
"Princess Beatrice" "Princess Eugenie"	99	973	233
"Prince Charles" "Zara Phillips"	89	977	233
"Prince George" "Princess Charlotte"	98	977	233
"Autumn Phillips" "Peter Phillips"	131	977	233
"Isla Elizabeth Phillips" "Savannah Phillips"	112	977	233
"Sarah Fergie Ferguson" "Peter Phillips"	102	995	234
"Princess Beatrice" "Prince William"	97	995	234
"Prince William" "Princess Anne"	135	1001	235
"Lady Sarah Chatto" "Charles Armstrong-Jones"	98	1003	236
"Princess Beatrice" "Princess Eugenie"	98	1003	237
"Serena Armstrong-Jones" "Lady Sarah Chatto"	78	1005	237
"Prince Charles" "Princess Eugenie"	130	1017	237
"Lady Sarah Chatto" "David Armstrong-Jones"	98	1017	237
"Princess Charlotte" "Meghan Markle"	124	1017	237
"Prince Harry" "Zara Phillips"	111	1017	238
"Prince William" "Princess Beatrice"	97	1019	238
"David Armstrong-Jones" "Arthur Chatto"	100	1019	239
"Mary Elphinstone" "Elizabeth II"	125	1035	244
"Prince Andrew" "Princess Anne"	121	1035	244
"Princess Anne" "Kate Middleton"	87	1037	245
"Princess Anne" "Elizabeth II"	135	1043	245
"Meghan Markle" "Princess Charlotte"	116	1049	246
"Princess Anne" "Zara Phillips"	108	1051	247
"Princess Eugenie" "Princess Beatrice"	96	1051	247
"Prince Louis" "Elizabeth II"	126	1055	148
"Prince Philip" "Princess Margaret"	111	1065	253
"David Armstrong-Jones" "Elizabeth II"	115	1069	253
"Peter Phillips" "Kate Middleton"	114	1069	255
"Princess Margaret" "Daniel Chatto"	146	1069	255

References

1. Ba, M.L.; Berti-Equille, L.; Shah, K.; Hammady, H.M. Vera: A platform for veracity estimation over web data. In Proceedings of the 25th International Conference Companion on World Wide Web. International World Wide Web Conferences Steering Committee, Montreal QC, Canada, 11 April 2016; pp. 159–162.
2. Hassan, N.; Adair, B.; Hamilton, J.T.; Li, C.; Tremayne, M.; Yang, J.; Yu, C. The quest to automate fact-checking. In Proceedings of the 2015 Computation+ Journalism Symposium, New York, NY, USA, 2–3 October 2015.

3. Hassan, N.; Zhang, G.; Arslan, F.; Caraballo, J.; Jimenez, D.; Gawsane, S.; Hasan, S.; Joseph, M.; Kulkarni, A.; Nayak, A.K.; others. ClaimBuster: the first-ever end-to-end fact-checking system. *Proc. Vldb Endow.* **2017**, *10*, 1945–1948.
4. Wu, Y.; Walenz, B.; Li, P.; Shim, A.; Sonmez, E.; Agarwal, P.K.; Li, C.; Yang, J.; Yu, C. iCheck: Computationally combating lies, d–ned lies, and statistics. In Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data. ACM, Snowbird, UT, USA, 22–27 June 2014; pp. 1063–1066.
5. Rashkin, H.; Choi, E.; Jang, J.Y.; Volkova, S.; Choi, Y. Truth of varying shades: Analyzing language in fake news and political fact-checking. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; pp. 2931–2937.
6. Nakashole, N.; Mitchell, T.M. Language-aware truth assessment of fact candidates. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore, MD, USA, 23–25 June 2014; Volume 1, pp. 1009–1019.
7. Bach, S.H.; Broecheler, M.; Huang, B.; Getoor, L. Hinge-loss markov random fields and probabilistic soft logic. *J. Machine Learn. Res.* **2017**, *18*, 3846–3912.
8. Bindris, N.; Sudhahar, S.; Cristianini, N. Fact Checking from Natural Text with Probabilistic Soft Logic. In *Advances in Intelligent Data Analysis XVII*; Duivesteijn, W., Siebes, A., Ukkonen, A., Eds.; Springer: Cham, Switzerland, 2018; pp. 52–61.
9. Carlson, A.; Betteridge, J.; Kisiel, B.; Settles, B., Jr.; Mitchell, T.M. Toward an Architecture for Never-Ending Language Learning. In Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010), Atlanta, GA, USA, 11–15 July 2010.
10. Soleimani, A.; Monz, C.; Worring, M. Bert for evidence retrieval and claim verification. In Proceedings of the European Conference on Information Retrieval, Lisbon, Portugal, 14–17 April 2020, pp. 359–366.
11. Ciampaglia, G.L.; Shiralkar, P.; Rocha, L.M.; Bollen, J.; Menczer, F.; Flammini, A. Computational fact checking from knowledge networks. *PloS ONE* **2015**, *10*, e0128193.
12. Thorne, J.; Vlachos, A. Automated Fact Checking: Task formulations, methods and future directions. *arXiv* **2018**, arXiv:1806.07687.
13. Mihaylova, T.; Nakov, P.; Marquez, L.; Barron-Cedeno, A.; Mohtarami, M.; Karadzhov, G.; Glass, J. Fact checking in community forums. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
14. Mohtarami, M.; Baly, R.; Glass, J.; Nakov, P.; Màrquez, L.; Moschitti, A. Automatic Stance Detection Using End-to-End Memory Networks. *arXiv* **2018**, arXiv:1804.07581.
15. Vlachos, A.; Riedel, S. Fact Checking: Task definition and dataset construction. In Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science, Baltimore, MD, USA, 26 June 2014; pp. 18–22.
16. Thorne, J.; Chen, M.; Myrianthous, G.; Pu, J.; Wang, X.; Vlachos, A. Fake news stance detection using stacked ensemble of classifiers. In Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism, Copenhagen, Denmark, 7 September 2017; pp. 80–83.
17. Popat, K.; Mukherjee, S.; Strötgen, J.; Weikum, G. CredEye: A credibility lens for analyzing and explaining misinformation. In Proceedings of the The Web Conference 2018, Lyon, France, 23–27 April 2018; pp. 155–158.
18. Thorne, J.; Vlachos, A. An extensible framework for verification of numerical claims. In Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of theAssociation for Computational Linguistics. Association for Computational Linguistics, Valencia, Spain, 3–7 April 2017; pp. 37–40.
19. Hassan, N.; Arslan, F.; Li, C.; Tremayne, M. Toward automated fact-checking: Detecting check-worthy factual claims by ClaimBuster. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Halifax, NS, Canada, 13–17 August 2017; pp. 1803–1812.
20. Karadzhov, G.; Nakov, P.; Màrquez, L.; Barron-Cedeno, A.; Koychev, I. Fully automated fact checking using external sources. *arXiv* **2017**, arXiv:1710.00341.
21. Baly, R.; Mohtarami, M.; Glass, J.; Màrquez, L.; Moschitti, A.; Nakov, P. Integrating stance detection and fact checking in a unified corpus. *arXiv* **2018**, arXiv:1804.08012.
22. Shi, B.; Weninger, T. Fact checking in heterogeneous information networks. In Proceedings of the 25th International Conference Companion on World Wide Web, Montreal, QC, Canada, 11 April 2016; pp. 101–102.

23. Popat, K.; Mukherjee, S.; Strötgen, J.; Weikum, G. Credibility assessment of textual claims on the web. In Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, Indianapolis, IN, USA, 24–28 October 2016; pp. 2173–2178.
24. Popat, K.; Mukherjee, S.; Strötgen, J.; Weikum, G. Where the truth lies: Explaining the credibility of emerging claims on the web and social media. In Proceedings of the 26th International Conference on World Wide Web Companion. International World Wide Web Conferences Steering Committee, Perth, Australia, 29 April 2017; pp. 1003–1012.
25. Kimmig, A.; Bach, S.; Broecheler, M.; Huang, B.; Getoor, L. A short introduction to probabilistic soft logic. In Proceedings of the NIPS Workshop on Probabilistic Programming: Foundations and Applications, Harrahs and Harveys, Lake Tahoe, CA, USA, 8 December 2012; pp. 1–4.
26. Richardson, M.; Domingos, P. Markov logic networks. *Mach. learn.* **2006**, *62*, 107–136.
27. Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; Eckstein, J.; others. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*; Now Publishers Inc: Norwell, MA, USA, 2011; pp. 1–122.
28. Thakker, D.; Osman, T.; Lakin, P. *GATE JAPE Grammar Tutorial*; Nottingham Trent University, UK: Phil Lakin, UK, 2009.
29. Soon, W.M.; Ng, H.T.; Lim, D.C.Y. A machine learning approach to coreference resolution of noun phrases. *Comput. Linguist.* **2001**, *27*, 521–544.
30. Cunningham, H.; Maynard, D.; Bontcheva, K.; Tablan, V.; Aswani, N.; Roberts, I.; Gorrell, G.; Funk, A.; Roberts, A.; Damljanovic, D. *Developing Language Processing Components With GATE Version 6 (A User Guide)*; University of Sheffield: Sheffield, UK, 2011.
31. Benatallah, B.; Venugopal, S.; Ryu, S.H.; Motahari-Nezhad, H.R.; Wang, W.; others. A systematic review and comparative analysis of cross-document coreference resolution methods and tools. *Computing* **2017**, *99*, 313–349.
32. Fleischman, M.; Hovy, E. Multi-document person name resolution. In Proceedings of the Conference on Reference Resolution and Its Applications, Barcelona, Spain, 21–26 July 2004; pp. 1–8.
33. Mayfield, J.; Alexander, D.; Dorr, B.J.; Eisner, J.; Elsayed, T.; Finin, T.; Fink, C.; Freedman, M.; Garera, N.; McNamee, P.; others. Cross-Document Coreference Resolution: A Key Technology for Learning by Reading. In Proceedings of the AAAI Spring Symposium: Learning by Reading and Learning to Read, Palo Alto, California, 23–25 March 2009; Volume 9, pp. 65–70.
34. Ji, H.; Grishman, R.; Chen, Z.; Gupta, P. Cross-document event extraction and tracking: Task, evaluation, techniques and challenges. In Proceedings of the International Conference RANLP-2009, Borovets, Bulgaria, 14–16 September 2009; pp. 166–172.
35. Dutta, S.; Weikum, G. Cross-document co-reference resolution using sample-based clustering with knowledge enrichment. *Transactions of the Association for Computational Linguistics*; MIT Press: Cambridge, MA, USA, 2015; pp. 15–28.
36. Hoffart, J.; Yosef, M.A.; Bordino, I.; Fürstenau, H.; Pinkal, M.; Spaniol, M.; Taneva, B.; Thater, S.; Weikum, G. Robust disambiguation of named entities in text. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Edinburgh, UK, 27–29 July 2011; pp. 782–792.
37. Lewis, D.D. Evaluating text categorization. In Proceedings of the Speech and Natural Language, Pacific Grove, CL, USA, 19–22 February 1991.
38. Lewis, D.D. Representation quality in text classification: An introduction and experiment. In Proceedings of the Speech and Natural Language, Hidden Valley, PA, USA, 24–27 June 1990.