18th International Conference on Information Fusion
Washington, DC - July 6-9, 2015

Markov Logic Networks for Multi-Level Fusion Support to Intelligence Analysis

Martin Oxenham and Zhuoyun Ao
Joint and Operations Analysis Division
Defence Science and Technology Organisation
Edinburgh, South Australia, Australia, 5111
Email: martin.oxenham@dsto.defence.gov.au
Email: zhuoyun.ao@dsto.defence.gov.au

Glenn Burgess and Edwin El-Mahassni
National Security and ISR Division
Defence Science and Technology Organisation
Edinburgh, South Australia, Australia, 5111
Email: glenn.burgess@dsto.defence.gov.au
Email: edwin.el-mahassni@dsto.defence.gov.au

***Abstract*—Intelligence analysts are faced with a complex information fusion problem that is characterised by the volume, velocity, variety and veracity of the observed data and information. That is, there are large quantities of data and information (volume) arriving at a high rate (velocity) which are, in general, highly heterogeneous (variety) and of inconsistent fidelity (veracity). Making sound decisions based on these observations is the aim and human decision making is essential, since humans are ultimately held responsible. Current industry tools are designed to support this process, by helping to process and curate the collection of observations automatically, storing it in databases and providing various analytical tools to retrieve and manipulate fragments of the collection. However, industry approaches for managing the inherent uncertainty in these observations and exploiting all the available higher level contextual information are inadequate. What is needed is a practical formalism that can deal with multiple types of uncertainty, can exploit contextual information and can operate at scale to reduce the cognitive burden on analysts. In this paper, we discuss the use of Markov logic networks for handling uncertainty and exploiting higher level contextual information and demonstrate how this provides a framework which is well suited to handling the real-world issues encountered in intelligence-based problems.**

I. Introduction

Intelligence analysis may be defined as "the application of individual and collective cognitive methods to weigh data and test hypotheses within a secret socio-cultural context" [1, p. 4]. Regarded as a process, it may also be viewed as one of exploiting observed data and information to support the establishment of situation awareness about a scenario and entities within the scenario, in a context in which deception is the rule, not the exception. Making sound decisions based on these data and information is the aim and human decision making is essential, since humans are ultimately held responsible. Minton et al [2] illustrate the nature of such an undertaking. They describe the real-life case of a Russian businessman who was the owner of several air cargo companies that were alleged to to have facilitated illicit trafficking of contraband. They state [2]:

> *"In tracking a suspected arms dealer such as [name supplied], intelligence analysts would collect as much information as possible about the air cargo companies he is associated with, the planes used by those companies, the plane's capabilities, known routes that they have flown, reported observations of those planes, and other people, organizations and companies that [he] deals with or communicates with. In addition, analysts would attempt to infer any aliases or front companies he uses, and document any suspicious and/or deceptive behavior.*
>
> *Deceptive behaviors come in many forms. One common behavior which [he] allegedly engaged in is transferring aircraft between multiple companies. Doing so makes tracking more difficult, since an aircraft can be re-registered with a new tail number when it is transferred."*

They then go on to list a number of open source intelligence (OSINT) resources that would assist in such an analysis including sites with millions of pictures of aircraft at airports around the world, aviation safety databases, databases with public records about aircraft and air transport companies, as well as news, blogs and internet fora.

While intelligence comes in many more forms than just OSINT, the example in this extract does highlight the complex fusion problem facing intelligence analysts which is characterised by the 4 V's of big data. There are large quantities of data (*volume*) arriving at a high rate (*velocity*) which, in general, are highly heterogeneous (*variety*) and of inconsistent fidelity (*veracity*). Current industry tools are designed to assist in dealing with this problem by helping to automatically process and curate the collection of observed data and information, storing them in databases, and providing various analytical tools to retrieve and manipulate relevant fragments from the collection. However, existing industry approaches have no means of handling the inherent uncertainty in these observations or of exploiting the available higher level contextual information which may influence and inform the intelligence analysis; these tasks are still left to the analysts. What is needed is a practical formalism that can deal with multiple types of uncertainty, can exploit contextual information, can operate at scale and can reduce the cognitive burden on analysts.

Employing statistical data mining approaches can provide useful insights for some problems due to the volume and velocity of the observations, but other problems are more like

finding a needle in a haystack. That is, the problem is to find a few pieces of information amongst the vast deluge that connect together to allow an analyst to draw conclusions about a particular topic, threat or system of interest. The two parts to this problem are then (i) finding the important pieces of information and (ii) assessing the strength of the conclusions. Automated methods would be useful for some parts of this problem, but in order to develop automated methods robust manual methods must first be developed to handle some of the preprocessing of the observations.

Dealing with the variety and veracity of the observations is a technical challenge that falls under the general banner of what is often called *data cleaning*. For example, information about locations can be specified in a wide variety of ways, such as by name, coordinates (a point), a region defined by a set of coordinates, or as a position or region relative to another location (e.g. 20 km North of town X). Making these observations accessible for reasoning and querying regardless of their input format and level of fidelity is one of our aims.

Given a number of observations of some real-world variable, which may differ in fidelity and uncertainty, *observation fusion* is the process of choosing the most likely fused value for that variable. Even better is to consider all probable values of the variable for downstream processing, such as in *entity resolution* where the aim is to determine if two distinct entities in a database actually represent the same entity.

While data cleaning, observation fusion and entity resolution have been described as separate processes, in intelligence applications entity resolution cannot be disentangled from data cleaning and observation fusion. Thus, while the discussion in the remainder of the paper is skewed towards entity resolution, our intention is to present an approach based on Markov logic networks which has the potential to address all three problems together in a homogeneous way, and identify what problems arise along the way.

The rest of the paper is structured as follows. In Section II a brief history and overview of entity resolution is described. Section III commences with an overview of the principal concepts from first-order logic which underpin Markov logic, and proceeds with an outline of the theory of Markov logic networks. Section IV then steps through a small-scale worked example of entity resolution in the intelligence domain to illustrate how Markov logic can be applied to this problem. Section V makes some observations about the approach and describes some of the lessons learnt. Some final thoughts are then presented in the conclusion.

II. Entity Resolution

As defined in the introduction, entity resolution refers to the problem of determining if two distinct entities in the same database or different databases actually represent the same entity. Historically, the notion of entity resolution arose in the context of the removal of equivalent references between two lists of attributes, where the lists were assumed to have different structures. The earliest acknowledged reference of the need for such a process, which was first known as *record linkage*, but which in more modern parlance could be regarded as data association between databases, was Dunn [3] in 1946 who raised the issue in the context of compiling a 'Book of Life' for human subjects whose vital records were scattered across different files and registers. However, it was Newcombe et al. [4] in 1959 who proposed the use of computers to automate record linkage. The first algorithm for achieving this automation was proposed by Fellegi and Suntner [5] in 1969. Based on a naive Bayesian model [6, p. 572], the Fellegi-Suntner algorithm has remained influential since then, with a number of other algorithms being based on it [6, p. 577]. Most of these algorithms function on a pairwise basis, producing a list of matched entity pairs. In principle, for any three entities e_1, e_2, e_3, if such an algorithm matches e_1 with e_2 and e_2 with e_3, then it should also match e_1 with e_3. However, this ideal is often violated when performing pairwise matching. As such, a separate step is then performed to remove such inconsistencies; this is referred to as computing the *transitive closure*.

In 2006, Singla and Domingos [6] introduced a markedly different approach to entity resolution. It was based on the concept of Markov logic networks that Domingos had pioneered several years before for combining first order logic and probability theory. What distinguished their approach was that it incorporated (i) the use of softened first-order logic formulae to represent conditions under which two entities could be considered identical and (ii) constraints based on the axioms for equivalence classes, which ensured that transitive inconsistencies could not arise, thus avoiding the need to compute the transitive closure after the fact. They demonstrated this approach to the problem of 'de-duplicating' different citations of academic papers to determine which citations referred to the same paper. In the sequel, we demonstrate how their general approach can be extended to incorporate both uncertainty in observations and contextual information for entity resolution (integrated with some data cleaning and observation fusion) in the intelligence domain, by developing an appropriate Markov logic knowledge base and a representative set of entity-based evidence to reason over using this knowledge base.

III. First Order Logic and Markov Logic Networks

The concept of Markov logic, which integrates first order logic with probabilistic graphical models known as *Markov networks*, was developed about a decade ago by Domingos [7, Ch. 12], [8] in a bid to exploit the best of both formal first order theories and probability theory, and to create an 'interface layer' for artificial intelligence which provides an economical means of representing constraints that must be met in all the worlds for a given application domain. A full account of this elegant theory would require a discussion of Markov networks and first order logic. However, regarding Markov networks, which are also sometimes referred to as *Markov random fields*, for the purposes of this paper it is sufficient to understand that they form a special class of probabilistic graphical models based on undirected graphs, which induce

so-called Markov (conditional independence) properties on the joint probability distributions which are defined over them (refer to Refs. [9], [10] and [11, Ch. 2] for detailed accounts). As such, only the requisite background on first-order logic will be presented here before Markov logic networks are defined.

A. First Order Logic Descriptions and Terminology

Regarded as a formal language, a first-order logic (FOL) [12] is a triple $L = (V, T, W)$, where V, T and W are defined as follows:

V is the *vocabulary*, consisting of logical symbols including:

- A countable collection of *variables* $x, y, z, \ldots$
- The *Boolean connectives* $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$ which are read *not*, *and*, *or*, *(material) implication* and *equivalence* respectively
- The *universal* and *existential quantifiers* $\forall$, $\exists$ which are read *for all* and *there exists* respectively
- Brackets "(" and ")"
- The *equality predicate* $=$

and non-logical symbols including (possibly empty) sets of symbols for *constants*, *predicates* and *functions*.

T is a collection of *terms*, each of which is either a constant, a variable over V, or a function $f(t_1, t_2, \ldots, t_i)$, where f is a function symbol over V and $t_1, t_2, \ldots, t_i$ are terms over V.

W is a collection of *well-formed formulae* over V, where the well-formed formulae *wff* (or simply *formulae* for short) are defined constructively according to the following criteria:

- *Atomic formulae* where, by definition, an atomic formula has the form $p(t_1, t_2, \ldots, t_i)$, where p is a predicate symbol, and $t_1, t_2, \ldots, t_i$ are terms
- If p and q are *wff* then so are $\neg p$, $(p \vee q)$, $(p \wedge q)$, $(p \rightarrow q)$ and $(p \leftrightarrow q)$
- If p is a *wff* and x is a variable, then both $\exists x\, p$ and $\forall x\, p$ are *wff*
- Nothing else is a *wff*.

□

Given a FOL, a *knowledge base* comprises a set of FOL formulae over a domain of interest. The constant symbols in V represent the objects in the domain of interest and the variables in V range over these objects. An *interpretation* specifies which objects, functions and relations in the domain are represented by which symbols [8, p. 9].

Three other main concepts of FOL are important for Markov logic networks, namely the *conjunctive normal form*, *grounding* and *possible worlds*.

Conjunctive Normal Form: Given a FOL, the *literals* of the FOL are the atomic formulae (*positive* literals) and their negations (*negative* literals). Assuming the FOL has a finite number n of atomic formulae, a *disjunct*[1] consists of the disjunction of n distinct *wff*, each of which is a literal. It is noted that, for FOLs with finite numbers of atomic formulae, every *wff* which is expressed in terms of the defined FOL Boolean connectives can be rewritten in so-called *conjunctive normal form (CNF)* which consists of the conjunction of distinct *disjuncts*. Furthermore, this CNF representation is unique up to the order of the disjuncts and the order of the literals in the disjuncts (this follows from corresponding results in Ref. [14, pp. 146-147] about the counterpart to the CNF known as the *disjunctive normal form* which is the disjunction of conjuncts). Descriptions of the means for converting FOL formulae into conjunctive normal form are given in Refs. [15, pp. 281-282] and [16].

[1]Also sometimes referred to as a *maxterm* (refer to Ref. [13, p. 494]).

Grounding: A *ground term* is a term containing no variables. A *ground atom* or *ground predicate* is an atomic formula, the arguments of which are all ground terms. Given a term, the process of grounding it refers to the substitution (instantiation) of each of its variables by a constant in the domain of the variable [8, p. 9].

Possible Worlds: Given a knowledge base and an interpretation over it, a *possible world* refers to the assignment of a binary truth value to each possible ground predicate. A formula in the knowledge base is said to be *satisfiable* if and only if there exists at least one possible world in which it is true [8, p. 9].

B. Markov Logic Networks

Given a traditional first-order knowledge base, it can be viewed as a set of hard constraints on the set of possible worlds, such that if a world violates even one formula, the probability of that world is zero. Often such hard constraints are not suitable for modelling the real world because almost invariably there will be some situation in which a given formula will not hold. It is therefore desirable to introduce a means of softening these constraints in some way. In a Markov logic network, this is achieved by attaching a weight to each formula in the knowledge base. A formula's weight reflects how strongly it imposes the constraint on possible worlds; the higher the weight, the lower the probability that there exists a possible world that violates it. More formally, a Markov logic network may be defined as follows:

A *Markov logic network (MLN)* [8, p. 12] L is a set of m pairs (F_i, w_i) where F_i is a formula in first-order logic and w_i is a real number. Together with a finite set of constants $C = \{c_1, c_2, \ldots, c_{|C|}\}$, it defines a (*ground*) Markov network $M_{L,C}$ as follows:

1) $M_{L,C}$ contains one binary node for each possible grounding of each predicate appearing in L. The value of the node is 1 if the ground predicate is true and is 0 otherwise.
2) $M_{L,C}$ contains one feature for each possible grounding of each formula F_i in L. The value of this feature is 1 if the ground formula is true and is 0 otherwise. The weight of the feature is the weight w_i associated with F_i in L.

The joint probability distribution over the set of k binary truth-valued ground predicates associated with this ground Markov network takes the following form. Representing the jth binary truth-valued ground predicate by a variable x_j and denoting by

$\boldsymbol{x}$ the random vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_k)$, the probability that $\boldsymbol{x} = \boldsymbol{X}$, where $\boldsymbol{X} = (X_1, X_2, \ldots, X_k)$ is a possible world, is given by[2]

$$P(\boldsymbol{x} = \boldsymbol{X}) = \frac{1}{Z} \exp\left(\sum_i^m w_i n_i(\boldsymbol{X})\right) \quad (1)$$

where $n_i(\boldsymbol{X})$ is the number of true groundings of F_i in the world $\boldsymbol{X}$ and Z is a scaling term known as the *partition function* which has the form

$$Z = \sum_{\boldsymbol{X}} \exp(\sum_i w_i n_i(\boldsymbol{X})). \quad (2)$$

It is noted that as the weight for a given formula approaches infinity, the probability of the formula approaches 1 if the formula is true, while it approaches 0 if the formula is false. Consequently, each weighted formula in a Markov logic network converges to the hard (ie strict FOL) version of the formula as the weight approaches infinity. It is further noted that if a formula F_i has a weight w_i, then the weight of the negation $\neg F_i$ of F_i is $-w_i$. Finally, a formula with a weight of 0 essentially contributes nothing to the overall joint distribution since $w_i n_i(\boldsymbol{X}) = 0$ for all possible worlds $\boldsymbol{X}$. As such, it represents a state of ignorance about the validity of the formula.

Inference in a Markov logic network takes two basic forms. Given a ground Markov network and evidence in the form of truth value assignments for a subset of the ground (*evidence*) predicates, the marginal probabilities of the remaining (*query*) predicates can be calculated. Alternatively, the most likely possible world or most probable explanation (MPE) can also be calculated. In practice, however, it is generally infeasible to calculate these quantities exactly, so it is necessary to estimate them instead. In the case of marginal probabilities, several algorithms based on Markov chain Monte Carlo (MCMC) techniques have been developed to estimate them, with the so-called MC-SAT algorithm being the most commonly used [8, Chap. 3], [17]. In contrast, to calculate the most likely possible world or MPE, which is interpreted as the possible world at which the maximum *a posteriori* probablility (MAP) of the joint distribution given the evidence is attained, a randomised satisfiability solver known as the MaxWalkSAT algorithm is typically employed [8, Chap. 3], [17].

IV. Entity Resolution in Markov Logic Networks

Having defined MLNs, we are now in a position to demonstrate how they may be applied to entity resolution for intelligence applications.

[2]In defining the probability in this way, there are several assumptions being made which ensure that the set of possible worlds is finite and that $M_{L,C}$ represents a unique, well-defined probability distribution. These are that (i) different constants refer to different entities (unless this is inferred by formulae in the MLN), (ii) the only objects in the domain are those representable using the constant and function symbols in C and L, and (iii) for every function appearing in L, the value of that function applied to every possible tuple of arguments is known and is an element of C [8, pp. 13-14].

A. MLN Worked Example

For illustration, we consider a situation where a Mystery Person is to be added to an existing knowledge base containing persons of interest and contextual information about them. This contextual information includes attributes of people, relations between people and relations between other entities in the knowledge base. Conceptually, whenever a new entity is to be added to the knowledge base (or indeed any new information) we want to determine if this entity is already represented in the knowledge base. For example, a person with the same surname, who was born in the same town on the same date is likely to be the same person, but not if they are siblings.

The representation of real data is often inconsistent, which makes comparison difficult, and relevant data is often sparse. In this worked example we demonstrate how MLNs can handle these kinds of comparisons, and also handle uncertainty. We then show how contextual information can be taken into account. Finally we combine these components into a global comparison of entity similarity.

The example is evaluated using *Alchemy* [18], which is the reference implementation of MLNs developed by Kok et al. at the University of Washington.

1) Inference Over Tree-Structured Data: One important aspect of spatial fidelity is the notion of containment. Suppose we have biographical data represented at varying levels of abstraction, such as in Table I, with the spatial containment relations shown in Table II.

TABLE I
Representative Input Data for Attribute *born in*

Person	Relation	Property	Alchemy Input
Mystery Person	*born in*	Brisbane	Born_in(Mystery_Person, Brisbane)
Bob	*born in*	Adelaide	Born_in(Bob, Adelaide)
Alice	*born in*	Kangaroo Point	Born_in(Alice, Kangaroo_Pt)
Germaine	*born in*	Australia	Born_in(Germaine, Australia)

TABLE II
Representative Input Data for Attribute *contains*

Location	Relation	Location	Alchemy Input
Australia	*contains*	Queensland	Contains_loc(Australia, Queensland)
Australia	*contains*	Victoria	Contains_loc(Australia, Victoria)
Australia	*contains*	South Australia	Contains_loc(Australia, South_Australia)
Brisbane	*contains*	Kangaroo Point	Contains_loc(Brisbane, Kangaroo_Pt)
Queensland	*contains*	Cairns	Contains_loc(Queensland, Cairns)
Queensland	*contains*	SE Queensland	Contains_loc(Queensland, SE_Queensland)
SE Queensland	*contains*	Brisbane	Contains_loc(SE_Queensland, Brisbane)
South Australia	*contains*	Barossa Valley	Contains_loc(South_Australia, Barossa_Valley)
Adelaide	*contains*	Magill	Contains_loc(Adelaide, Magill)
Victoria	*contains*	Melbourne	Contains_loc(Victoria, Melbourne)

We would like to be able to program the system to 'understand' the spatial relationships between different locations so that a query like "Who was born in Queensland?" would return all people born in Queensland or its subregions (ie. Alice and Mystery Person). In a logical programming language this can be achieved by specifying the properties of the containment relation, and a set of facts outlining the hierarchy of containment. The basis of this problem is to determine the

probability that any two described entities are in fact referring to the same person. Thus, the relation *same_person* is also defined.

TABLE III
LOGICAL FORMULAE AND CORRESPONDING MLN RULES FOR *contains* AND *born_in*

Formula	Encoding	Syntax
1	**Logical Formula**	$\forall x, y, z\ contains(x, y) \wedge\ contains(y, z) \rightarrow contains(x, z)$
	Alchemy Input	Contains_loc(a,b) ^ Contains_loc(b,c) => Contains_loc(a,c).
2	**Logical Formula**	$\forall x, y\ contains(x, y) \rightarrow \neg contains(y, x)$
	Alchemy Input	Contains_loc(a,b) => !Contains_loc(b,a).
3	**Logical Formula**	
	Alchemy Input	8 !Contains_loc(a,b)
4	**Logical Formula**	$\forall p, l1, l2\ born\_in(p, l2) \wedge contains(l1, l2) \rightarrow born\_in(p, l1)$
	Alchemy Input	Born_in(p,l2) ^ Contains_loc(l1,l2) => Born_in(p,l1).
5	**Logical Formula**	$\forall p, l1, l2, l3\ born\_in(p, l2) \wedge\ contains(l1, l2) \wedge contains(l1, l3) \wedge\ l2 \neq l3 \wedge \neg contains(l2, l3)\ \wedge\ \neg contains(l3, l2) \rightarrow \neg born\_in(p, l3)$
	Alchemy Input	Born_in(p,l2) ^ Contains_loc(l1,l2) ^ Contains_loc(l1,l3) ^ !(l2=l3) ^ !Contains_loc(loc2, loc3) ^ !Contains_loc(loc3, loc2) => !Born_in(p,l3).
6	**Logical Formula**	$\forall p\ \exists l\ born\_in(p, l)$
	Alchemy Input	exist 1 Born_in(p,l).

The logical formulae in Table III define the conditions under which their corresponding predicates are true, and which are false otherwise. In logical programming languages such as Prolog this is approximated by the semantics of negation-as-finite-failure. That is, if we can fail to prove something, Q, in a finite time then we can treat that as negation: not(Q). In Alchemy, the default or prior probability is assumed to be 0.5 unless otherwise stated. In order to get the desired semantics we then add formula 3, which essentially sets the prior probability of *contains_loc* to near zero.

2) Observation Fusion: Of course the purpose of using MLNs rather than a logical programming language is to be able to represent and appropriately deal with uncertainties. In principle the MLN formulation allows the use of probabilistic (soft) formulae with hard evidence, but it is also possible to use hard formulae, as we have been doing in this example, as well as *probabilistic (soft) evidence*, which can be achieved by adding a formula consisting of just the evidence predicate with an appropriate weight to the MLN program. As the weights for a single formula behave like log odds, the effective input probability is given by the formula:

$$p = \frac{e^w}{1 + e^w} \tag{3}$$

in which case the corresponding weight is given by the formula:

$$w = \log(\frac{p}{1 - p}). \tag{4}$$

For example, we can add the weighted formulae in Table IV to an MLN program to soften the evidence for *born_in*.

TABLE IV
ADDING UNCERTAIN EVIDENCE FOR ALCHEMY

Alchemy Input		Equivalent Probability
3	Born_in(Anne, Adelaide)	0.95
1	Born_in(Anne, Magill)	0.73
0.5	Born_in(Anne, Cairns)	0.62
-8	Born_in(Germaine, Melbourne)	0.0003
0	Born_in(Frank, Cairns)	0.5

The marginal probabilities for the *born_in* predicate calculated from this program (Tables I-IV) using MC-SAT in Alchemy [18] are shown in Table V. We can see that the system correctly propagates *born_in* up the hierarchy of locations specified. Qualitatively it also appears to handle observation fusion using soft evidence (eg. Anne), and the appropriate combination of hard and soft evidence (Germaine).

TABLE V
MARGINAL INFERENCE USING ALCHEMY
† INDICATES CELLS WHERE *soft evidence* FROM TABLE IV WAS PROVIDED
* INDICATES CELLS WHERE *hard evidence* FROM TABLE I WAS PROVIDED

Born_in					Mystery Person	Anne	Bob	Frank	Germaine	Alice
Australia					1.0	1.0	1.0	1.0	1*	1.0
	Queensland				1.0	0.05	0.0	0.4	0.47	1.0
		SE Queensland			1.0	0.03	0.0	0.25	0.31	1.0
			Brisbane		1*	0.02	0.0	0.2	0.25	1.0
				Kangaroo Pt.	0.48	0.01	0.0	0.12	0.15	1*
		Cairns			0.0	0.02†	0.0	0.11	0.11	0.0
	Victoria				0.0	0.01	0.0	0.17	0.08	0.0
		Melbourne			0.0	0.01	0.0	0.12	0.0†	0.0
	South Australia				0.0	0.94	1.0	0.38	0.4	0.0
		Adelaide			0.0	0.92†	1*	0.25	0.23	0.0
			Magill		0.0	0.66†	0.51	0.16	0.15	0.0
		Barossa Valley			0.0	0.01	0.0	0.08	0.11	0.0

3) Contextual Information: Hierarchical relationships between geographic locations are just one example of the contextual information that needs to be considered when performing entity resolution in general. MLNs allow logical relationships between ground predicates and other constraints to be captured and handled in a probabilistic framework. For example, attributes can differ in their cardinality and whether they are required or optional. These requirements can be encoded in MLN formulae. The predicate *born_in* is an example of an attribute that is required and singular, whereas *marriage_date* would be optional and potentially multiple. Some attributes represent properties of an entity that can change over time. Entity resolution requires comparing the observed values of such attributes over time to determine the extent to which they agree or disagree. Allen's interval calculus [19] provides one way of classifying the temporal relationship between time intervals which can be encoded by MLN formulae. We have used this approach to enable comparison of entities based on where they have resided during different time intervals, represented by the relation *lives_in*.

We can also make use of relations between entities, such as kinship relations. For instance, *sibling_of* is a symmetric, transitive and irreflexive relation that is optional and potentially multiple. These constraints can be modelled with MLN rules and used to help with entity resolution.

4) Entity Resolution: As mentioned earlier, to perform entity resolution we introduce an explicit equality predicate, *same_person*, which is reflexive, symmetric and transitive. See Table VI.

TABLE VI
LOGICAL FORMULAE AND CORRESPONDING MLN RULES FOR *same_person*

Rule	Logical Formula	Alchemy Input
1	$\forall x\ same\_person(x,x)$	Same_Person(a,a)
2	$\forall x,y\ same\_person(x,y) \rightarrow same\_person(y,x)$	Same_Person(a,b) => Same_Person(b,a).
3	$\forall x,y,z\ same\_person(x,y) \wedge same\_person(y,z) \rightarrow same\_person(x,z)$	Same_Person(a,b) ^ Same_Person(b,c) ^ Same_Person(b,c) => same_person(a,c).

We also need to modify the constraint on the predicate *born_in* from Table III to include the predicate *same_person*, so that formula 5 becomes:

$$\forall p1,p2,l1,l2,l3\ born\_in(p1,l2) \wedge contains(l1,l2) \wedge contains(l1,l3) \wedge same\_person(p1,p2) \wedge \neg(l2=l3) \wedge \neg location\_match(l2,l3) \rightarrow \neg born\_in(p2,l3). \quad (5)$$

where *location_match* is a symmetric predicate defined in terms of *contains* that is true if and only if its two arguments are consistent with each other.

In addition, we add one further formula that states the prior probability that two people, born in the same place, are the same person:

$$[0.5]\ \forall p1,p2,l1,l2\ born\_in(p1,l1) \wedge born\_in(p2,l2) \wedge location\_match(l1,l2) \rightarrow same\_person(p1,p2) \quad (6)$$

where the leading [0.5] is the weight applied to this rule in Alchemy.

For illustration, we consider a situation where a 'Mystery Person' is to be added to an existing knowledge base. We will then evaluate the *same_person* predicate to determine which entities are most likely to be the same person.

Table VII summarises the data that are being compared in this entity resolution example. The table identifies which attributes of the existing entities are consistent (coloured in green) or inconsistent (coloured in red) with the entity that is to be added. The observation that Anne is a sibling of the Mystery Person generates a conflict with (ie. cannot match) Anne because Anne cannot be her own sibling.

All of the constraints imposed by these attributes are encoded as MLN rules. Alchemy is then used to convert them to a ground Markov network, which includes all possible groundings of the evidence and query predicates, and to determine the marginal probabilities of *same_person* for each pair of entities in our knowledge base via an MCMC-based algorithm. The results of doing so via the MC-SAT algorithm are shown in Table VIII (partial matches of different entities are coloured in orange, while matches of entities with themselves are shaded grey). This shows that the Mystery Person is most likely to be Alice, which is as expected since there are three instances of agreement between their attributes, and no conflicts. Each of the other people has at least one conflicting attribute, which is why all others have very low probability of match with the Mystery Person. Note that the evaluation indicates a moderate probability that Anne could be the same person as Bob, Frank or Germaine. This is because Anne has no known conflicting attributes with Bob, Frank or Germaine. The match with Bob is most likely because Anne and Bob's reported birthplaces are in agreement (Magill is part of Adelaide).

Another interesting observation is that, while Bob could be Anne and Anne could be Frank, the system concludes that Bob cannot be Frank, even though *same_person* is explicitly transitive. This is because Bob and Frank are siblings - so they cannot be the same person.

TABLE VII
ATTRIBUTE COMPARISON TO MYSTERY PERSON AT A GLANCE.

Relation	Mystery Person	Anne	Bob	Frank	Germaine	Alice
gender	Female		Male	Male	Female	
born in	Brisbane	Magill	Adelaide		!Melbourne	Kangaroo Pt.
lives in (interval)	SE Queensland 78-138			Kangaroo Pt. 125-150	Cairns 94-113	Brisbane 104-129
lives in (interval)	Melbourne 145-160	Victoria 140-150				Melbourne 151-160
sibling of	Anne		Frank			

TABLE VIII
OVERALL ESTIMATE OF ENTITY SIMILARITY. ALICE AND THE MYSTERY PERSON ARE THE CLOSEST MATCH; AND ANNE COULD POTENTIALLY BE BOB, FRANK OR GERMAINE.

Same_person	Mystery Person	Alice	Anne	Bob	Frank	Germaine
Mystery Person	1.0	0.98	0.0	0.01	0.01	0.0
Alice	0.98	1.0	0.01	0.0	0.0	0.01
Anne	0.0	0.01	1.0	0.47	0.19	0.17
Bob	0.01	0.0	0.47	1.0	0.0	0.0
Frank	0.01	0.0	0.19	0.0	1.0	0.0
Germaine	0.0	0.01	0.17	0.0	0.0	1.0

V. DISCUSSION AND LESSONS LEARNT

While we regard the results of the example in Section IV as quite promising, our experience with MLNs has shown that developing and configuring them for conservative use where quality assurance is paramount can at times be quite challenging. In order to help progress the employment of MLNs for multi-level information fusion applications, in this section we discuss several issues for consideration when configuring MLNs, not only for entity resolution, but also for more general problems.

To begin with, a cautionary approach to building the MLNs is advised. Trying to introduce too much evidence or too many 'submodels' all at once to solve the problem might work, but if unexpected results occur, trying to debug or trace back the source of the error is difficult. Starting with a basic model and checking at each step to ensure consistent results are being given ensures that any potential anomalies can be more easily

found. By using small amounts of data and constant testing, the complexity of an MLN can be slowly increased until the problem at hand is modelled satisfactorily.

In regards to quality assurance, another issue to take into consideration when configuring MLNs is how the formulae are processed by some of the software implementations of MLNs. Adhering strictly to the definition of an MLN, one would expect that the formulae for the MLN model would be treated as a whole, which is to say that for any possible world, the overall truth value of a formula F_i would be evaluated by assigning the truth values for that possible world to the ground predicates in F_i and using Boolean algebra to evaluate the final result. However, this is not the default behaviour of some of the main MLN software implementations including Alchemy, which we have used for this paper. How they handle each soft formula F_i with weight w_i is to compute the CNF of F_i and then treat the N disjuncts in the CNF as N individual soft formulae each with a weight of w_i/N. The resulting ground Markov network is completely different and so in general will yield different results, as various of the individual CNF components will likely be satisfiable in some possible worlds in which the original formula F_i is not. For some domains, this added flexibility may be desirable, but in the intelligence domain, where the strength of the assessments must also be taken into account, it is completely undesirable. It is noted that this behaviour can be overridden in Alchemy and does not apply to hard formulae which are treated indivisibly, but not being aware of this unexpected behaviour in the first place can be a trap for the unwary.

TABLE IX
ASSERTED PROLOG FACTS FOR ATTRIBUTE *contains*

Location	Relation	Location	Prolog Input
Australia	*contains*	Queensland	contains_loc0(australia, queensland)
Australia	*contains*	Victoria	contains_loc0(australia, victoria)
Australia	*contains*	South Australia	contains_loc0(australia, south_australia)
Brisbane	*contains*	Kangaroo Point	contains_loc0(brisbane, kangaroo_pt)
Queensland	*contains*	Cairns	contains_loc0(queensland, cairns)
Queensland	*contains*	SE Queensland	contains_loc0(queensland, se_queensland)
SE Queensland	*contains*	Brisbane	contains_loc0(se_queensland, brisbane)
South Australia	*contains*	Barossa Valley	contains_loc0(south_australia, barossa_valley)
Adelaide	*contains*	Magill	contains_loc0(adelaide, magill)
Victoria	*contains*	Melbourne	contains_loc0(victoria, melbourne)

Finally, for potential users of MLNs who are familiar with the negation-as-finite-failure behaviour of Prolog, it is important to note that MLNs do not behave in this manner. This point was raised in Section IV-A1. To clarify the nature of the issue which we identified, consider the FOL predicate *contains*. The evidence for this ground predicate is specified in Table II and the formulae governing its (hard) FOL properties are specified in formulae 1 and 2 of Table III. Consider a similar formulation in Prolog using the asserted facts in Table IX, along with rules 7 and 8 below (where *contains_loc0* signifies an asserted fact and *contains_loc* signifies an inferred fact):

$$contains\_loc(X,Y) \;:-\; contains\_loc0(X,Y). \quad (7)$$

$$contains\_loc(X,Z) \;:-\; contains\_loc0(X,Y), \; contains\_loc(Y,Z). \quad (8)$$

While all versions of Prolog will reply in the affirmative to the query *contains_loc(australia,melbourne)* because

$$contains\_loc(victoria, melbourne) :- \; contains\_loc0(victoria, melbourne). \quad (9)$$

and

$$contains\_loc(australia, melbourne) :- \; contains\_loc0(australia, victoria), \; contains\_loc(victoria, melbourne). \quad (10)$$

are both satisfied, it will reply in the negative to a similar query such as *contains_loc(south_australia,brisbane)* because this fact has not been asserted and no instantiation of rules 7 and 8 enable it to be inferred. In reality, South Australia does not contain Brisbane, so the Prolog knowledge base is behaving as desired. However, what happens in the corresponding MLN is that (i) the ground predicate *Contains_loc(South_Australia,Brisbane)* becomes a query variable in the ground Markov network because it hasn't been asserted as evidence and (ii) while it is not logically entailed by the evidence and the hard FOL formulae 1 and 2 of Table III, it is consistent with them. Consequently a non-zero probability of *Contains_loc(South_Australia,Brisbane)* being true is returned. To suppress this type of behaviour, we added formula 3 from Table III to introduce a negation-as-finite-failure type effect into Alchemy for the *contains* predicate. By assigning a weight of 8 to !*Contains_loc(a,b)*, it has the effect of assigning almost zero probability to any ground predicate *Contains_loc(a,b)* meeting the two conditions above, while returning a unit probability to any other grounding *Contains_loc(c,d)* which is either asserted as evidence or entailed by evidence and formulae 1 and 2 in Table III.

VI. CONCLUSION

In this paper, we have discussed the use of Markov logic networks for incorporating multi-level information such as uncertain attribute data of entities and contextual information into the entity resolution process and have demonstrated how this provides a framework that is well suited for handling the real-world issues encountered in intelligence-based entity resolution problems. Due to their basis in FOL, MLNs are conceptually easy to understand and reasonably intuitive, allowing an intelligence analyst to be able to comprehend them and even make recommendations on how they could be constructed according to the operating environment, although, as we have identified, there are still technical issues to be properly understood which are blocking the systematic use of such a technology at this stage by other than experts. More effort also needs to be devoted to incorporating temporal reasoning into MLN models (support for integrating floating point operations would also be highly desirable for this).

The example we presented demonstrated primarily how MLNs may be applied to assist with entity resolution. However, as we noted in the introduction, there is also scope

to use MLNs to assist in data cleaning and observation fusion. Yet another area of intelligence where MLNs are applicable is that of situation and threat / impact assessment. In particular, given contextual information about individuals and their known previous movements, MLNs can be employed to determine their next likely movements or even infer their likely intentions. Some research in this area has already been conducted. Snidaro et al. have looked at MLNs in this context but in the maritime environment [20], [21], and Mooney et al. have investigated abductive MLNs for activity recognition [22], [23], while Ao et al. have incorporated MLNs inter alia into a scientific inquiry fusion theory for high level information fusion [24].

Lastly, one issue that we haven't addressed so far is the speed of inference and the scalability of the models. While Alchemy has been useful for demonstrating the proof-of-concept of employing MLNs for intelligence purposes, these two issues remain valid concerns. Efforts to address them are being made by a research program at Stanford University. To date, they have implemented another software application called *Tuffy* [25] which (i) uses a bottom-up approach to grounding which is faster than the top-down approach of Alchemy, (ii) is underpinned by a relational database management system to address scalability problems, and (iii) implements novel partitioning, loading and parallel algorithms. Currently, they are working on a successor to Tuffy called *DeepDive* [26]. Investigations into the evaluation of these approaches will be the subject of further research.

REFERENCES

[1] R. Johnston, *Analytic Culture in the US Intelligence Community: An Ethnographic Study*. Imaging and Publication Support, CIA, 2005. [Online]. Available: https://www.cia.gov/library/center-for-the-study-of-intelligence/csi-publications/books-and-monographs/analytic-culture-in-the-u-s-intelligence-community/analytic_culture_report.pdf

[2] S. N. Minton, S. A. Macskassy, P. M. LaMonica, K. See, C. A. Knoblock, G. Barish, M. Michelson, and R. Liuzzi, "Monitoring entities in an uncertain world: Entity resolution and referential integrity," in *Twenty-Third Innovative Applications of Artificial Intelligence Conference (AAAI-11), San Francisco, 7-11 August*, 2011, pp. 1681–1688.

[3] H. L. Dunn, "Record linkage," *American Journal of Public Health*, vol. 36, no. 12, pp. 1412–1416, 1946.

[4] H. B. Newcombe, J. M. Kennedy, S. Axford, and A. James, "Automatic linkage of vital records," *Science*, vol. 130, no. 3381, pp. 954–959, 1959.

[5] I. P. Fellegi and A. B. Suntner, "A theory for record linkage," *Journal of the American Statistical Association*, vol. 64, no. 328, pp. 1183–1210, 1969.

[6] P. Singla and P. Domingos, "Entity resolution with Markov logic," in *IEEE International Conference on Data Mining (ICDM 2006), Hong Kong, 18-22 December*. IEEE Computer Society Press, 2006, pp. 572–582.

[7] P. Domingos and M. Richardson, "Markov logic - a unifying framework for statistical relational learning," in *Introduction to Statistical Relational Learning*, L. Getoor and B. Taskar, Eds. MIT Press, 2007.

[8] P. Domingos and D. Lowd, *Markov Logic: An Interface Layer for Artificial Intelligence*, 1st ed., ser. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2009.

[9] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.

[10] S. Lauritzen, *Graphical Models*, 2nd ed., ser. Oxford Statistical Science Series. Clarendon Press, Oxford, 2004, no. 17.

[11] D. Koller, N. Friedman, L. Getoor, and B. Taskar, "Graphical models in a nutshell," in *Introduction to Statistical Relational Learning*, L. Getoor and B. Taskar, Eds. MIT Press, 2007.

[12] G. Tourlakis, *Cambridge studies in advanced mathematics, Lectures in Logic and Set Theory Volume I: Mathematical Logic*, 1st ed. Cambridge University Press, 2003.

[13] K. A. Ross and C. R. Wright, *Discrete Mathematics*, 2nd ed. Prentice-Hall International Editions, 1988.

[14] P. A. Grossman, *Discrete Mathematics for Computing*, 2nd ed. Macmillan Education Australia Pty. Ltd., 1995.

[15] S. J. Russell and P. Norvig, *Artificial Intelligence - A Modern Approach*, ser. Artificial Intelligence. USA: Prentice-Hall International, Inc., 1995.

[16] Z. Ao and D. Lambert, "Semantic inference by first order logic," in *15th International Conference on Information Fusion (Fusion 2012), Singapore, 9-12 July*, 2012.

[17] S. Kok, P. Singla, M. Richardson, P. Domingos, M. Summer, and H. Poon, *The Alchemy System for Statistical Relational AI: User Manual*, Department of Computer Science and Engineering, University of Washington, Seattle, Washington State, USA, 2007.

[18] S. Kok, P. Singla, M. Richardson, and P. Domingos, "The Alchemy system for statistical relational AI," Department of Computer Science and Engineering, University of Washington, Seattle, Washington State, USA, Tech. Rep., 2005.

[19] J. F. Allen, "Maintaining knowledge about temporal intervals," *Communications of the ACM*, vol. 26, pp. 832–843, 1983.

[20] L. Snidaro, I. Visentini, and K. Bryan, "Fusing uncertain knowledge and evidence for maritime situational awareness via Markov logic networks," *Information Fusion*, vol. 21, pp. 159–172, 2015.

[21] L. Snidaro, I. Visentini, K. Bryan, and G. L. Foresti, "Markov logic networks for context integration and situation assessment in maritime domain," in *15th International Conference on Information Fusion (Fusion 2012), Singapore, 9-12 July*, 2012, pp. 1534–1539.

[22] R. J. Kate and R. J. Mooney, "Probabilistic abduction using Markov logic networks," in *Proceedings of the IJCAI-09 Workshop on Plan, Activity, and Intent Recognition (PAIR-09), Pasadena, California, USA, 11 July*, 2009.

[23] P. Singla and R. J. Mooney, "Abductive Markov logic for plan recognition," in *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, San Francisco, California, USA, 7-11 August*, 2011, pp. 1069–1075.

[24] Z. Ao, J. Scholz, and M. Oxenham, "A scientific inquiry fusion theory for high-level information fusion," in *17th International Conference on Information Fusion (Fusion 2014), Salamanca, Spain, 7-10 July*, 2014.

[25] F. Niu, C. Ré, A. Doan, and J. Shavlik, "Tuffy: Scaling up statistical inference in Markov logic networks using an RDBMS," University of Wisconsin-Madison, Tech. Rep., 2010.

[26] C. Ré, M. Cafarella, A. A. Sadeghian, Z. Shan, J. Shin, F. Wnag, S. Wu, and C. Zhang, "DeepDive," 2015. [Online]. Available: http://deepdive.stanford.edu/