Online Inference for Knowledge Graph Construction

Jay Pujara, Ben London
University of Maryland, College Park
{jay,blondon}@cs.umd.edu

Lise Getoor
University of California, Santa Cruz
getoor@soe.ucsc.edu

William W. Cohen
Carnegie Mellon University
wcohen@cs.cmu.edu

Abstract

The task of knowledge graph construction presents a confounding challenge for statistical relational models. While the uncertainty of extractions from NLP tools and the ontological structure of knowledge are a perfect match for the strengths of statistical relational techniques, the vast and continually growing evidence from which knowledge graphs are constructed can make such models prohibitively expensive. We address this challenge by presenting two lines of research that provide a foundation for online knowledge graph construction. The first is work on knowledge graph identification, a scalable probabilistic model for combining statistical features from uncertain extractions and ontological constraints to efficiently construct a knowledge graph. The second is the necessary theory and accompanying algorithms for partially updating an inferred knowledge graph. We illustrate how combining these components presents the opportunity to apply sophisticated statistical relational models to complex domains, such as knowledge graph construction, without sacrificing quality or efficiency.

The Challenge of Lifelong Learning

Statistical relational models have a key strength: they capture and represent the myriad relationships found in real-world data while incorporating probabilistic semantics to quantify uncertainty. Unfortunately, the sophisticated models our community builds are accompanied by a hefty price tag. For many practical applications, models will contain millions of statistical dependencies, making inference slow or even intractable. This obstacle is compounded by "lifelong learning" settings where new evidence continually arrives – if repeating inference is costly, the resulting inferences can quickly become outdated and inaccurate.

One task that fits the paradigm of lifelong learning is knowledge graph construction. Knowledge graphs are structured knowledge bases where nodes represent entities and edges represent relationships between these entities. A growing number of projects have built vast knowledge graphs from Web data sources, including (among many others) YAGO (Suchanek, Kasneci, and Weikum 2007), OLLIE (Etzioni et al. 2008), DeepDive (Niu et al. 2012), and the Knowledge Vault (Dong et al. 2014). As new information is added to the Web – whether through news stories, user postings, or site updates – these knowledge bases must be updated to reflect the new information.

As a case study, we consider a project that explicitly approaches knowledge graph construction as a lifelong learning task: NELL (Carlson et al. 2010), the Never-ending Language Learner. On a roughly daily basis, NELL employs a collection of information extraction techniques to discover candidate facts to add to its knowledge base. Using patterns of agreement and disagreement between extraction techniques, NELL selectively promotes candidate extractions that are likely to be true and uses these promotions as a feedback signal to train each of the extractors. Crucial to the success of NELL is deciding which candidate extractions are true using the rich dependencies between extracted facts. However, the tens to hundreds of millions of facts in NELL's knowledge base make running inference to construct a complete knowledge graph impractical. Instead, the problem of promoting instances in NELL requires efficient updates to the knowledge graph using new extractions to bolster inferences on uncertain facts.

In this paper, we summarize two existing lines of work that form the foundation for the online construction of massive knowledge graphs. The first component is knowledge graph identification (Pujara et al. 2013), a statistical relational model that combines ontological constraints and statistical features from uncertain extractions to infer a complete knowledge graph. The second component is recent work on online collective inference (Pujara, London, and Getoor 2015) which offers a theoretical bound on inference regret when making partial inference updates and algorithms for selectively updating the MAP state while maintaining low error. Finally, we illustrate how combining these two components can be applied to the large-scale lifelong learning problem of knowledge graph construction.

Background: Knowledge Graph Identification

In this section we provide background on knowledge graph identification, the task of transforming a noisy extraction graph to a consistent knowledge graph. Candidate facts from an information extraction system can be represented as an *extraction graph* where entities are nodes, categories are labels associated with each node, and relations are directed edges between the nodes. Unfortunately, the output from an information extraction system is often incorrect; the graph constructed from it has spurious and missing nodes and edges, and missing or inaccurate node labels. Our approach,

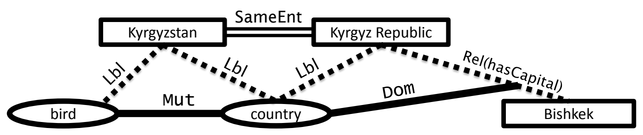

Figure 1: An illustration of the example showing how knowledge graph identification can resolve conflicting information in a knowledge graph. Entities are shown in rectangles, dotted lines represent uncertain information, solid lines show ontological constraints and double lines represent co-referent entities found with entity resolution.

knowledge graph identification (KGI) (Pujara et al. 2013), performs collective classification, link prediction, and entity resolution in the presence of rich ontological information and multiple sources of uncertain information, ultimately producing a better knowledge graph.

Unlike earlier work on graph identification (Namata, Kok, and Getoor 2011) we use a very different probabilistic framework, probabilistic soft logic PSL (Bach et al. 2015). PSL allows us to apply global constraints instead of relying only on local features. PSL models are expressed using a set of universally-quantified logical rules that, when combined with data such as the noisy extractions and ontological information, define a probability distribution over possible knowledge graphs. In the case of KGI, we introduce rules to relate uncertain extractions to the true relations and labels in the knowledge graph, pool these facts across co-referent entities, and constrain relations and labels with rules that use ontological information such as domain and range constraints, mutual exclusion relationships. We explain how these components of KGI map to PSL rules, motivating these rules with examples of challenges found the Never-Ending Language Learner (NELL).

Representation of Uncertain Extractions

NELL produces candidate extractions from a web corpus which often contains noise. Candidate extractions from the NELL corpus include labels such as `bird(kyrgyzstan)` and `country(kyrghyzstan)` as well as relations such as `locatedIn(kyrghyzstan, Russia)`, `locatedIn(kyrgyz republic, Asia)`, `locatedIn(kyrghyzstan, US)`, and `locatedIn(kyrgyzstan, Kazakhstan)`. These extractions can contain mistakes that include variations in spelling and inconsistencies in relationships, and NELL assigns confidence values to each extraction.

In PSL, we represent these candidate extractions with predicates CANDLBL and CANDREL, e.g. CANDLBL(`kyrgyzstan, bird`) and CANDREL(`kyrgyz republic, Asia, locatedIn`). NELL has multiple extractors that generate candidates, and we can use different predicates for each extractor to capture the confidence of that extractor. For a given extractor T, we introduce predicates CANDREL_T and CANDLBL_T to represent the candidates extracted by T. We relate these candidates to the facts that we wish to infer, LBL and REL, using the following rules:

$$\text{CANDREL}_T(E_1, E_2, R) \overset{w_{CR-T}}{\Rightarrow} \text{REL}(E_1, E_2, R)$$

$$\text{CANDLBL}_T(E, L) \overset{w_{CL-T}}{\Rightarrow} \text{LBL}(E, L)$$

PSL uses *soft logic*, so we can represent noisy extractions by translating confidences into real-valued truth assignments in the $[0, 1]$ range. For example, if NELL extracts the relation `locatedIn(kyrgyz republic,Asia)` and assigns it a confidence value of 0.9, we would assign the atom CANDREL(`kyrgyzstan,Asia,locatedIn`) a soft-truth value of 0.9. Similarly, our output values for unknown facts are in the $[0, 1]$ range allow us to trade-off precision and recall by using a truth threshold. By learning the weights of these rules, w_{CL_T} and w_{CR_T}, our model combines multiple sources of information to label nodes and predict links.

Reasoning About Co-Referent Entities

While the previous PSL rules provide the building blocks of predicting links and labels using uncertain information, KGI employs entity resolution to pool information across co-referent entities. In the example above, many different forms for the country Kyrgystan appear: `kyrgyzstan`, `kyrghyzstan`, and `kyrgyz republic`. A key component of KGI is identifying possibly co-referent entities and determining the similarity of these entities. We use the SAMEENT predicate to capture the similarity of two entities. While any similarity metric can be used, we compute the similarity of entities using a process of mapping each entity to the YAGO knowledge base (Suchanek, Kasneci, and Weikum 2007), extracting a set of Wikipedia articles for each entity and then computing the Jaccard index of possibly co-referent entities. We incorporate this information about co-referent entities by constraining the labels and relations of these entities through PSL rules:

$$\text{SAMEENT}(E_1, E_2) \wedge \text{LBL}(E_1, L) \overset{w_{EL}}{\Rightarrow} \text{LBL}(E_2, L)$$

$$\text{SAMEENT}(E_1, E_2) \wedge \text{REL}(E_1, E, R) \overset{w_{ER}}{\Rightarrow} \text{REL}(E_2, E, R)$$

$$\text{SAMEENT}(E_1, E_2) \wedge \text{REL}(E, E_1, R) \overset{w_{ER}}{\Rightarrow} \text{REL}(E, E_2, R)$$

These rules define an equivalence class of entities, such that all entities related by the SAMEENT predicate must have the same labels and relations. The soft-truth value of the SAMEENT, derived from our similarity function, mediates the strength of these rules. When two entities are very similar, they will have a high truth value for SAMEENT, so any label assigned to the first entity will also be assigned to the second entity. On the other hand, if the similarity score for two entities is low, the truth values of their respective labels and relations will not be strongly constrained.

Incorporating Ontological Information

Although entity resolution allows us to relate extractions that refer to the same entity, knowledge graphs can employ ontological information to specify rich relationships between many facts. Our ontological constraints are based on the logical formulation proposed in (Jiang, Lowd, and Dou 2012). Each type of ontological relation is represented as a predicate, and these predicates represent ontological knowledge of the relationships between labels and relations. For example, the constraints DOM(`hasCapital`, country) and $\text{R}\textsc{ng}$(hasCapital, city) specify that the relation hasCapital is a mapping from entities with label country to entities with label city. The constraint $\text{M}\textsc{ut}$(country, bird) specifies that the labels country and bird are mutually exclusive, so that an entity cannot have both the labels country and bird. We similarly use constraints for subsumption of labels ($\text{S}\textsc{ub}$) and inversely-related functions ($\text{I}\textsc{nv}$). To use this ontological knowledge, we introduce rules relating each ontological relation to the predicates representing our knowledge graph. We specify seven types of ontological constraints in our experiments:

$$\begin{array}{llll}
\text{D}\textsc{om}(R, L) & \wedge\ \text{R}\textsc{el}(E_1, E_2, R) & \overset{w_O}{\Rightarrow} & \text{L}\textsc{bl}(E_1, L) \\
\text{R}\textsc{ng}(R, L) & \wedge\ \text{R}\textsc{el}(E_1, E_2, R) & \overset{w_O}{\Rightarrow} & \text{L}\textsc{bl}(E_2, L) \\
\text{I}\textsc{nv}(R, S) & \wedge\ \text{R}\textsc{el}(E_1, E_2, R) & \overset{w_O}{\Rightarrow} & \text{R}\textsc{el}(E_2, E_1, S) \\
\text{S}\textsc{ub}(L, P) & \wedge\ \text{L}\textsc{bl}(E, L) & \overset{w_O}{\Rightarrow} & \text{L}\textsc{bl}(E, P) \\
\text{RS}\textsc{ub}(R, S) & \wedge\ \text{R}\textsc{el}(E_1, E_2, R) & \overset{w_O}{\Rightarrow} & \text{R}\textsc{el}(E_1, E_2, S) \\
\text{M}\textsc{ut}(L_1, L_2) & \wedge\ \text{L}\textsc{bl}(E, L_1) & \overset{w_O}{\Rightarrow} & \neg\text{L}\textsc{bl}(E, L_2) \\
\text{RM}\textsc{ut}(R, S) & \wedge\ \text{R}\textsc{el}(E_1, E_2, R) & \overset{w_O}{\Rightarrow} & \neg\text{R}\textsc{el}(E_1, E_2, S)
\end{array}$$

Putting It All Together

Inferring a knowledge graph becomes challenging as we consider the many interactions between the uncertain extractions that we encounter in KGI (Figure 1). For example, NELL's ontology includes the constraint that the attributes bird and country are mutually exclusive. While extractor confidences may not be able to resolve which of these two labels is more likely to apply to kyrgyzstan, reasoning collectively using entity resolution and ontological constraints can provide a solution. For example, NELL is highly confident that kyrgyz republic has a capital city, Bishkek. The NELL ontology specifies that the domain of the relation hasCapital has label country. Entity resolution allows us to infer that kyrgyz republic refers to the same entity as kyrgyzstan. Deciding whether Kyrgyzstan is a bird or a country now involves a prediction where we include the confidence values of the corresponding bird and country facts from co-referent entities, as well as collective features from ontological constraints of these co-referent entities, such as the confidence values of the hasCapital relations.

To accomplish this collective reasoning, we use PSL to define a joint probability distribution over knowledge graphs. The universally-quantified rules described are a PSL model and provide the basis for defining this probability distribution. In a PSL program, $\mathbf{\Pi}$, this model is *grounded* by substituting values from NELL's noisy extractions into the rule template. For example, the rule:

$$\text{D}\textsc{om}(R, L) \ \wedge\ \text{R}\textsc{el}(E_1, E_2, R) \ \overset{w_O}{\Rightarrow}\ \text{L}\textsc{bl}(E_1, L)$$

can be grounded by substituting atoms from NELL, $\text{D}\textsc{om}$(hasCapital, country), $\text{R}\textsc{el}$(kyrgyzstan, Bishkek, hasCapital), and $\text{L}\textsc{bl}$(kyrgyzstan, country), into the rule template. We refer to the collection of ground rules in the program as G.

Unlike Boolean logic where each grounding would have a binary truth value, our choice of soft logic requires a different definition of truth value. We refer to an assignment of soft-truth values to atoms as an interpretation, which, in this case, corresponds to a possible knowledge graph, and use the *Lukasiewicz t-norm* and *co-norm* to determine the truth values of logical formulas under an interpretation. This t-norm defines a relaxation of logical connectives as follows:

$$p \wedge q = \max(0, p + q - 1)$$
$$p \vee q = \min(1, p + q)$$
$$\neg p = 1 - p$$

With this definition, for each possible knowledge graph, I, we can assign a truth value $T_r(I)$ to every grounding $g \in G$ and define a *distance to satisfaction*, $\phi_r(I) = 1 - T_r(I)$ for each grounding. Since PSL rules allow antecedents consisting of conjuncted atoms and consequents with disjuncted atoms, $\phi_r(I)$ takes the familiar form of the hinge-loss, and PSL models are an instance of *hinge-loss Markov random fields* (Bach et al. 2013).

The probability distribution over knowledge graphs, $P_{\mathbf{\Pi}}(I)$ can now be defined using a weighted combination of the distance to satisfaction of ground rules in the PSL program:

$$P_{\mathbf{\Pi}}(I) = \frac{1}{Z}\exp\left[-\sum_{g \in G} w_r \phi_r(I)^p\right]$$

where $p \in 1, 2$ specifies a linear or quadratic combination and Z is a normalization constant.

Most probable explanation (MPE) inference corresponds to identifying a graph that maximizes $P_{\mathbf{\Pi}}(G)$, which can then be mapped to a set of labels and relations that comprises the true knowledge graph. In our work, we choose a soft-truth threshold and determine the true entities, labels and relations by using those atoms whose truth value exceeds the threshold. MPE inference can be formulated as convex optimization in PSL, and using the alternating direction method of multipliers (ADMM), which scales linearly with the number of ground rules in the PSL program. In the next section, we provide a deeper analysis of ADMM optimization.

Background: Online Collective Inference

In this section, we introduce the concept of inference regret, which is informally defined as the distance between the MAP state resulting from full inference over all evidence and the MAP state produced through a partial inference update. We then summarize the inference regret bound in (Pujara, London, and Getoor 2015), which applies for models with a strongly-convex inference objective, such as the HL-MRFs used in PSL. The remainder of the section provides an in-depth analysis of the ADMM optimization algorithm and a detailed explanation of how features from the optimization can be used to selectively update the MAP state.

Regret Bound for Online Inference

Fix a model with energy function defined as $E(\mathbf{y} \| \mathbf{x}; \dot{\mathbf{w}}) \triangleq \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}) + \frac{w_p}{2} \|\mathbf{y}\|_2^2$. Suppose we are given evidence, $\mathbf{X} = \mathbf{x}$, from which we make a prediction, $\mathbf{Y} = \mathbf{y}$, using MAP inference. Then, some subset of the unknowns are revealed. Conditioning on the new evidence, we have two choices: we can recompute the MAP state of the remaining variables, using full inference; or, we can fix some of the previous predictions, and only update a certain subset of the variables. To understand the consequences of fixing our previous predictions we must answer a basic question: how much will the partial update differ from the MAP state resulting from full inference?

We formalize the above question in the following concept.

Definition 1. Fix a budget $m \geq 1$. For some subset $\mathcal{S} \subset \{1, \ldots, n\}$, such that its complement $\overline{\mathcal{S}} \triangleq \{1, \ldots, n\} \setminus \mathcal{S}$, has size $\left|\overline{\mathcal{S}}\right| = m$, let $\mathbf{Y}_{\mathcal{S}}$ denote the corresponding subset of the variables, and let $\mathbf{Y}_{\overline{\mathcal{S}}}$ denote its complement. Assume there is an operator Γ that concatenates $\mathbf{Y}_{\mathcal{S}}$ and $\mathbf{Y}_{\overline{\mathcal{S}}}$ in the correct order. Fix a model, $\dot{\mathbf{w}}$, and an observation, $\mathbf{X} = \mathbf{x}$. Further, fix an assignment, $\mathbf{Y}_{\mathcal{S}} = \mathbf{y}_{\mathcal{S}}$, and let

$$h(\mathbf{x}, \mathbf{y}_{\mathcal{S}}; \dot{\mathbf{w}}) \triangleq \Gamma\left(\mathbf{y}_{\mathcal{S}}, \operatorname*{arg\,min}_{\mathbf{y}_{\overline{\mathcal{S}}}} E\left(\Gamma(\mathbf{y}_{\mathcal{S}}, \mathbf{y}_{\overline{\mathcal{S}}}) \| \mathbf{x}; \dot{\mathbf{w}}\right)\right)$$

denote the new MAP configuration for $\mathbf{Y}_{\overline{\mathcal{S}}}$ after fixing $\mathbf{Y}_{\mathcal{S}}$ to $\mathbf{y}_{\mathcal{S}}$. We define the *inference regret* for $(\mathbf{x}, \mathbf{y}_{\mathcal{S}}; \dot{\mathbf{w}})$ as

$$\mathfrak{R}_n(\mathbf{x}, \mathbf{y}_{\mathcal{S}}; \dot{\mathbf{w}}) \triangleq \frac{1}{n} \left\| h(\mathbf{x}; \dot{\mathbf{w}}) - h(\mathbf{x}, \mathbf{y}_{\mathcal{S}}; \dot{\mathbf{w}}) \right\|_1. \quad (1)$$

Given this definition of regret, we introduce a bound for strongly-convex inference objectives. Here the L^2 prior with weight w_p guarantees strong convexity.

Proposition 1. *Fix a model with weights* $\dot{\mathbf{w}}$. *Assume there exists a constant* $B \in [0, \infty)$ *such that, for any* $\mathbf{x}$, *and any* $\mathbf{y}, \mathbf{y}'$ *that differ at coordinate* i,

$$\left\| \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}) - \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}') \right\|_2 \leq B \left| y_i - y_i' \right|. \quad (2)$$

Then, for any observations $\mathbf{x}$, *any budget* $m \geq 1$, *any subset* $\mathcal{S} \subset \{1, \ldots, n\} : \left|\overline{\mathcal{S}}\right| = m$, *and any assignments* $\mathbf{y}_{\mathcal{S}}$, *with* $\hat{\mathbf{y}} \triangleq h(\mathbf{x}; \dot{\mathbf{w}})$, *we have that*

$$\mathfrak{R}_n(\mathbf{x}, \mathbf{y}_{\mathcal{S}}; \dot{\mathbf{w}}) \leq \sqrt{\frac{1}{n}\left(\frac{3}{2} + \frac{B \left\|\mathbf{w}\right\|_2}{w_p}\right) \left\| \mathbf{y}_{\mathcal{S}} - \hat{\mathbf{y}}_{\mathcal{S}} \right\|_1}.$$

Summarizing the bound, we show that, beyond fixed model parameters such as the Lipschitz constant (B) and model weights, inference regret depends on the L^1 distance between the fixed variables ($\mathbf{y}_{\mathcal{S}}$) and their values in full inference ($\hat{\mathbf{y}}_{\mathcal{S}}$). This conclusion provides a crucial insight: successfully approximating the MAP state with a partial update requires fixing those variables that change the least. In the following sections we provide an analysis of the optimization used to find the MAP state and how features from the optimization algorithm allow us to design activation algorithms which choose which variables to infer.

Alternating Direction Method of Multipliers

(Bach et al. 2012) have shown that applying consensus optimization using the Alternating Direction Method of Multipliers (ADMM) (Boyd et al. 2011) provides scalable inference for HL-MRFs. For clearer exposition, we express the inference in terms of the set of ground rules, $\mathcal{G}$ and rewrite the energy function as:

$$E(\mathbf{y} \| \mathbf{x}; \dot{\mathbf{w}}) \triangleq \sum_{g \in \mathcal{G}} w_g f_g(\mathbf{x}, \mathbf{y}) + \frac{w_p}{2} \left\| \mathbf{y} \right\|_2^2$$

Here, $w_g f_g(\mathbf{x}, \mathbf{y})$ is a weighted potential corresponding to a single ground rule. ADMM operates by substituting the global optimization problem with local optimizations for each potential that use independent copies of the variables. For each grounding $g \in \mathcal{G}$, let $\mathbf{y}_g$ denote the variables involved in g and $\tilde{\mathbf{y}}_g$ indicate the local copy of those variables. To reconcile the local optimizations, ADMM introduces a constraint that local copies agree with the global "consensus" variables for each variable i involved in the grounding; that is, $\mathbf{y}_g(i) = \tilde{\mathbf{y}}_g(i)$. This constraint is transformed into an augmented Lagrangian with penalty parameter $\rho > 0$ and Lagrange multipliers $\boldsymbol{\alpha}_g$:

$$\min_{\tilde{\mathbf{y}}_g} \; w_g \, f_g(\mathbf{x}, \tilde{\mathbf{y}}_g) + \frac{\rho}{2} \left\| \tilde{\mathbf{y}}_g - \mathbf{y}_g + \frac{1}{\rho} \boldsymbol{\alpha}_g \right\|_2^2, \quad (3)$$

where w_g and f_g are the weight and potential associated with g. ADMM iteratively alternates optimizing the local potentials, then updating the consensus estimates and associated Lagrange multipliers.

$$\tilde{\mathbf{y}}_g \leftarrow \operatorname{argmin}_{\tilde{\mathbf{y}}_g} w_g \, f_g(\mathbf{x}, \tilde{\mathbf{y}}_g) + \frac{\rho}{2} \left\| \tilde{\mathbf{y}}_g - \mathbf{y}_g + \frac{1}{\rho} \boldsymbol{\alpha}_g \right\|^2;$$

$$\mathbf{y}[i] \leftarrow \operatorname{mean}_g(\tilde{\mathbf{y}}_g[i]); \;\; \boldsymbol{\alpha}_g[i] \leftarrow \boldsymbol{\alpha}_g[i] + \rho(\tilde{\mathbf{y}}_g[i] - \mathbf{y}_g[i]).$$

A key element of this optimization is the interplay of two components: the weighted potential corresponding to a grounding and the Lagrangian penalty for deviating from the consensus estimate. As optimization proceeds, the Lagrange multipliers are updated to increase the penalty for deviating from the global consensus. At convergence, a balance exists between the two components reconciling the local minimizer and the aggregate of global potentials.

ADMM Features

The goal of activation is to determine which variables are most likely to change in a future inference. From the analysis in the previous section, we can identify several basic elements for each variable in the model that serve as features for an activation algorithm. For each variable, we have its value at convergence ($\mathbf{y}(i)$), and for each grounding g, the weight (w_g), the value of the potential ($f_g(\mathbf{x}, \tilde{\mathbf{y}}_g)$), and the Lagrange multipliers ($\boldsymbol{\alpha}_g$) measuring the aggregate deviation from consensus. We discuss each of these features to motivate their importance in an activation algorithm.

The value of a variable at convergence can provide a useful signal in certain situations, where a model has clear semantics. For example, the formulation of HL-MRFs often lends itself to a logical interpretation with binary outcomes, as in the cases of collective classification of attributes that are either present or absent. In this setting, assignments in the vicinity of $1/2$ represent uncertainty, and therefore provide good candidates for activation. While, this feature is not universal, in cases where the HL-MRF models Boolean variables it can capture a useful signal.

The weighted potentials of each variable contribute directly to the probability of the MAP configuration. Since the log-probability is proportional to the *negated* energy, $-E$, high weights and high potential values decrease the probability of the assignment. Intuitively, activating those variables that contribute high weighted potentials provides the best mechanism for approaching the full inference MAP state. A complication to this approach is that each weighted potential can depend on many variables. However, the potential value is a scalar quantity and there is no general mechanism to apportion the loss to the contributing variables.

In contrast, the Lagrange multipliers provide a granular perspective on the loss. For each variable in a potential, the Lagrange multiplier measures the aggregate deviation of that variable from the global consensus. High Lagrange multipliers indicate a discord between the minimizer of the potential and the global minimizer. Activating variables with high Lagrange multipliers provides the opportunity to resolve this discord in future inference using updated evidence, and thus find a higher-probability configuration. Nonetheless, since the Lagrange multipliers are artifacts of the previous epoch, there is the possibility that they might not identify volatile variables in the current epoch.

Activation Algorithms

From this analysis of the ADMM, we introduce two activation algorithms for online collective inference, both of which produce a ranking over all variables. The key difference between these algorithms is whether they operate independently of the updates to the evidence. We differentiate between "agnostic activation" and "relational activation". Agnostic activation scores variables at inference time based on their susceptibility to change in future inferences. In contrast, relational activation has access to both the updated set of evidence and the previous inference state.

Each approach has different advantages. Agnostic activation runs concurrently with inference, which provides a performance advantage since the scoring algorithm does not delay a future run of inference. However, this technique has slower responsiveness to updated evidence since scores are only updated after the inference is run. Relational activation can respond to newly-arrived evidence and choose variables based on their relationship to the new evidence. Yet, this requires running activation before inference, potentially introducing a delay in updated inference.

Both activation algorithms output a ranking of the variables, which requires a scoring function. The scoring function uses the ADMM features described in the previous section. Our first scoring function, VALUE, uses the consensus value of the variable ($\mathbf{y}(i)$) and measures uncertainty with the function $1 - |0.5 - \mathbf{y}(i)|$. The second scoring function, WLM, combines the potential weight (w_g) and Lagrange multipliers ($\boldsymbol{\alpha}_g(i)$). One insight from our analysis of ADMM is that the magnitudes of the Lagrange multipliers associated with each variable provide a useful signal for determining activation. We further intuit that the weight of the associated potential can be used to scale the importance of the Lagrange multiplier. Thus, for each variable, we define a set of weighted Lagrange multiplier magnitudes, $\mathcal{A}_w(i) \triangleq \{|w_g \boldsymbol{\alpha}_g(i)|\}$. To obtain a single scalar score, we take the maximum value of $\mathcal{A}_w(i)$.

The agnostic activation algorithm sorts the variables by their associated scores, irrespective of the new evidence. The RELATIONAL algorithm combines the score with information about the new evidence. Using the existing ground model, the algorithm first identifies all ground potentials that depend on the new evidence. Then, using these ground potentials as a seed set, the algorithm performs a breadth-first search of the factor graph and adds the variables involved in each factor it encounters to the activation order. Traversing the factor graph can quickly identify many candidate variables, so we prioritize variables in the frontier using the scoring function.

The ranking output by either agnostic or relational activation lets us prioritize which variables to activate. Given a budget for the number or percentage of variables to infer, we activate a corresponding number of variables from the ranking. The remaining variables are constrained to their previously inferred values. We selectively ground the model, including only those rules that involve an activated variable. Following inference on the ground model, we use the updated optimization state to produce new scores.

When an inactive variable is treated as a constant, it does not have any associated Lagrange multipliers, and therefore will not have any features in the next round of inference. Therefore, instead of treating fixed variables as constants, we introduce them as constrained variables in the optimization. This allows us to still produce features for inactive variables to capture the discrepancy between the constrained value and the value of local copies in the groundings of activated variables.

Our implementation of the agnostic activation algorithm is extremely efficient; all necessary features are byproducts of the inference optimization. Once scores are computed and the activated atoms are selected, the optimization state can be discarded to avoid additional resource commitments. In relational activation, scoring is similarly efficient, but there is an additional overhead of preserving the ground model to allow fast traversal of the factor graph. By selectively grounding the model we replace queries that scan the entire database, potentially many times, with precise queries that exploit indices for faster performance. Finally, selectively activating atoms produces an optimization objective with fewer terms, allowing quicker optimization.

Online Inference for KGI

To conclude, we sketch how to combine knowledge graph identification with our activation algorithms for online inference, and identify open research questions that confront our ongoing research. While we are unable to include experimental results in this draft, we describe the experiments we are pursuing and intend to include preliminary results (and source code) from our experiments in the camera-ready version of this paper.

Extending knowledge graph identification to cope with new extractions, such as those that NELL generates on a daily basis, is possible using the machinery we have introduced for online inference. The activation algorithms presented here have shown promise in our prior research. These algorithms leverage uncertainty and discord during inference to effectively choose variables to activate in future rounds of inference. What are the ramifications of these algorithms for knowledge graph identification?

In NELL, uncertainty is often the result of inadequate information; extractions with little supporting evidence are assigned confidence 0.5, making them ideal candidates for the VALUE algorithm to refine in future inference. Conversely, high discord, and the correspondingly high Lagrange multipliers, are a likely symptom of strong disagreement between extractions and ontological constraints such as mutual exclusion. Re-evaluating facts that disagree via the WLM algorithm can help improve the precision of a knowledge graph. Finally, as new evidence accumulates the greatest opportunity is to infer and correct related facts, such as those activated by the RELATIONAL algorithm. Each of these algorithms has a potential strength for online knowledge graph identification.

We are testing our online inference methods on data from 20 iterations of a new NELL instance. The NELL instance produces approximately 150K new extractions in each iteration, and the 20th iteration has 3.5M total extractions. In our previous experience with knowledge graph identification, we have observed that the complete knowledge graph can be three times as large as the extraction graph as a consequence of inferences from ontological constraints. Our goal is to compare the results of full inference to approximate inference using the activation strategies detailed here and demonstrate fast inference with low regret. We are committed to releasing source code and data for our experiments, and we anticipate that this problem setting will stimulate the StaRAI community to pursue challenging lifelong learning problems.

While we believe applying our online inference methods to knowledge graph identification will be successful, we also acknowledge that constructing knowledge graphs confronts the limits of our theoretical bounds. One limitation of our existing work is that it makes a closed-world assumption, providing a valid bound when the inferred variables remain fixed. In lifelong learning tasks, such as knowledge graph construction, the variables inferred will naturally grow over time as new evidence is encountered. Providing bounds for the performance of online inference in a setting where arbitrary variables (and arbitrary statistical dependencies) can be introduced at any time presents a difficult theoretical quandary.

References

Bach, S. H.; Broecheler, M.; Getoor, L.; and O'Leary, D. P. 2012. Scaling MPE Inference for Constrained Continuous Markov Random Fields with Consensus Optimization. In *NIPS*.

Bach, S. H.; Huang, B.; London, B.; and Getoor, L. 2013. Hinge-loss Markov random fields: Convex inference for structured prediction. In *UAI*.

Bach, S. H.; Broecheler, M.; Huang, B.; and Getoor, L. 2015. Hinge-loss Markov random fields and probabilistic soft logic. arXiv:1505.04406 [cs.LG].

Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; and Eckstein, J. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends Machine Learning* 3(1):1–122.

Carlson, A.; Betteridge, J.; Kisiel, B.; Settles, B.; Hruschka, E. R.; and Mitchell, T. M. 2010. Toward an Architecture for Never-Ending Language Learning. In *AAAI*.

Dong, X. L.; Gabrilovich, E.; Heitz, G.; Horn, W.; Lao, N.; Murphy, K.; Strohmann, T.; Sun, S.; and Zhang, W. 2014. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In *KDD*.

Etzioni, O.; Banko, M.; Soderland, S.; and Weld, D. S. 2008. Open Information Extraction from the Web. *Communications of the ACM* 51(12).

Jiang, S.; Lowd, D.; and Dou, D. 2012. Learning to Refine an Automatically Extracted Knowledge Base Using Markov Logic. In *ICDM*.

Namata, G. M.; Kok, S.; and Getoor, L. 2011. Collective Graph Identification. In *KDD*.

Niu, F.; Zhang, C.; Ré, C.; and Shavlik, J. 2012. Elementary: Large-scale knowledge-base construction via machine learning and statistical inference. *Int. J. Semantic Web Inf. Syst.* 8(3).

Pujara, J.; Miao, H.; Getoor, L.; and Cohen, W. 2013. Knowledge graph identification. In *ISWC*.

Pujara, J.; London, B.; and Getoor, L. 2015. Budgeted online collective inference. In *UAI*.

Suchanek, F. M.; Kasneci, G.; and Weikum, G. 2007. Yago: A Core of Semantic Knowledge. In *WWW*.