Context-Aware Online Collective Inference for Templated Graphical Models

Charles Dickens [*1] **Connor Pryor** [*1] **Eriq Augustine** [1] **Alex Miller** [1] **Lise Getoor** [1]

Abstract

In this work, we examine *online collective inference*, the problem of maintaining and performing inference over a sequence of evolving graphical models. We utilize *templated graphical models* (TGM), a general class of graphical models expressed via templates and instantiated with data. A key challenge is minimizing the cost of instantiating the updated model. To address this, we define a class of exact and approximate context-aware methods for updating an existing TGM. These methods avoid a full re-instantiation by using the context of the updates to only add relevant components to the graphical model. Further, we provide stability bounds for the general online inference problem and regret bounds for a proposed approximation. Finally, we implement our approach in probabilistic soft logic, and test it on several online collective inference tasks. Through these experiments we verify the bounds on regret and stability, and show that our approximate online approach consistently runs two to five times faster than the offline alternative while, surprisingly, maintaining the quality of the predictions.

1. Introduction

In many practical machine learning settings it is common for both the data and structure of the model to change incrementally over time. Incoming data may provide additional evidence or add to a set of predictive targets a system has to infer, and new information or a shifting context may alter a model's defining parameters or underlying structure. For instance, a new product review may change recommendations made for the reviewer and users who bought related products, or a recent event could impose constraints on a product's availability which were previously not considered by a predictor. These are two of many examples of problems that are both online and require joint predictions.

Online inference is especially challenging for structured prediction settings (Baki et al., 2007). Structured prediction algorithms utilize the underlying relational properties of the data and problem domain to improve predictive performance and typically require collective (i.e., joint) inference over a probabilistic model. However, updates to the evidence and dependency structure of these methods can have cascading effects on the predictions that are expensive to perform. Updating inference is a long-standing problem in the machine learning community, and there is a vast body of literature on the topic for graphical models subject to structural updates (Buntine, 1991; Friedman & Goldszmidt, 1997; Li et al., 2006; Acar et al., 2009; Sümer et al., 2011) and for dynamic models (Murphy, 2002; Nodelman et al., 2002).

In this work we develop theory and methodologies for updating maximum a posteriori (MAP) estimates in undirected graphical models with density functions belonging to log-concave exponential families. We address the scalability of online inference by proposing exact and approximate model instantiation methods which leverage the context of an existing model and structured updates. Then, we develop a novel framework for analyzing the stability of MAP states subject to changes in the model evidence and structure. The stability results are applied to derive bounds on the distance to optimality of warm start MAP states and a measure of regret that is incurred by performing inference on an approximated model.

Our work is most closely related to Pujara et al. (2015), however, here the definition of online collective inference is much more general, supporting the introduction of new random variables and changes to the structure of the graphical model. Further, we develop a theory and method for performing exact inference on evolving models, as opposed to approximate budgeted inference. We define the problem of online collective inference using the framework of *templated graphical models* (TGMs). In this framework, graphical models are specified by functions of relations and attributes of entities in a dataset, typically expressed as weighted logical rules. Then, *online collective inference* is the task of performing inference on a series of TGMs related by sequences of updates to the dataset and dependency

[*]Equal contribution [1]Department of Computer Science and Engineering, University of California Santa Cruz, California, United States. Correspondence to: Charles Dickens <cadicken@ucsc.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

structure.

Our key contributions are as follows: 1) we define the problem of online collective inference using the framework of templated graphical models, 2) we propose principled approximations to updating an existing model, 3) we analyze the stability of MAP states of models subject to sequences of model updates, 4) we bound the loss incurred by performing approximate model updates, 5) we implement an online collective inference system using probabilistic soft logic (Bach et al., 2017), and 6) through experiments over three tasks, *online recommendation*, *online demand forecasting*, and *online model selection*, we show that our approximate online approach consistently provides a up to a 5 times speedup over an offline variant while, surprisingly, maintaining the quality of the predictions.

2. Online Collective Inference

A key challenge of online collective inference is expressing how two models are related and modified. To address this we leverage a general framework for defining probability distributions, referred to as *templated graphical models* (TGM) (Koller & Friedman, 2009). With this framework, we formally define online collective inference.

2.1. Templated Graphical Models

A TGM encodes dependencies between relations and attributes of entities in a domain using functions called *template factors* with arguments referred to as *template variables*. Template factors are commonly expressed as weighted logical rules, for example:

$$w : \textsc{Likes}(\mathtt{P}_1, \mathtt{P}_2) \rightarrow \textsc{Knows}(\mathtt{P}_1, \mathtt{P}_2) \tag{1}$$

This template factor represents the idea that entities who like each other will often know each other.

Every template variable is associated with a range. For instance, the range of LIKES is {LIKES(Alice, Bob), LIKES(Bob, Charlie)} and the range of KNOWS is {KNOWS(Alice, Bob), KNOWS(Bob, Charlie)}. These ranges are subsets of a provided dataset and define the random variables in our domain. Template factors are instantiated by realizing combinations of template variable instantiations. With the above templates and data, the resulting set of instantiated template factors is:

$$w : \textsc{Likes}(\mathit{Alice}, \mathit{Bob}) \rightarrow \textsc{Knows}(\mathit{Alice}, \mathit{Bob})$$
$$w : \textsc{Likes}(\mathit{Bob}, \mathit{Charlie}) \rightarrow \textsc{Knows}(\mathit{Bob}, \mathit{Charlie})$$

Given the instantiated set of template factors, a TGM defines a joint probability distribution over the instantiated template variables. More formally,

Definition 1 (Template Variables). *A template variable V is a function of entity variables $V(e_1, \cdots, e_k)$, with a range denoted by $Val(V)$.*

Template variables are related by template factors, which, when instantiated, are referred to as *potentials*.

Definition 2 (Template Factors, Potentials). *A template factor τ defines a mapping from a subset of a cartesian product of template variable ranges, $\Gamma \subseteq Val(V_1) \times \cdots \times Val(V_n)$, to $\mathbb{R}$. Given a tuple of random variables $R \in \Gamma$, we use $\phi(R)$ to denote an instantiated template factor, referred to as a potential.*

For the template factor defined earlier, the potential for the weighted rule can be logical (mapping to 0 or 1) or represent the distance to satisfaction (used later in Section 5). Generating a set of potentials by realizing every tuple of random variables in the union of the domains of the template factors is referred to as *grounding* (Section 3). A TGM is defined as a collection of one or more template factors, a set of potentials, and the set of random variables present in the dataset.

Definition 3 (Templated Graphical Model). *Given a set of template factors T with a corresponding set of random variables Z and potentials $\Phi = \{\phi_1, \cdots, \phi_m\}$ a templated graphical model (TGM) defines the joint probability distribution* [1]

$$P(Z) = \frac{1}{\mathcal{Z}} \prod_{i=1}^{m} \phi_i(Z) \tag{2}$$

where $\mathcal{Z} = \int_z \prod_{i=1}^{m} \phi_i(Z = z)$ normalizes $P(Z)$. A TGM is denoted by the tuple $\mathbb{T} = (T, \Phi, P(Z), Z)$. [2]

Any distribution can be expressed in the form of Equation 2 for some set of potentials Φ. Throughout this paper we assume the random variables of a TGM, Z, are partitioned into observed random variables X and unobserved random variables Y. This partition defines a conditional distribution:

$$P(Y|X) = \frac{1}{\mathcal{Z}(X)} \prod_{i=1}^{m} \phi_i(Y, X) \tag{3}$$

where $\mathcal{Z}(X) = \int_y \prod_{i=1}^{m} \phi_i(Y = y, X)$. An equivalent way to express a TGM with a random variable partition and conditional distribution is $\mathbb{T} = (T, \Phi, P(Y|X), Y, X)$.

2.2. Online Collective Inference

TGMs provide a convenient means for systematically defining modifications to a graphical model, which are broken into fundamental steps called *model updates*.

[1] Abuse of notation: $\phi_i(\cdot)$ is written as a function of Z when it actually is defined as a function of a tuple of random variables from Z.

[2] This definition assumes continuous valued potential functions. Mass functions are defined by replacing the integral in the definition by summation.

Definition 4 (Model Update)**.** *Given the TGM* $\mathbb{T} = (T, \Phi, P(Y|X), Y, X)$*. We define a model update as one of the following:*

1. *Update the value of an observed variable* $x_i \in X$
2. *Add or delete a random variable* $x_i \in X$ *or* $y_i \in Y$
3. *Add or delete a template factor* $\tau_i \in T$

With TGMs and model updates we define the task of *online collective inference*.

Definition 5 (Online Collective Inference)**.** *Let* $\mathbb{T}_1 = (T_1, \Phi_1, P_1(Y_1|X_1), Y_1, X_1)$ *be a TGM. Then apply a series of updates that converts* T_1*,* X_1*, and* Y_1*, to* T_2*,* X_2*, and* Y_2*. Online collective inference is the task of instantiating the potentials* Φ_2 *using every* $\tau_i \in T_2$ *to get* $\mathbb{T}_2 = (T_2, \Phi_2, P(Y_2|X_2), Y_2, X_2)$ *and performing inference over the newly defined probability distribution* $P_2(Y_2|X_2)$*.*

A common inference task is obtaining a *maximum-a-posteriori (MAP) estimator* of the random variables. For a distribution $P(Y|X = \mathbf{x})$, a MAP estimator, $\mathbf{y}^*$, achieves the mode, i.e., $\mathbf{y}^* = \arg\max_{\mathbf{y}} P(Y = \mathbf{y}|X = \mathbf{x})$.

3. Model Instantiation

As mentioned in Section 2.1, a vital subproblem of online collective inference is generating potentials, i.e., grounding.

Definition 6 (Grounding)**.** *Let* $T = \{\tau_1, \cdots, \tau_s\}$ *and* $\Gamma_1, \cdots, \Gamma_s$ *be a set of template factors and corresponding domains. Grounding is the process of, for every* $\tau_i \in T$*, realizing all tuples of random variables* $(Z_1, \cdots, Z_{n^i}) \in \Gamma_i$*, and instantiating every potential* $\phi_{\hat{i}}(Z_1, \cdots, Z_{n^i})$*.*

Realizing every tuple of random variables in a domain, Γ, of a template factor, τ, is difficult as Γ can be large and a non-trivial subset of the full n-ary Cartesian product of the range of template variables associated with τ. Reductions to this set, such as removing potentials that will not modify the optimal setting of the random variables, i.e., *trivial* potentials, are typically performed. There has been a considerable amount of research on identifying trivial potentials and designing scalable algorithms for grounding (Richardson & Domingos, 2006; Bach et al., 2017). We expand upon these efforts to scale the grounding process in an orthogonal direction through online-specific improvements.

3.1. Context-Aware Grounding

Rather than re-perform the entire grounding process to instantiate an updated model, we introduce *context-aware grounding*. This class of methods leverage the practical observation that model updates will typically preserve much of the initial TGM's existing structure. Context-aware grounding makes a minimal edit, adding and deleting potentials from the initial potential set.

Definition 7 (Context-Aware Grounding)**.** *Let* $\mathbb{T}_1 = (T_1, \Phi_1, P(Z_1), Z_1)$ *and* $\mathbb{T}_2 = (T_2, \Phi_2, P(Z_2), Z_2)$*, be two TGMs. A context-aware grounding instantiates* Φ_2 *by grounding the set* $\Phi_+ = \Phi_2 \setminus \Phi_1$ *and removing* $\Phi_- = \Phi_1 \setminus \Phi_2$*, i.e.,* $\Phi_2 = \Phi_+ \cup (\Phi_1 \setminus \Phi_-)$*.*

We derive a tight upper bound on the number of new potentials generated by a context-aware grounding.

Theorem 1. *Let* $\mathbb{T} = (T, \Phi, P(Z), Z)$ *be a TGM, and* T_+*,* Z_+ *be the sets of template factors and random variables added after a series of model updates, respectively. Define a set of context template factors* $T_c \subseteq T_+ \cup T$ *such that each* $\tau_i \in T_c$ *is a function of at least one tuple of random variables containing some* $Z_j \in Z_+$*. Build the set of context variables for each* τ_i*, denoted as* Z_{τ_i}*, to be the complete set of variables* $Z \cup Z_+$ *if* $\tau_i \in T_+$*, or* Z_+*, otherwise. Then*

$$|\Phi_+| \leq \sum_{\tau_i \in \mathrm{T}_c} \sum_{j=1}^{|\tau_i|} \binom{|\tau_i|}{j} (|Z_{\tau_i}|^j \cdot |Z|^{|\tau_i|-j}) \tag{4}$$

where $|\tau_i|$ *is the number of template variables defining* τ_i*.*

3.2. Approximate Context-Aware Groundings

Theorem 1 shows that the number of potentials instantiated by a context-aware grounding will grow exponentially with respect to the number of context variables and context template factors. Thus, generating all of the new potentials can be a costly operation that may not be viable in all settings. We therefore propose *approximate context-aware grounding*:

Definition 8 (Approximate Context-Aware Grounding)**.** *Let* Φ_+ *be a set of potentials generated from a context-aware grounding. An approximate context-aware grounding is any process which generates a set of potentials* $\tilde{\Phi}_+$ *that is a proper subset of* Φ_+*, i.e.,* $\tilde{\Phi}_+ \subset \Phi_+$*.*

An example of an approximate context-aware grounding is one that generates potentials which contain less than some threshold number of context variables, say κ. This is motivated by observing that this restricts the upper limit of the inner summation in the bound of Theorem 1. This results in a degree κ polynomial growth rate with respect to the number of context variables, context template factors, and random variables.

4. Stability and Regret of Online Inference

In this section we analyze twp aspects, stability and regret, of the MAP states of TGMs belonging to a log-concave and continuous exponential family (Wainwright & Jordan, 2008). This implies potentials in Equation 3 are constrained to the form: $\phi(\mathbf{y}, \mathbf{x}) = \exp(-\theta\psi(\mathbf{y}, \mathbf{x}))$ for some continuous convex function $\psi(\cdot)$ and a non-negative real valued parameter vector θ. The MAP inference problem of a TGM in such a class of distributions with m potential functions is:

$$\max_{\mathbf{y}} P(Y = \mathbf{y}|X = \mathbf{x}) \equiv \min_{\mathbf{y}} \sum_{i=1}^{m} \theta_i \psi_i(\mathbf{y}, \mathbf{x}) \qquad (5)$$

Further, we let $H(\mathbf{y}, \mathbf{x}) = \sum_{i=1}^{m} \theta_i \psi_i(\mathbf{y}, \mathbf{x})$ denote the *MAP objective*. Formally, for all TGMs in this section, we make the following assumption.

Assumption 1. *$P(Y|X)$ is a member of a log-concave exponential family.*

4.1. Stability

Broadly speaking, stability ensures that small changes to the input result in bounded variations in the output. A result of Assumption 1 is that the MAP objectives are convex, which follows from the definition of log-concavity. Specifically for stability analysis, an additional useful assumption is strong convexity. Strong convexity of a function ensures a unique minimizer and thus stability of MAP states simplifies to analyzing the distance between points rather than sets.

Definition 9 (Strong Convexity)**.** *$f : \mathbb{R}^n \to \mathbb{R}$ is α-strong convex iff for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$ and $g \in \partial f(\mathbf{x}_1)$* [3]

$$f(\mathbf{x}_2) - f(\mathbf{x}_1) \geq g^T(\mathbf{x}_2 - \mathbf{x}_1) + \frac{\alpha}{2}\|\mathbf{x}_2 - \mathbf{x}_1\|_2^2 \qquad (6)$$

Strong convexity of the MAP objective can be ensured by squared L_2 regularization. This is a consequence of the fact that an $\alpha > 0$ parameterized squared L_2 regularizer is α-strong convex, and strong convexity is preserved when summed with convex functions.

Lemma 1. *Let $l = f + h$ where f is convex and h is α-strong convex. Then l is α-strong convex.* [†]

The strong convexity condition yields an upper bound on the distance a point is from a minimizer.

Lemma 2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be α-strong convex. Suppose $\mathbf{x}^*$ is a minimizer of f, for all $\mathbf{x} \in \mathbb{R}^n$ and $g \in \partial f(\mathbf{x})$* [†]

$$\|\mathbf{x}^* - \mathbf{x}\|_2 \leq \frac{2}{\alpha}\|g\|_2$$

This lemma yields a bound on the distance of any setting of the unobserved random variables to the unique minimizer of an L_2 regularized MAP objective. We thus have a direction for analyzing the stability of MAP states by bounding the magnitude of gradients of MAP states after a sequence of model updates. Our approach leverages an additional assumption bounding the rate of change of gradients in the MAP objective of the updated TGM, H_2, i.e., β-smoothness.

Definition 10 (β-Smoothness)**.** *$f : \mathbb{R}^n \to \mathbb{R}$ is β-smooth over $\Omega \subseteq \mathbb{R}^n$ iff for all $\mathbf{x}_1, \mathbf{x}_2 \in \Omega$*

$$\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\|_2 \leq \beta\|\mathbf{x}_1 - \mathbf{x}_1\|_2 \qquad (7)$$

Assumption 2. *The MAP objective $H(\mathbf{y}, \mathbf{x})$ of the distribution $P(Y|X)$, is β-smooth as a function of both arguments $\mathbf{y}$ and $\mathbf{x}$.*

To account for the complexity of a sequence of updates relating the models $\mathbb{T}_1$ and $\mathbb{T}_2$, we define the delta model as the set of potentials which are added and removed by a sequence of model updates.

Definition 11 (Delta Model)**.** *Given the TGMs $\mathbb{T}_1 = (T_1, \Phi_1, P(Z_1), Z_1)$ and $\mathbb{T}_2 = (T_2, \Phi_2, P(Z_1), Z_2)$. Define $\Phi_- = \{\exp(\theta\psi_i) | \phi_i = \exp(-\theta\psi_i) \in \Phi_1 \setminus \Phi_2\}$ and $\Phi_+ = \Phi_2 \setminus \Phi_1$. The potential set $\Phi_\triangle = \Phi_- \cup \Phi_+$ is a delta model with a corresponding MAP objective $H_\triangle = \sum_{\phi \in \Phi_\triangle} -\log(\phi_i(\mathbf{y}, \mathbf{x}))$.*

Applying the lemmas and assumptions introduced in this section, we derive the following bound on the distance between MAP states for two TGMs.

Theorem 2. *Let $\mathbb{T}_1 = (T_1, \Phi_1, P_1(Y_1|X_1), Y_1, X_1)$ and $\mathbb{T}_2 = (T_2, \Phi_2, P(Y_2|X_2), Y_2, X_2)$ be two TGMs defining the MAP objectives H_1 and H_2. Suppose $P_1(Y_1|X_1)$ and $P_2(Y_2|X_2)$ satisfy Assumption 1 and $P_2(Y|X)$ satisfies Assumption 2. Let $\Phi_\triangle$ be the delta model relating $\mathbb{T}_1$ to $\mathbb{T}_2$ with MAP objective $H_\triangle$. Denote the vectors of observed random variable values of $\mathbb{T}_1$ and $\mathbb{T}_2$ as $\mathbf{x}_1 \in \mathbb{R}^{|X_1|}$ and $\mathbf{x}_2 \in \mathbb{R}^{|X_2|}$, and MAP states $\mathbf{y}_1^* = \arg\min_{\mathbf{y}} H_1(\mathbf{y}, \mathbf{x}_1)$ and $\mathbf{y}_2^* = \arg\min_{\mathbf{y}} H_2(\mathbf{y}, \mathbf{x}_2)$. Let $\tilde{\mathbf{y}}_1^*$ and $\tilde{\mathbf{x}}_1$ be the vectors $\mathbf{y}_1^*$ and $\mathbf{x}_1$, such that values corresponding to deleted variables are removed, and values corresponding to new variables and changing partitions are added. Note, values for variables moving from the unobserved to the observed partition are the values from $\mathbf{y}_1^*$. Let δ to be the change in the observed variable values, i.e., $\delta = \|\tilde{\mathbf{x}}_1 - \mathbf{x}_2\|_2$. Then*

$$\|\tilde{\mathbf{y}}_1^* - \mathbf{y}_2^*\|_2 \leq 2\frac{\beta}{\alpha}\delta + \frac{2}{\alpha}\|\nabla_{\mathbf{y}} H_\triangle(\tilde{\mathbf{y}}_1^*, \tilde{\mathbf{x}}_1)\|_2 \qquad (8)$$

Proof. First note that

$$\delta = \|\tilde{\mathbf{x}}_1 - \mathbf{x}_2\|_2 = \left\| \begin{bmatrix} \tilde{\mathbf{y}}_1^* \\ \tilde{\mathbf{x}}_1 \end{bmatrix} - \begin{bmatrix} \tilde{\mathbf{y}}_1^* \\ \mathbf{x}_2 \end{bmatrix} \right\|_2$$

Thus β-Smoothness of H_2 implies

$$\begin{aligned} \beta^2\delta^2 &\geq \|\nabla H_2(\tilde{\mathbf{y}}_1^*, \tilde{\mathbf{x}}_1) - \nabla H_2(\tilde{\mathbf{y}}_1^*, \mathbf{x}_2)\|_2^2 \\ &= \|\nabla_{\mathbf{y}} H_\triangle(\tilde{\mathbf{y}}_1^*, \tilde{\mathbf{x}}_1) - \nabla_{\mathbf{y}} H_2(\tilde{\mathbf{y}}_1^*, \mathbf{x}_2)\|_2^2 \\ &\quad + \|\nabla_{\mathbf{x}} H_2(\tilde{\mathbf{y}}_1^*, \tilde{\mathbf{x}}_1) - \nabla_{\mathbf{x}} H_2(\tilde{\mathbf{y}}_1^*, \mathbf{x}_2)\|_2^2 \\ &\geq (\|\nabla_{\mathbf{y}} H_2(\tilde{\mathbf{y}}_1^*, \mathbf{x}_2)\|_2 - \|\nabla_{\mathbf{y}} H_\triangle(\tilde{\mathbf{y}}_1^*, \tilde{\mathbf{x}}_1)\|_2)^2 \\ \implies \beta\delta &\geq \|\nabla_{\mathbf{y}} H_2(\tilde{\mathbf{y}}_1^*, \mathbf{x}_2)\|_2 - \|\nabla_{\mathbf{y}} H_\triangle(\tilde{\mathbf{y}}_1^*, \tilde{\mathbf{x}}_1)\|_2 \end{aligned}$$

[3] $\partial f(\mathbf{x})$ here denotes the set of subgradients of f at $\mathbf{x}$.

[†] Proof available in (Shalev-Shwartz, 2012).

Rearranging terms we have

$$\beta\delta + \|\nabla_{\mathbf{y}} H_{\Delta}(\tilde{\mathbf{y}}_1^*, \tilde{\mathbf{x}}_1)\|_2 \geq \|\nabla_{\mathbf{y}} H_2(\tilde{\mathbf{y}}_1^*, \mathbf{x}_2)\|_2$$

Lastly, applying the α-strong convexity bound, Lemma 2, on $\|\tilde{\mathbf{y}}_1^* - \mathbf{y}_2^*\|_2$ we have

$$\|\tilde{\mathbf{y}}_1^* - \mathbf{y}_2^*\|_2 \leq 2\frac{\beta}{\alpha}\delta + \frac{2}{\alpha}\|\nabla_{\mathbf{y}} H_{\Delta}(\tilde{\mathbf{y}}_1^*, \tilde{\mathbf{x}}_1)\|_2$$

□

4.1.1. Warm-Starts

A direct application of the above theorem motivates the notion that a related TGM's MAP estimator can be used to seed inference in a state closer than random initialization. We define a *warm-start*, $\tilde{\mathbf{y}}_1^*$, for a TGM $\mathbb{T}_2$ as the augmented MAP state of a related TGM $\mathbb{T}_1$. A *cold-start*, $\mathbf{y}_{\text{cold}}$, is drawn uniformly at random over the domain of the random variable values. If the MAP objectives of the distributions defined by $\mathbb{T}_1$ and $\mathbb{T}_2$ satisfy the necessary assumptions for Theorem 2, then the bound in the theorem can be applied to prove that a warm-start is closer to $\mathbb{T}_2$'s MAP state than a cold-start in expectation. More formally, if it can be shown that a sequence of model updates results in a change in observed variable values $\delta = \|\tilde{\mathbf{x}}_1 - \mathbf{x}_2\|_2$ and a delta model gradient $\|\nabla_{\mathbf{y}} H_{\Delta}(\tilde{\mathbf{y}}_1^*, \tilde{\mathbf{x}}_1)\|_2$ such that $2\frac{\beta}{\alpha}\delta + \frac{2}{\alpha}\|\nabla_{\mathbf{y}} H_{\Delta}(\tilde{\mathbf{y}}_1^*, \tilde{\mathbf{x}}_1)\|_2 \leq E\left[\|\mathbf{y}_{\text{cold}} - \mathbf{y}^*\|_2\right]$, then Theorem 2 yields $E\left[\|\tilde{\mathbf{y}}_1^* - \mathbf{y}_2^*\|_2\right] \leq E\left[\|\mathbf{y}_{\text{cold}} - \mathbf{y}^*\|_2\right]$.

4.2. Regret Bounds

Regret in online optimization is typically defined as the difference between the total loss incurred by two competing settings of the random variable values, called hypotheses (Shalev-Shwartz, 2012). Typically in collective inference, the total loss of a hypothesis is defined as a function of the set of potentials. However, in our setting, it is possible for the set of potentials to be modified. For this reason it is important to explicitly specify the set of potentials functions over which the regret is being computed.

Definition 12 (Regret)**.** *Let Φ be a set of potential functions, $\mathcal{L}(\mathbf{x}; \Phi) : \mathbb{R}^n \to \mathbb{R}$ be a loss function parameterized by Φ, and $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$. The regret with respect to the two competing hypothesis, $\mathbf{x}_1$, and $\mathbf{x}_2$, and Φ is defined as*

$$\textit{Regret}(\mathbf{x}_1, \mathbf{x}_2; \mathbf{\Phi}) = \mathcal{L}(\mathbf{x}_1; \mathbf{\Phi}) - \mathcal{L}(\mathbf{x}_2; \mathbf{\Phi}) \tag{9}$$

For MAP inference, the loss function is the MAP objective, i.e., $\text{Regret}_H(\mathbf{y}_1, \mathbf{y}_2) = H(\mathbf{y}_1, \mathbf{x}) - H(\mathbf{y}_2, \mathbf{x})$.

We are specifically interested in bounding the MAP regret relative to MAP estimators for two distinct TGMs defined over the same set of random variables. To do this we reformulate the problem into that of analyzing the stability of MAP states of TGMs subject to model updates. We then apply Theorem 2 together with an added assumption bounding the rate of change of the MAP objective, namely L-Lipschitz continuity.

Definition 13 (L-Lipschitz Continuity)**.** *$f : \mathbb{R}^n \to \mathbb{R}$ is L-Lipschitz continuous over $\Omega \subseteq \mathbb{R}^n$ iff for all $\mathbf{x}_1, \mathbf{x}_1 \in \Omega$*

$$|f(x_1) - f(x_2)| \leq L\|x_1 - x_2\|_2 \tag{10}$$

Assumption 3. *The MAP objective $H(\mathbf{y}, \mathbf{x})$, of the distribution $P(Y|X)$, is L-Lipschitz as a function of $\mathbf{y}$ for any $\mathbf{x}$.*

Theorem 3. *Let $\mathbb{T}_1$ and $\mathbb{T}_2$ be two TGMs defined over the same random variables. Further, suppose the distributions defined by $\mathbb{T}_1$ and $\mathbb{T}_2$, $P_1(Y|X)$ and $P_2(Y|X)$, satisfy Assumption 1, and $P_2(Y|X)$ satisfies Assumption 2 and Assumption 3. Let Φ_Δ be the delta model relating $\mathbb{T}_1$ to $\mathbb{T}_2$ with MAP objective H_Δ. Then, let $\mathbf{y}_1^*$ and $\mathbf{y}_2^*$ be the MAP states of $\mathbb{T}_1$ and $\mathbb{T}_2$, respectively.*

$$\textit{Regret}_{H_2}(\mathbf{y}_1^*, \mathbf{y}_2^*) \leq 2\frac{L}{\alpha}\|\nabla_{\mathbf{y}} H_{\Delta}(\mathbf{y}_1^*, \mathbf{x})\|_2 \tag{11}$$

Proof. As $H_2(\mathbf{y}, \mathbf{x})$ is L-Lipschitz continuous,

$$\|H_2(\mathbf{y}_1^*, \mathbf{x}) - H_2(\mathbf{y}_2^*, \mathbf{x})\| \leq L\|\mathbf{y}_1^* - \mathbf{y}_1^*\|_2 \tag{12}$$

Then, since $\mathbf{y}_2^* = \arg\min_{\mathbf{y}} H_2(\mathbf{y}, \mathbf{x})$

$$H_2(\mathbf{y}_1^*, \mathbf{x}) - H_2(\mathbf{y}_2^*, \mathbf{x}) \leq L\|\mathbf{y}_1^* - \mathbf{y}_1^*\|_2 \tag{13}$$

Next, noting that $\delta = 0$ and applying Theorem 2 yields

$$H_2(\mathbf{y}_1^*, \mathbf{x}) - H_2(\mathbf{y}_2^*, \mathbf{x}) \leq 2\frac{L}{\alpha}\|\nabla_{\mathbf{y}} H_{\Delta}(\mathbf{y}_1^*, \mathbf{x})\|_2 \tag{14}$$

□

5. Probabilistic Soft Logic

Probabilistic soft logic (PSL) (Bach et al., 2017) is a language for defining a specific type of TGM. A PSL program is a collection of weighted logical statements and linear arithmetic inequalities which operate as the template factors. Logical potentials use a continuous relaxation known as *Łukasiewicz* logic to define a hinge loss. Arithmetic potentials are defined to be the distance to satisfaction of a linear inequality constraint.

PSL defines a distribution that is a member of an exponential family with potentials constrained to the form:

$$\phi(\mathbf{y}, \mathbf{x}) = exp(-w \max\{\ell(\mathbf{y}, \mathbf{x}), 0\}^p) \tag{15}$$

where $w > 0$ is inherited from the instantiating template factor, ℓ is a linear function, and $p \in \{1, 2\}$. Furthermore, all random variables are constrained to $[0, 1]$. This class of graphical model is referred to as a *hinge-loss Markov random field (HL-MRF)*. The resulting HL-MRF MAP inference problem is: $\min_{\mathbf{y}\in[0,1]} \sum_{i=1}^{m} -\log(\phi_i(\mathbf{y},\mathbf{x}))$.

A useful property of HL-MRFs is that they are log-concave distributions and hence have a convex MAP inference objective function. Moreover, the regularized HL-MRF MAP inference objective is L-Lipschitz continuous and α-strongly convex.

Lemma 3. *Suppose $P(Y|X)$ is an HL-MRF distribution. Let $\alpha > 0$, then the L_2 regularized MAP inference objective, $H(\mathbf{y},\mathbf{x}) + \frac{\alpha}{2}\|\mathbf{y}\|_2$, is α strongly convex and L-Lipschitz continuous for some $L > 0$.*

The proof follows from direct application of Lemma 1 and the fact that each potential ϕ_i is individually L_i-Lipschitz continuous, and hence the MAP inference objective, which is a positive sum of the potentials, is L-Lipschitz.

With an added assumption, the HL-MRF MAP inference objective is β-smooth.

Assumption 4. *$P(Y|X)$ is an HL-MRF distribution with strictly squared potentials, i.e., every potential has the form $\phi(\mathbf{y},\mathbf{x}) = exp(-w\max\{\ell(\mathbf{y},\mathbf{x}),0\}^2)$.*

Lemma 4. *Suppose $P(Y|X)$ is an HL-MRF distribution satisfying Assumption 4. Then the MAP inference objective $H(\mathbf{y},\mathbf{x})$ is β-smooth for some $\beta > 0$.*

The proof follows by observing that the magnitude of any second sub-gradient is bounded over the domain of the random variables.

Therefore, L_2 regularized HL-MRFs satisfying Assumption 4 and instantiated with PSL fulfill all requirements necessary to utilize the results from Section 4.

5.1. Projected Stochastic Subgradient Descent in PSL

The general MAP inference problem of HL-MRFs can be classified as linearly constrained, non-smooth convex finite-sum optimization. Projected stochastic subgradient descent was introduced as a method for performing HL-MRF MAP inference in Srinivasan et al. (2020). At every step, j, a random potential, ϕ_i, is sampled and a projected stochastic subgradient descent step is taken

$$\mathbf{y}_{j+1} = \Pi_{[0,1]^n}\left(\mathbf{y}_j + \eta g_{\mathbf{y}} \log(\phi_i(\mathbf{y}_i,\mathbf{x}_i))\right) \tag{16}$$

where $g_y \in \partial_{\mathbf{y}}(\log(\phi_i))$ and η is a step size hyperparameter.

Since the potentials of the HL-MRF take the specific form of a hinge-loss, a subgradient is easily computable. For all potentials ϕ_i, let a_i be an $n = |Y|$ dimensional vector such that $a_i[j]$ is the corresponding coefficient of the variable $\mathbf{y}[j]$ in the linear function ℓ_i defining ϕ_i. The potential subgradient used in the update is then:

$$g_{\mathbf{y}} = \begin{cases} 0 & \log(\phi_i) \geq 0 \\ -w_i a_i & p_i = 1 \wedge \log(\phi_i) < 0 \\ -2a_i\log(\phi_i) & p_i = 2 \wedge \log(\phi_i) < 0 \end{cases} \tag{17}$$

The projected subgradient update is run until the change in the MAP objective between epochs falls below a threshold tolerance, ϵ. In an online problem, setting ϵ to a fixed value introduces a subtle challenge. Observe that the objective function scales with the number of potentials, which changes after performing a model update. For this reason we scale ϵ by the number of potentials involved in inference, i.e., we set $\epsilon = \epsilon' \cdot m$ where ϵ' is a fixed scalar hyperparameter and m is the number of potentials.

6. Empirical Evaluation

In this section, we evaluate the performance of an online collective inference system implemented in PSL (*Online PSL*), and empirically validate the theory introduced in this work. We answer the following questions: Q1) Are model updates performed via context-aware groundings faster than grounding a new model? Q2) How does approximate context-aware grounding affect the runtime and performance of online collective inference? Q3) How well do the theoretical bounds introduced in Section 4 predict the movement of the MAP states ? Q4) In practice, are warm-starts typically closer to the updated model's MAP state than cold-starts?

6.1. Datasets and Models

We explore these questions on three online collective inference tasks: *online recommendation*, *online demand forecasting*, and *online model selection* [4]. These tasks showcase how online collective inference can be applied in practical settings, while demonstrating common model update patterns. The datasets for the tasks are as follows:

MovieLens – MovieLens is a movie recommendation dataset containing approximately 1M timestamped ratings made by 6K users on 4K movies (Harper & Konstan, 2015). Although often used as an offline recommendation dataset, timestamps in the data allow us to turn this dataset into an online recommendation problem using the following procedure. 10 splits are uniformly sampled, each using 70% of the original data. Each split is then partitioned into 20 time steps, where the first time step contains one third of the split's data and the rest are evenly partitioned. Two variants of the dataset are created for the online recommendation task: *MovieLens-Fixed* and *MovieLens-TimeSeries*. In *MovieLens-Fixed*, all ratings are always present as either observations or unknowns. At each time step, previously

[4] Data and code: https://github.com/linqs/dickens-icml21

unknown ratings are observed. In *MovieLens-TimeSeries*, ratings are added incrementally. At each time step, 20% of the unknowns from the previous time step are observed and all ratings in the new time step are unknown.

BikeShare – *BikeShare* is a dataset that contains information for 650k trips between 70 stations by customers of the bicycle sharing service, Bay Area Bike Share (Bay Area Bike Share, 2016). The task for this dataset is to predict the demand for bikes at each station. The data is divided into 10 overlapping splits, where each split contains one third of the original data. Splits are partitioned into 20 time steps, where the first time step contains one third of the split's data and the rest are evenly partitioned. At each time step, the demand for the previous time step becomes fully observed and the demand for the next time is added as unknown.

Epinions – *Epinions* is a trust prediction dataset with 2k users with 8.5k directed links representing whether one user trusts the other. The data is divided into 8 splits with the trust links partitioned into observed and unknown sets following the same procedure as Bach et al. (2017). Each split is then partitioned into 10 time steps, where the first time step contains the full model and each subsequent time step adds and removes templates from the model.

6.2. Methods

Across all tasks and data variants, we evaluate the performance of three methods for executing model updates and performing MAP inference. First, **Offline** is a standard PSL implementation that executes model updates by grounding the full model at each time step. Then, **Regret-Free** and **Regretful** are Online PSL implementations executing model updates via context-aware grounding and approximate context-aware grounding, respectively. The regretful models are grounded using the example described in Section 3.2 with $\kappa = 1$.

6.3. Performance and Regret Analysis

To address questions Q1 and Q2, we compare the performance of our online and offline methods across each dataset. Figure 1 shows the normalized MAP inference objective (top), a problem-specific evaluation metric (middle), and the cumulative time in seconds to obtain a MAP prediction at each time step (bottom). Because the MAP inference problem is strongly convex, the Online PSL implementation employing exact context-aware grounding yields the same predictions as the offline method.

First, we will discuss the results of the MovieLens-Fixed and Epinions experiments. In these experiments, both online methods perform significantly faster than the offline method, with more than a 4 times speedup on the MovieLens-Fixed dataset and over a 20 times speedup on the Epinions dataset. In these setting no variables are added or deleted, thus there is no significant difference between the exact and approximate online methods in terms of time, evaluation, and MAP objective performance (regret).

Next we examine the results of the MovieLens-TimeSeries and BikeShare datasets in Figure 1. Both of these datasets operate similarly, where unknowns in the previous time step become observed as well as unknowns being introduced in the new time step. In both problems, the regretful online method performs $2 - 5$ times faster than the regret-free online method, but this comes at the cost of an inferior MAP inference objective and evaluation score.

6.4. Stability Analysis

To address question Q3 and Q4, we estimate model complexity parameters of the TGMs instantiated by PSL and measure the movement of variables with warm-start and cold-start initializations. Figure 2 shows the measured variable movement on all problems. Also shown in the figure are two empirical estimates of the warm-start bound derived in Section 4, one pessimistic and one optimistic. As optimization is performed, β and α estimates are created by sampling the rate of change in the gradients. Both the optimistic and pessimistic warm-start bounds take β to be the maximum observed rate of change of gradient. Then, the α estimate is the average of the rates in the optimistic bound and the average between the regularization parameter of the model, 0.1, and the minimum observed rate for the pessimistic bound. We also plot the expectation of the cold-start variable movements which we assume to be the expected distance between two randomly sampled points in a hypercube (Anderssen et al., 1976). Moreover, a trivial upper bound on the possible movement considering only the $[0, 1]$ variable box constraints enforced by PSL is plotted.

The expected and actual cold-start movements are aligned in all experiments except BikeShare. This behavior suggests that the assumption that the distance between a random initialization and the MAP state of a TGM instantiated by PSL is reasonable. The BikeShare exception can be explained by the fact that truth values in this setting are mostly near zero.

Next, we see in the MovieLens-Fixed, MovieLens-TimeSeries, and Epinions experiments, the cold-start movement of the variables is significantly larger than that of the warm-start. This matches the intuition of the settings, as there are many unobserved variables shared between time steps that are already optimized for a subset of the potentials in the updated model. Then, as expected, the exception of this behavior is in the BikeShare experiments. Cold-start and warm-start initializations are in fact equivalent in BikeShare because all unobserved variables for a time step are introduced by the model update.

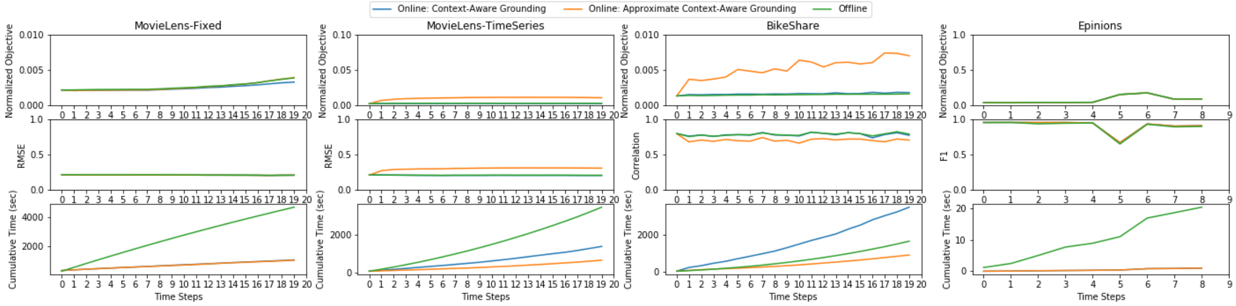

Figure 1. Comparison of normalized MAP inference objective (top), domain-specific evaluation metrics (middle), and runtime (bottom) of the three systems.

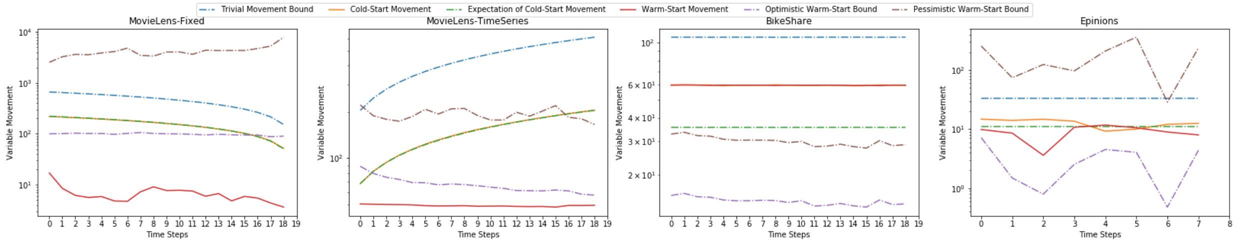

Figure 2. Comparison of warm-start and cold-start variable movements plotted with various theoretical and empirically estimated bounds.

In all experiments, the optimistic warm-start bound closely follows the actual warm-start movement, verifying theoretical predictions of Theorem 2 and hence Theorem 3. Furthermore, the optimistic warm-start bound consistently falls below the expected cold-start movement. The pessimistic bound on the other hand typically overestimates the warm-start variable movement by a large-margin, even predicting more movement than the trivial bound in some cases. The optimistic and pessimistic warm-start bounds do drop below the actual warm-start movement in both the BikeShare and Epinions experiments. This can be explained by the empirical nature of this experiment, the true α and β and parameters of this model are only estimated via a sampling.

7. Related Work

Online learning describes a class of machine learning methods where data arrives sequentially and the goal is to obtain the best predictor for future data using the most up-to-date information (Shalev-Shwartz, 2012). Online machine learning in the independent and identically distributed (IID) setting has been studied extensively (Mairal et al., 2010; Kushner & Yin, 2003; Williams & Zipser, 1989) and major advances have been made in creating efficient and scalable algorithms. Many online IID models can make updates to their predictor using the loss incurred from a single prediction (Bottou, 2010) or by updating a set of summary statistics (Cesa-Bianchi & Lugosi, 2006).

Updating graphical models is a long-standing problem in the machine learning community. There is substantial research in the area of updating Bayesian networks (Buntine, 1991; Friedman & Goldszmidt, 1997; Li et al., 2006), both with evolving structure and updating parameters. Likewise, there has been much work in the area of dynamic and sequential modeling, such as dynamic and continuous time Bayesian Networks (Murphy, 2002; Nodelman et al., 2002) and hierarchical hidden Markov models (Fine et al., 1998). Adaptive inference is another related area that aims to make efficient updates to the inference result of a general graphical model defined over discrete valued variables (Sümer et al., 2011).

The task of updating the MAP state of a TGM conditioned on evolving evidence was first considered in Pujara et al. (2015). In this setting the graphical model has a fixed dependency structure, i.e., no variables or potentials are added or deleted, and the MAP state is found using a technique which constrains the number of variables that may be updated to a predefined budget. The authors bound the *inference regret* induced by performing this type of approximate inference on an updated model. The definition of inference regret is defined as the normalized L_1 distance between a setting of the unobserved variables and the MAP state of a fixed model. Our work differs significantly in that the templated graphical model is not fixed and inference is not constrained by a budget.

The notion of collective stability, introduced by (London et al., 2013; 2014), measures the change in the output of a structured predictor subject to updates to a set of evidence. However, it does not consider updates which add or delete variables or potentials defining the graphical model. For this reason we introduced the notion of the delta model, an analytical tool for representing the differences in the structure and evidence of two TGMs.

8. Conclusions and Future Work

In this work, we develop a general methodology for defining and analyzing the problem of online collective inference using the framework of TGMs. To address the difficulty of model instantiation in online collective inference, we introduce a class of exact and approximate approaches for utilizing the context of an existing graphical model to instantiate an updated model without the need to fully rebuild it, referred to as context-aware grounding. We then use the complexity of a sequence of model updates to bound the possible change in the inferred variables values. These results are applied to upper bound the regret incurred by employing a proposed approximate context-aware grounding scheme. Our theoretical analysis is general enough to be used in any modeling framework which ultimately performs MAP inference over a distribution that is a member of a log-concave exponential family. Further, we show that assumptions made in our analysis are adhered to by models instantiated using the PSL framework. This makes our methods directly applicable to models over many domains ranging from bioinformatics (Kouki et al., 2019) to recommendation systems (Kouki et al., 2015). Finally, we implement an online collective inference system in PSL and verify our theoretical results. Moreover, the approximate method for executing model updates consistently yields a 2-5 times speedup over offline models in online tasks while still achieving nearly the same level of prediction performance.

A potential direction for future work is to introduce methods for reducing the size of the instantiated graphical model by summarizing or forgetting some of the model. This could further improve the speed ups over offline methods and reduce the memory requirements of the system. An orthogonal approach to forgetting that could also result in runtime improvements is implementing approximate inference techniques.

9. Acknowledgements

This work was partially supported by the National Science Foundation grants CCF-1740850, CCF-2023495, IIS-1703331, and DMS-2023495; AFRL; and the Defense Advanced Research Projects Agency.

References

Acar, U., Ihler, A., Mettu, R., and Sumer, O. Adaptive updates for map configurations with applications to bioinformatics. In *IEEE Workshop on Statistical Signal Processing (SSP)*, 2009.

Anderssen, R., Brent, R., Daley, D., and Morgan, P. Concerning $\int_0^1 \cdots \int_0^1 (x_k^2 + \cdots + x_k^2)^{1/2} dx_1 \cdots dx_k$ and a taylor series method. *SIAM Journal on Applied Mathematics*, 30(1):22–30, 1976.

Bach, S. H., Broecheler, M., Huang, B., and Getoor, L. Hinge-loss markov random fields and probabilistic soft logic. *Journal of Machine Learning Research (JMLR)*, 18(1):1–67, 2017.

Baki, G., Hofmann, T., Schölkopf, B., Smola, A., Taskar, B., and Vishwanathan, S. V. N. *Predicting Structured Data*. The MIT Press, 2007.

Bay Area Bike Share. OPEN DATA, 2016. URL http://www.bayareabikeshare.com/open-data.

Bottou, L. Large-scale machine learning with stochastic gradient descent. In *International Conference on Computational Statistics (COMPSTAT)*, 2010.

Buntine, W. Theory refinement on bayesian networks. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 1991.

Cesa-Bianchi, N. and Lugosi, G. *Prediction, learning, and games*. Cambridge University Press, 2006.

Fine, S., Singer, Y., and Tishby, N. The hierarchical hidden markov model: Analysis and applications. *Machine Learning (ML)*, 32:41–62, 1998.

Friedman, N. and Goldszmidt, M. Sequential update of bayesian network structure. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 1997.

Harper, M. and Konstan, J. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TIIS)*, 5(4), 2015.

Koller, D. and Friedman, N. *Probabilistic Graphical Models*. The MIT Press, 2009.

Kouki, P., Fakhraei, S., Foulds, J., Eirinaki, M., and Getoor, L. Hyper: A flexible and extensible probabilistic framework for hybrid recommender systems. In *ACM Conference on Recommender Systems (RecSys)*, 2015.

Kouki, P., Pujara, J., Marcum, C., Koehly, L., and Getoor, L. Collective entity resolution in multi-relational familial networks. *International Journal on Knowledge and Information Systems (KAIS)*, 61(3):1547–1581, 2019.

Kushner, H. and Yin, G. *Stochastic approximation and recursive algorithms and applications*. Springer, 2003.

Li, W., Beek, P. V., and Poupart, P. Performing incremental bayesian inference by dynamic model counting. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2006.

London, B., Huang, B., Taskar, B., and Getoor, L. Pac-bayes generalization bounds for randomized structured prediction. In *NIP Workshop on Perturbation, Optimization and Statistics*, 2013.

London, B., Huang, B., Taskar, B., and Getoor, L. Pac-bayesian collective stability. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, 2014.

Mairal, J., Bach, F., Ponce, J., and Sapiro, G. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research (JMLR)*, 12(1):19–60, 2010.

Murphy, K. *Dynamic Bayesian networks: representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

Nodelman, U., Shelton, C., and Koller, D. Continuous time bayesian networks. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2002.

Pujara, J., London, B., and Getoor, L. Budgeted online collective inference. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2015.

Richardson, M. and Domingos, P. Markov logic networks. *Machine Learning (ML)*, 62:107–136, 2006.

Shalev-Shwartz, S. Online learning and online convex optimization. *Foundations and Trends in Machine Learning (FTML)*, 4(2):107–194, 2012.

Srinivasan, S., Augustine, E., and Getoor, L. Tandem inference: An out-of-core streaming algorithm for very large-scale relational inference. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

Sümer, Ö., Acar, U., Ihler, A., and Mettu, R. Adaptive exact inference in graphical models. *Journal of Machine Learning Research (JMLR)*, 12:3147–3186, 2011.

Wainwright, M. and Jordan, M. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning (FTML)*, 1(1 - 2):1–305, 2008.

Williams, R. and Zipser, D. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989.