Fast and Accurate Prediction via Evidence-Specific MRF Structure

Veselin Stoyanov and Jason Eisner
Johns Hopkins University

Goal

We want to learn **evidence-specific structures** for MRFs. I.e., different examples may rely on different factors to predict the outputs.

Motivation

1. Data may be missing heterogeneously.

- Often the case with relational data.
- Our models can learn to rely more on the evidence and less on other inferred variables.

2. We want our models to perform fast test-time prediction.

- We are willing to trade-off some accuracy:
  - We will optimize a interpolation of the loss function and the speed:

    loss + λ.runtime

Approach

1. Gates

The formalism of gates [Minka and Winn, 2008] allows us to capture contextual (in)dependencies.

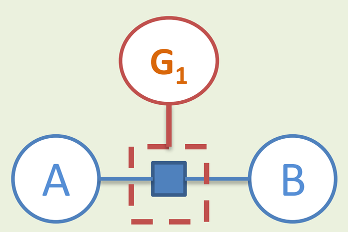
2. ERMA (Empirical Risk Minimization under Approximations)

We use the ERMA algorithm [Stoyanov, Ropson and Eisner, 2011] to jointly learn gate and MRF parameters by performing Empirical Risk Minimization.

Gates

Gates [Minka and Winn, 2008] can capture conditional (in)dependencies. A gate is a random variable that turns a factor on or off:

$$p_\theta(V=v) = \frac{1}{Z}\prod_{\alpha\in F}(\psi_\alpha(v_\alpha))^{g_\alpha}$$



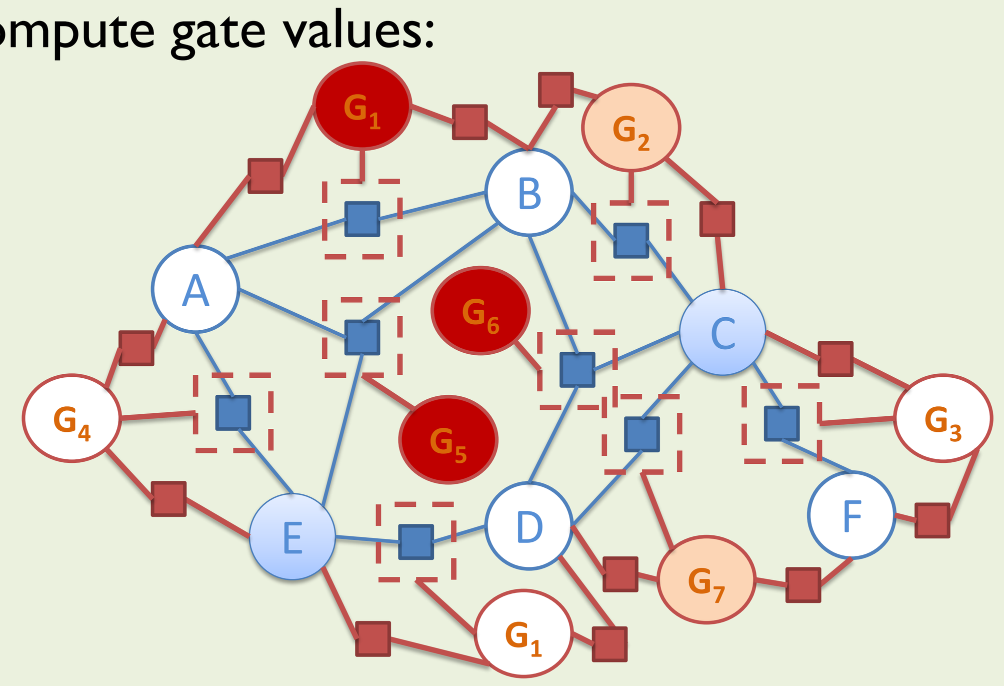
where $g_\alpha \in \{0,1\}$

In our model, gates are classifiers that decide whether a factor should be on/off based on the observation pattern (see results section).

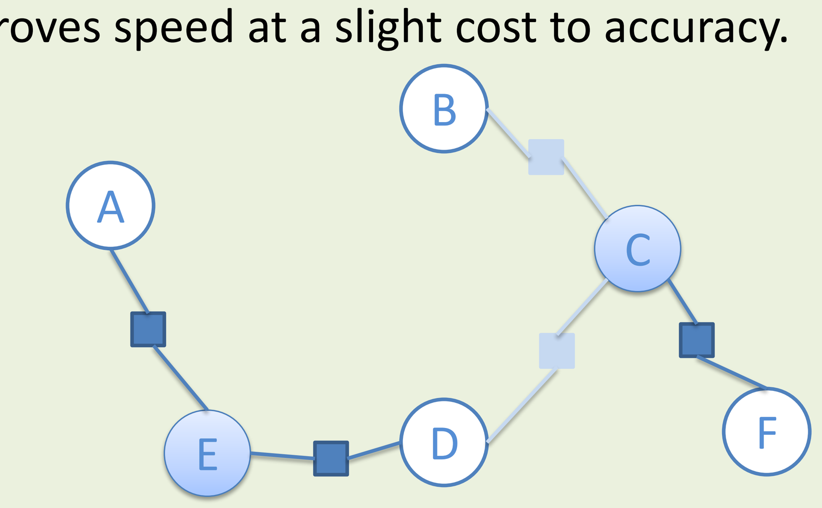
Gates can be partially on: we use the posterior marginal probability of a gate to damp down the messages through a given factor.

A two-step test-time procedure:

1. Compute gate values:



2. Turn off (or damp) factors and run inference. If the gate marginal probability is low, we prune the factor, which improves speed at a slight cost to accuracy.



ERMA

Empirical Risk Minimization under Approximations [Stoyanov, Ropson and Eisner, 2011] is an algorithm for learning in probabilistic graphical models by matching test-time conditions and performing empirical risk minimization.

ERMA utilizes **back-propagation of error** to compute gradients of the output loss. Empirical evidence shows ERMA **performs well on real-world problems** that require approximations [Stoyanov and Eisner, 2012] .

It is easy to extend ERMA so that it optimizes **gate parameters and MRF parameters jointly**.

ERMA implementation at: http://www.clsp.jhu.edu/~ves/software

Preliminary Results (Synthetic Data)

We generate a random 4-ary MRF and we sample training and test data. We forget the structure and start learning with a fully connected but binary graph. For each data point we randomly designate each r.v. as either input, hidden or output.

Training Procedure:

Our training objective is: loss + λ.runtime

We train by replacing the hard pruning threshold with a soft one, so we can compute the gradient of error and runtime w.r.t. the parameters.

Gate features:

Gate G_{AB} that controls the factor between r.v.'s A and B is conditioned on the values of A and B. Each r.v. (A or B) can be {0,1, hidden, output}.

Gate features are the conjunction of the factor id and the two variable values.

When both A and B are observed, we can just turn the factor off. *The total number of features is 4x4-2x2=12.*

We compare evidence-specific MRF to a L1-regularized model.