Guided inference: a protocol for teaching approximations

Anonymous Author 1
Unknown Institution 1

Anonymous Author 2
Unknown Institution 2

Anonymous Author 3
Unknown Institution 3

Abstract

We propose a protocol for modeling the exchange of advice between two approximations to a statistical model. In our protocol a "student" advertises marginal probabilities, and a "teacher" chooses an example state to show to the student. The student observes the model's unnormalized joint distribution at the new state and updates his beliefs. This interaction is repeated over a number of turns. We present results from experiments evaluating the ways in which the teacher might choose states to show the student.

1 Introduction

We often approach approximate inference, as with many other NP-hard problems, using algorithms which have a parallelizable structure. In designing such algorithms, we must specify how a problem should be divided into multiple sub-problems, and how the results should later be combined. In approximate inference, for instance with message passing methods, this subdivision is performed according to the structure of the model; whereas with sampling methods it is straightforward to execute multiple sampling runs in parallel and then average the results. A more flexible and intelligent algorithm might approach this subdivision by devoting separate threads to compiling approximate representations of a model, which could then be queried in various ways, for example to elicit conditioned marginals. Merging these separate representations would require some way of resolving their inevitable disagreements. This would in turn depend on a protocol by which one approximation could exchange information with another, for example about difficult areas of the model. In this paper we propose such a protocol. To illustrate its application, we perform experiments measuring the effectiveness of different ways of selecting which information should be transmitted.

One might imagine a number of ways in which this cooperative interaction could take place, but we use a simple scenario, in which a "teacher" shows a "student" examples consisting of states (full assignments of all the variables) of the model. His choice of these examples can be based on the student's marginals, to which he has access. We shall call this kind of interaction *guided inference*.

In anticipation of the results of our experiments, we shall consider these interactions as one potential application of the *conditional game*, proposed recently (Eaton, 2011), which is a procedure for comparing the accuracy of two approximations to a model. If the purpose of the conditional game is to be a protocol by which approximations can "argue" about areas of disagreement, the purpose of guided inference could be seen as a protocol through which these disagreements can be resolved. We find that the conditional game is useful for the second task as well.

2 Prior work

We relate our present work to previous research in machine learning.

2.1 Active learning

The problem of learning about a model by observing samples from some or all of its variables is common to most branches of machine learning. When some property of these samples is specified prior to sampling by the learner, in order to optimize the acquisition of information, then the process is called "active learning" (Tong, 2001) (two related topics are "query learning" and "optimal experiment design"). Active learning is perhaps the closest existing body of research to what we are calling "guided inference". As with other forms of learning, active learning can be employed to learn the parameters or structure of a model. Typically, an

Preliminary work. Under review by AISTATS 2011. Do not distribute.

active learner is allowed to specify values of some subset of the variables in a model, and the rest of the variables are then sampled from the true distribution conditioned on these assignments. Alternatively, a learner may have access to a set of unlabeled data points, from which he is allowed to select examples whose labels should be determined by an oracle. The learner presumably chooses these examples in such a way as to optimize some measure of his expected future accuracy in predicting labels for the rest of the data. Active learning can be seen as a way of modeling a scientist, who tries to learn about a system of interest by performing a series of experiments in which he constrains some aspects of the system's behavior and measures the effects of such interventions on other parts of the system.

Guided inference can be seen as a kind of active learning for inference. In the framework of active learning, a learner is trying to learn about a model given data points which are partly specified by the learner and partly drawn at random from a "true distribution". In guided inference, by contrast, the model is considered to be fully specified and the learner is trying to learn how to improve his approximation to this model. He does this by receiving "interesting" states of the model from a "teacher" whose approximation he wishes to emulate. The teacher of guided inference corresponds to the "true distribution" of active learning. Whereas the states in active learning are sampled partly at random, and this randomness is used to learn about the probability mass assigned by the model to different variable values, in guided inference there is not necessarily any randomness in the states, and the behavior of the model at the example states is inferred from the unnormalized joint rather than from any empirical distribution. In contrast to the sampling basis of active learning, guided inference is more similar to "curve fitting", where the values of an unknown function (the model's unnormalized joint) must be interpolated by a learner. What separates guided inference from fitting is its origins in probabilistic modeling: of interest is not the values of the function itself but their normalized sums, corresponding to variable marginals. This idea of integrating a probability distribution by fitting the density function to a set of samples has been explored before in Rasmussen and Ghahramani (2003), but their approach is more suited to low-dimensional, continuous integrands and does not address the problem of how to best choose samples (assuming them to be drawn randomly).

One point of similarity between the new guided inference framework and existing work in active learning is the use of a cost function to guide the choice of new states. In active learning, the learner has a sense of the areas of the model about which he would like to be more accurate. His preferences can be expressed in terms of an expected gain for each possible query he can make - perhaps he would like to minimize some form of entropy in his beliefs about the model. In guided inference, the teacher is trying to improve some measure of the student's error, and might use one of a number of cost functions in deciding which state to show the student at each turn (for examples, see the experiments in section 5).

2.2 Conditional games

The *conditional game* (CG) (Eaton, 2011) is a deterministic protocol which can be used to identify the more accurate of two approximations. It takes the form of a two-player zero-sum game, and its structure can be compared to that of a debate or a legal trial. Since the CG is played between two approximations and yields a state of the model at its conclusion, it can be used by the teacher to choose example states in the guided inference setting. In our experiments, we will find that the CG is effective for this purpose.

Here we give a brief review of the CG. Given a model with n variables, defined by a factor graph as in equation 3 below, play alternates between the "marginal player" (MP) and the "conditional player" (CP) over a total of n turns. At turn t the MP expresses marginals conditioned on the variable values previously selected by CP, say $Q(\dots|x^*_{i_{1:t-1}})$. The CP then chooses a new variable index i_t and value $x^*_{i_t}$. The variable x_{i_t} is fixed to take value $x_{i_t} = x^*_{i_t}$ for the rest of the game. Play finishes when the variables are all fixed, giving a complete assignment $x = x^*$. A quantity which we will call the "value" of the game is then defined in terms of x^* and Q:

$$V = \log \frac{\prod_{t=1}^{n} Q(x^*_{i_t}|x^*_{i_{1:t-1}})}{\prod_\alpha \psi_\alpha(x^*_\alpha)} \tag{1}$$

In a given game, CP will be trying to minimize V and MP to maximize it, or vice-versa. As long as the players have opposite goals, the game leads to sensible strategies. In particular, the optimal strategy of MP is to play exact marginals, so that $V = -\log Z$. Given an approximation R which CP trusts, a greedy strategy (for CP minimizing V) is to play

$$(i_t, x^*_{i_t}) = \operatorname*{argmin}_{\substack{(j,x_j) \\ j \notin i_{1:t-1}}} \frac{Q(x_j|x^*_{i_{1:t-1}})}{R(x_j|x^*_{i_{1:t-1}})} \tag{2}$$

If CP is trying to maximize V, then replace argmin with argmax above. In our experiments, the teacher plays as CP using this greedy strategy.

3 Guided inference

3.1 Preliminaries

We review our definition of approximate inference. We assume that we are given a model specified by a "factor graph", which is a general representation for statistical models. A factor graph defines a distribution over n variables $x := (x_1, \ldots, x_n)$ as a normalized product of non-negative factors ψ_α

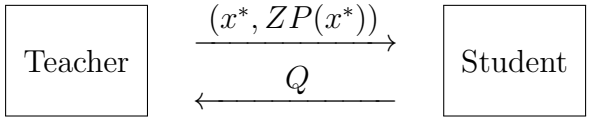
$$P(x) = \frac{1}{Z} \prod_\alpha \psi_\alpha(x_\alpha) \tag{3}$$

where α indexes a collection of sets of variables and Z is a normalizing constant called the "partition function". We will call $ZP(x) = \prod_\alpha \psi_\alpha(x_\alpha)$ the *unnormalized joint* (at state x). Note that ZP is always tractable. The problem of approximate inference is to estimate the marginals of the model:

$$P(x_i) = \frac{1}{Z} \sum_{x_{\setminus i}} \prod_\alpha \psi_\alpha(x_\alpha) \tag{4}$$

3.2 Protocol

One of the classical ways in which approximate inference can fail to produce good marginals is by failing to find all of the modes - areas of high probability - of a distribution. This is especially true for Markov Chain Monte Carlo (MCMC) methods, which may have difficulty in moving from one mode to another, but it is also true for deterministic methods such as Belief Propagation, whose accuracy seems to deteriorate on distributions with three or more modes. With these observations in mind, we designed the interaction cycle of guided inference to pass single states of the model x^* from teacher to student. It is easy to imagine using such a protocol to transfer knowledge about the modes of a distribution between two approximations, since each mode can be identified by one or more states.



The interaction cycle is formalized in the following pseudo-code.

Algorithm 1. *Guided inference*

Repeat for an arbitrary number of turns:

At turn m:

1. *Student proposes a distribution Q based on the example states and unnormalized joints he has seen so far: $\{(x^{*(i)}, ZP(x^{*(i)}))\}_{i=1:m-1}$*
2. *Teacher selects a new example $x^{*(m)}$ (perhaps in response to errors in the student's distribution)*

Our model of interaction is based on the key simplifying assumption that the only information which goes from the teacher to the student consists of examples of states of the model. The student has access to unnormalized joint probabilities ZP, but only evaluates them at the states recommended by the teacher; hence it is also possible to imagine the values $ZP(x^*)$ as being transmitted by the teacher together with the states x^*, as in the above diagram. This behavior is meant to capture the extreme of laziness, where the student knows the model specification, but not wanting to decide where to start with his analysis, only performs computations when prompted by receiving example states from the teacher. Importantly, the student does not get to know any of the teacher's opinions about the marginals or partition function. This means that the student is not limited by the accuracy of the teacher (and could learn just as easily from multiple teachers as from one).

4 Implementation

In a practical application of the guided inference framework, the "teacher" and "student" might both be approximations, and the student would be an approximation Q of a kind which can be parametrized by a set of states so that he learns to be more accurate with every state (example) shown to him by the teacher. But we do not yet concern ourselves with figuring out how an approximate student should make the best use of these examples for learning. *Here we restrict our attention to the question of how the teacher should choose examples to present to the student.* We are limiting ourselves, in other words, to investigating one half of the guided inference problem. A practical application of the conclusions we derive from our experiments would also require a solution to the second half, namely, a specification of the student's approximation. We leave this question for future work.

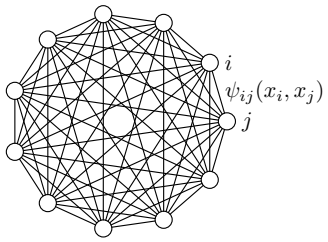
Now, if we were to experiment in a setting with an approximate Q, then it would be difficult to know whether to credit some property of the system to the teacher's or the student's approximate inference algorithm (which we are not particularly interested in) or to the teacher's protocol for choosing example points to present to the student (which is what we are interested in). In order to simplify the interpretation of our results, in our experiments we use two *exact* inference algorithms. The inference of the "teacher" is simply exact, while the student performs "exact" inference on a simple distribution over models, conditioned to agree with the observations of the unnormalized joint which

the teacher has shown to him.

A consequence of the exploratory nature of this research is that we are constrained to study small models on which exact inference is tractable. Our expectation is that the results will generalize to larger models as well.

4.1 Distributions over models

We now describe how the "exact" student of our experiments maintains and reasons about a distribution over models, given a set of example entries from the unnormalized joint distribution $ZP(x) = \prod_\alpha \psi_\alpha(x_\alpha)$. We will restrict ourselves to representing distributions over the potentials of fully connected binary pairwise factor graphs. (Thus, we do not consider the problem of reasoning about models of differing structure, e.g. by averaging over multiple hypotheses which specify different sized factors or different sparse connectivity.) We have the student represent the potentials of the graph as exponentials of normal random variables, $\psi_{ij}(x_i, x_j) = \exp(\beta W)$ where $W \sim N(0, 1)$. This is a conventional way of generating random models for experiments in approximate inference.



(5)

Initially the potentials are believed by the student to be sampled independently, but when he incorporates his set of observations of the unnormalized joint $\{(x^{*(i)}, ZP(x^{*(i)}))\}_{i=1:m-1}$, then correlations will be introduced in his beliefs. If he represents the log-potentials using a multivariate normal distribution, then these correlations can be represented in a covariance matrix, and after each observation the posterior of his beliefs will be in the same class as the prior (i.e., it is a conjugate prior). The observations

$$z^* = ZP(x^*) \tag{6}$$

are equivalent to

$$z^* = \prod_{jk} \psi_{jk}(x^*_{jk}) \tag{7}$$

$$\implies \log z^* = \sum_{jk} \log \psi_{jk}(x^*_{jk}) \tag{8}$$

The quantities $\log \psi_{jk}(x_j, x_k)$ are distributed according to a multivariate normal distribution with mean μ and variance Σ (indexed by $(j, k > j, x_{jk})$ and initialized to $\beta^2 I$) so this is a statement that some subset of the dimensions of this normal distribution should have a certain sum (namely $\log z$). A set of such constraints is in turn a special case of a linear constraint, say

$$B \cdot y = v \tag{9}$$

on draws y from a multivariate normal, where B is a matrix and v a vector. More specifically, in our experiments, y is indexed by $(j, k > j, x_j, x_k)$ and represents a vector of log potentials specifying the whole model, while B contains entries which are 0 or 1 according to whether a particular potential entry contributes to a given state, and v is a column vector of the log unnormalized joint entries corresponding to each example $x^{*(i)}$:

$$v_i = \log z^{*(i)} = \log ZP(x^{*(i)}) \tag{10}$$

Conditioning on this linear constraint is equivalent to transforming the mean and the variance of the multivariate normal distribution:

$$\mu' = \mu - \Sigma B^T (B\Sigma B^T)^{-1}(B\mu - v) \tag{11}$$

$$\Sigma' = \Sigma - \Sigma B^T (B\Sigma B^T)^{-1} B\Sigma \tag{12}$$

Although we were able to discover this analytic form for the updates to the distribution parameters in this case, there appears to be no simple analytic expression for the expected marginals of factor graphs drawn from the distribution. So, when such quantities as expected marginals are needed, we simply draw many sample graphs, compute their marginals, and average together the results. This averaging seems to be more sensible in the log domain since some marginals may be very close to 0 or 1. Averaging sampled marginals in the log domain is equivalent to taking the geometric average and renormalizing.

4.2 Optimal selection criteria

Below, we write $Z\tilde{P}(x)$ for a random variable representing a draw from the student's beliefs about $ZP(x)$, while $Z\hat{P}(x)$ represents an estimator of $ZP(x)$. We use a geometric average unless stated otherwise: $Z\hat{P}(x) \equiv \exp \mathbb{E}[\log Z\tilde{P}(x)]$.

We can derive an analytic expression for the change in μ when a new example x^* is seen. From equation 11 we have

$$\Delta\mu = \Sigma\Delta B^T (\Delta B\Sigma\Delta B^T)^{-1}\Delta \log Z\hat{P}(x^*) \tag{13}$$

where ΔB encodes the new constraint $z^* = ZP(x^*)$ and $\Delta \log Z\hat{P}(x^*) = B\mu - v$ is the difference between the old and new estimates of $ZP(x^*)$. Note that $\Delta B\Sigma\Delta B^T$ is just the variance of $\log Z\tilde{P}(x^*)$. If Σ

is initialized to a multiple of the identity matrix $\beta^2 I$, encoding a spherical normal distribution, then $\beta^{-2}\Sigma$ will be a projection and we will have $\Sigma\Sigma = \beta^2\Sigma$. This allows us to write a simple expression for the L_2 norm of $\Delta\mu$, which we obtain by multiplying equation 13 by its transpose and taking the square root:

$$||\Delta\mu||_2 = \beta \frac{\left|\Delta \log Z\hat{P}(x^*)\right|}{\sqrt{\text{Var}(\log Z\tilde{P}(x^*))}} \tag{14}$$

Since each update brings μ closer to the parameters of the true model, this equation tells us how to find updates which will maximize one measure of the speed of convergence of the student's representation: We should choose states x^* for which the error in his point estimate of $\log ZP(x^*)$, relative to the standard deviation of his beliefs, is greatest.

5 Experiments

In our experiments, we compare five different ways for the teacher to choose example states at which to update the student's distribution over potentials.

- **max L_2 delta μ** - (ML2DM) Chooses states which maximize $||\Delta\mu||_2$ according to equation 14.
- **max diff entry** - (MDE) At each turn, all states are examined and the state is chosen at which the entry in the student's unnormalized joint estimate is most different from the teacher's (exact) unnormalized joint, e.g. $\text{argmax}_x \left|ZP(x) - Z\hat{P}(x)\right|$. This is similar to ML2DM above, but using $\left|\Delta Z\hat{P}(x^*)\right|$ instead of $\left|\Delta \log Z\hat{P}(x^*)\right|$, and without weighting the states by the the inverse of the standard deviation of $\log Z\tilde{P}(x^*)$).
- **conditional game** - (CG) The student's marginals (their geometric average, calculated by sampling) are used to play as MP in a conditional game against the teacher's CP. Whether CP is trying to maximize or minimize the game outcome is decided uniformly at random before each game. The state chosen by the game is used as the next example. The teacher uses an exact distribution, except in section 6.1 which describes experiments with an approximate teacher.
- **max var log entry** - (MVLE) The state is chosen at which the student is maximally uncertain about the value of $\log ZP(x)$, as measured by the variance of this value over sampled models. I.e. $\max_x \text{Var}(\log Z\tilde{P}(x))$
- **uniform random** - (UR) A state is chosen uniformly at random.

The first three methods (ML2DM, MDE, and CG) compare estimates from the student's distribution with the true distribution. The the fourth (MVLE) only uses information from the student, and fifth method (UR) makes no reference to either.

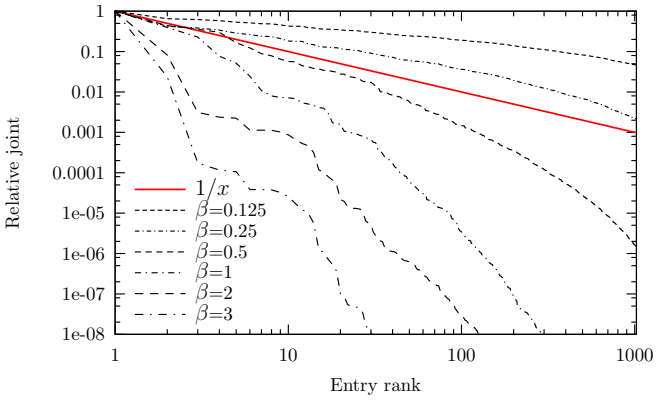


Figure 1: Zipf plots for entries in the joint distributions of six typical factor graphs, with reference line

We tested the five methods (ML2DM, MDE, CG, UR, MVLE) on four different models, all with 11 variables but differing in the variance β of the log-potentials. We explored values of 0.125, 1, and 3 for β. One useful way to visualize the difference in the models produced by these values of β is by making a Zipf plot of the entries in the joint distribution (this means that the entries for each state are sorted and plotted against their rank on a log-log scale; additionally we normalized the values so that the largest is 1). See Figure 1. The steeper the slope of the line, the more probability mass is placed on a few dominant states, rather than being spread out across many states. A line with slope -1 has been included for reference.[1]

For a fully-connected binary pairwise factor graph of n variables, there are $\frac{n(n+1)}{2}$ parameters in the potentials, thus we expect most methods to require 66 examples to learn our 11-variable example models completely.

The student's distribution over models uses as a prior the same normal parameters from which the true graph was generated. Using other parameters (including putting a normal prior on univariate factors ψ_i, and

[1] The slope -1 is a special case, which corresponds to a relationship $P(x_r) \propto \frac{1}{r}$ where r is the rank of the state x_r. Considering r to be continuous-valued, we note that $\int_0^1 \frac{1}{r^\alpha} dr$ is ∞ if $\alpha \geq 1$, and $\int_1^\infty \frac{1}{r^\alpha} dr$ is ∞ if $\alpha \leq 1$. Only when $\alpha = 1$ are both integrals infinite. This can therefore be compared to a case where probability mass is fairly divided between likely and unlikely states. From the plot, we see that the value $\beta = 0.5$ has a slope which is close to -1 over the first decade, but decreases thereafter.

on a scalar factor for the whole graph) did not produce substantially different behavior.

6 Results

The outcomes of running the five methods on each model are shown in Figure 2. We have plotted $L_1^{\log}$ error, but L_1 error gives similar results. The student's marginals are calculated by geometrically averaging the marginals of 256 models drawn from the student's distribution, and renormalizing.

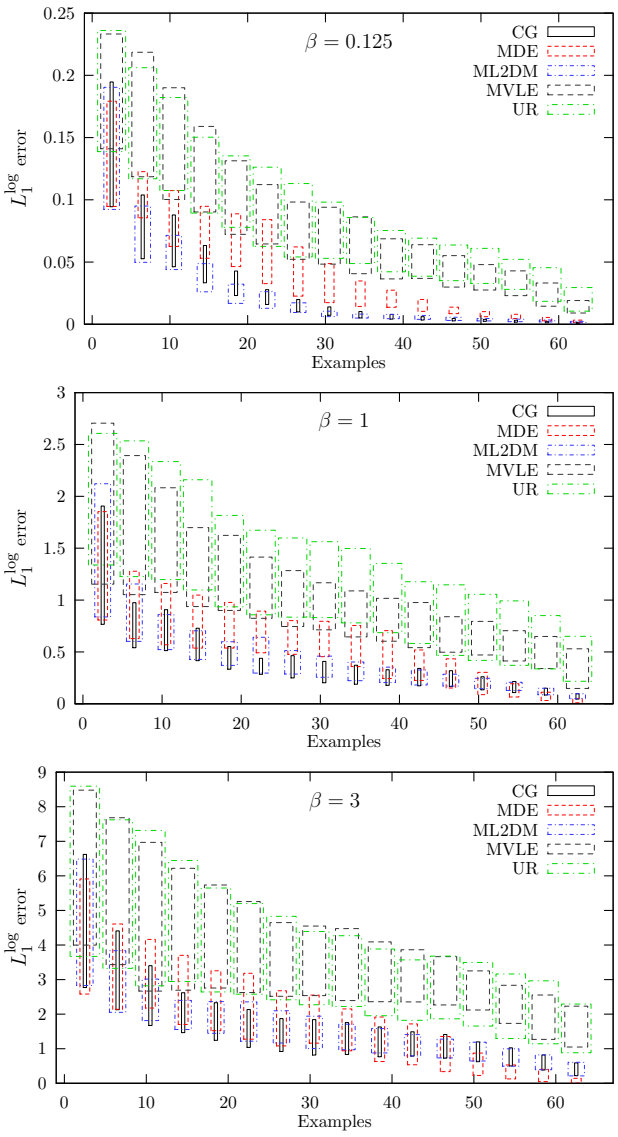


Figure 2: Student's error as a function of example count, for three values of β. Sample counts are grouped into bins of size four. For each β, 10 random models were generated. For each interaction method and bin, a box with ± 1 standard deviation in log error is plotted, showing the variation in error over these models.

The plots demonstrate a number of consistent relationships between the five methods. MVLE is close to UR, although usually better, and both perform relatively poorly. CG and ML2DM seem to have the best overall performance. Although ML2DM sometimes does better than CG, they are generally indistinguishable. For small β, CG and ML2DM are clear winners. For $\beta = 1$ or 3, they sometimes trail MDE, but without ever being far behind. On the other hand, there are conditions under which MDE performs significantly worse.

It is perhaps surprising that CG does so well, since most of the other methods (all except UR) require examining every entry of the unnormalized joint ZP and would be prohibitively expensive to implement on large graphs, except in some heuristic form. CG only requires conditioned marginals, which may be easily obtained from most approximations.

It is also interesting that CG overlaps so closely with ML2DM. The CG can be seen as a greedy procedure for finding the biggest $\Delta \log Z\hat{P}(x^*)$, while ML2DM directly maximizes a scaled version of the same quantity. The CG only looks at marginals, while ML2DM examines every state; we might have predicted that this distinction would give the two methods very different behavior, but apparently it doesn't.

A number of other methods for choosing examples were explored but were found to perform poorly, and are not shown in the plots. The application of the conditional game we used chooses randomly to have CP maximize or minimize the value. If instead we eliminate the random choice and select each point with CP only maximizing or only minimizing, then the error does not decrease (in some instances the same state is chosen repeatedly). Choosing states of the factor graph in decreasing order of the true joint distribution at each state, so that only the states of highest probability mass are shown to the student, performs very poorly. Presumably this is because the examples generated, while distinct, yield degenerate constraints and so are uninformative. Choosing an example at random from the exact distribution performs about as well as UR, as does choosing a random state from a model drawn from the student. Choosing a state by selecting the variable and value with the largest $\mathrm{Var}(\sum_{x_{\setminus i}} Z\tilde{P}(x))$, conditioning the variable to that value and recursing, works about as well as MVLE. For the CG, having the student's MP use marginals from a sampled $Z\tilde{P}$ rather than an averaged estimator $Z\hat{P}$ gave worse performance, as did using arithmetic rather than geometric averaging of the marginals.

6.1 Approximate teacher

It is interesting to ask how the CG performs when the teacher's beliefs differ from the true distribution. In Figure 3, we plotted the "learning curve" with the CG method for three different choices of teacher: with exact beliefs (as before), and with beliefs calculated from the Belief Propagation (BP) or Mean Field (MF) approximations. The $L_1^{\log}$ error of BP and MF are shown by horizontal lines. For the easiest model, $\beta = 0.125$, the teacher's inaccuracy seems to have no adverse effect. For $\beta = 3$, the most difficult model, with MF it is easy to see that the student's performance is being held back by the teacher, although it is possible for the student to improve on the teacher to a limited extent. For BP the situation is apparently similar but the effect is less pronounced because of BP's higher accuracy. Since it is possible to achieve perfect accuracy for the student simply by choosing 66 random states (which are almost certain to be linearly independent), we would guess that the reason for the student's accuracy to plateau in this experiment is that the teacher is showing the same state repeatedly. This is confirmed by the plot of Figure 4, which shows the number of unique states as a function of round.

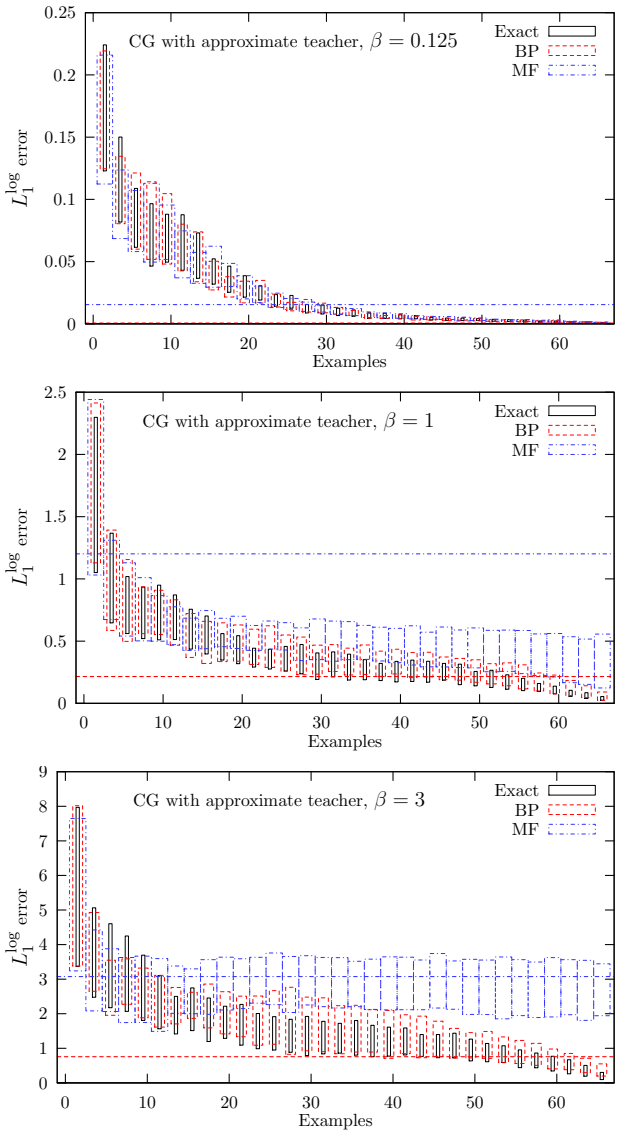


Figure 3: Learning curves for approximate teachers, for three values of β. As in Figure 2, but with bins of size 2.

7 Discussion and future work

In summary of the above results, we would say that the CG method seems to be the best approach for choosing data points for guided inference. This might be surprising, considering the independent purpose for which it was designed. We were unable to find a method which does uniformly better, even including generally intractable methods which required examining every state in the model. Although there may be other superior methods which we neglected to consider, the approaches we tried were appropriate and well-motivated. One might be tempted to conclude that the most efficient mode of interaction between a teacher and student, at least in the protocol we have defined, is that of having a debate.

The present investigations were to some extent motivated by the hope of eventually designing an approximate inference algorithm based on a simulated natural selection. Such an algorithm would use cooperation and competition between candidate approximations to try to "evolve" approximations of higher accuracy. When one imagines a context for interactions in which approximations are made to compete using games, and also to cooperate by sharing information, the question arises of the proper relationship between these two modes of interaction: Is it (a) possible for an approximation to do well in competitions as a result of having hidden knowledge which he never has to share with his colleagues? Or (b) does the expertise which allows one approximation to outperform another get revealed as soon as we arrange a competition between them? In case (b), our task of deriving a useful interaction framework might be simplified by the sufficiency of a single form of interaction to accomplish the goals of both (competitively) *comparing* and (cooperatively) *educating* approximate inference algorithms. In case (b) there would also be an intrinsic motive for each party to participate in such interactions - the winner receives the prestige of winning, while the loser receives useful training. It is possible for an evolutionary framework to provide these motives artificially, by some deliberate innovation of the design, but we would prefer them to arise as a natural consequence of the interaction itself. We were pleased to confirm

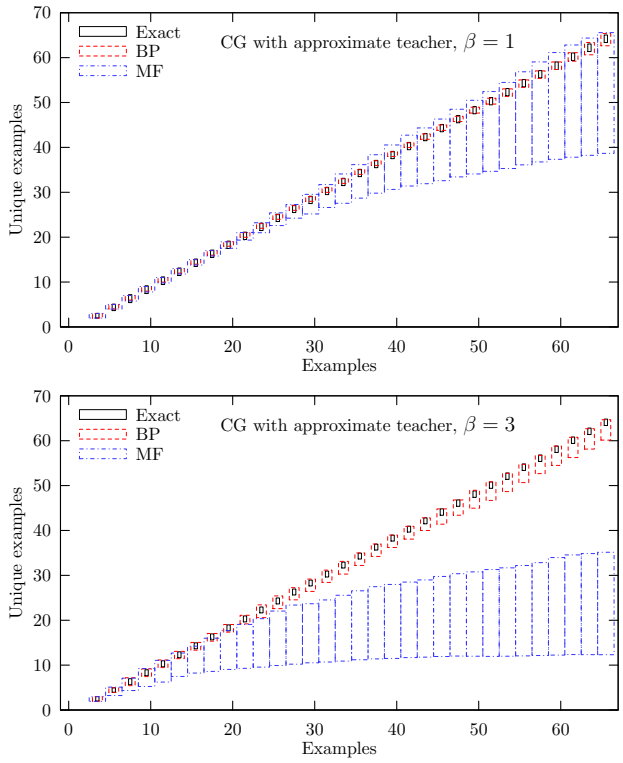


Figure 4: Unique examples as a function of examples, for approximate teachers and $\beta = 1$ and 3. Not shown is $\beta = 0.125$, where all examples were unique.

through the experiments presented here that, at least to good approximation, the second case indeed holds - the same competitive interaction (i.e., the CG) efficiently accomplishes both comparison and education. But it remains to be seen how one should best utilize the consequences of this principle in an algorithm.

There are some discernible directions in which this research could be extended. Apparently, for our ideas to become practically useful, at some point it will be necessary to replace the intractable exact inference methods used in these experiments with a suitable approximate inference algorithm, one which can be parametrized by states of the model so that it can "learn" from a teacher. As with Rasmussen and Ghahramani (2003), our problem may also benefit from a kernel-based method. It is interesting to ask what the appropriate kernel should be.

Additionally, we would like to be able to modify the protocol to work on subsets of the variables, so that multiple guided-inference-style interactions could be conducted in parallel on separate regions of a large graph. A desire for a similar modification - to adapt the the ideas of the CG so that only a subset of variables must be traversed in a single game - was voiced in Eaton (2011). Apparently neither design goal is possible without losing much of the original simplicity. The CG is based on the fact that we may easily evaluate the unnormalized joint ZP. To adapt it to use only a partial assignment of the variables, x_C for $C \subseteq \{1 \dots n\}$, we would need a way to estimate the equivalent "marginal" quantity, say $ZP_C(x_C)$. This could easily be provided by a third approximation (for instance using the $\log Z$ estimate of a conditioned BP or MF), but would then make the game "relative" to that approximation. Similarly, a "guided inference" protocol in such a partial framework, if it uses a third approximation to provide the (partial) ZP values which had in our treatment been directly computable, would necessarily lose some desirable properties: the student's accuracy would be limited by the accuracy of this third party, even if not by the accuracy of the teacher. Thus additional complexities arise on the way to a practical application of both the CG and guided-inference ideas. However, a comparison to the real world suggests that introducing a third approximation is the right approach: in a trial, it would be analogous to the role of a judge, the traditional means by which the boundaries of relevance are established. Without the shared reservoir of "common sense" provided by a judge, a fair trial, to which the CG can be seen as vaguely analogous, would be impractical. In the guided-inference framework, because there is no "winner" or "loser", the need for such a third party is not so clear. To make do with "partial" states, one still needs to construct a "marginalized" version of ZP, which must be approximate. But these estimates could conceivably be provided by the student or the teacher himself, although in both cases we still sacrifice some desirable properties of the protocol. Understanding how to fit together all of the above ideas to produce a practical approximate inference algorithm is likely to be a challenging project. But the findings of this paper might encourage us to look for a solution in which multiple design goals - the ability to simulate both cooperation and competition - are satisfied by the same design elements.

Acknowledgments

(omitted from reviewer copy)

References

Eaton, F. (2011). A conditional game for comparing approximations. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15.

Rasmussen, C. and Ghahramani, Z. (2003). Bayesian monte carlo. *Advances in neural information processing systems*, pages 505–512.

Tong, S. (2001). *Active learning: theory and applications*. PhD thesis, Stanford University.