Herding Dynamical Weights to Learn

Max Welling WELLING@ICS.UCI.EDU
Donald Bren School of Information and Computer Science, University of California Irvine, CA 92697-3435, USA

Abstract

A new "herding" algorithm is proposed which directly converts observed moments into a sequence of pseudo-samples. The pseudo-samples respect the moment constraints and may be used to estimate (unobserved) quantities of interest. The procedure allows us to sidestep the usual approach of first learning a joint model (which is intractable) and then sampling from that model (which can easily get stuck in a local mode). Moreover, the algorithm is fully deterministic, avoiding random number generation) and does not need expensive operations such as exponentiation.

1. Introduction

Imagine we wish to invest our money in stocks. To optimize our portfolio we are interested in the probability distribution $P(k, N)$ of k companies defaulting out of N companies in our portfolio (within a certain time horizon). However, we are only given the pairwise probabilities of two companies defaulting together, $P_{ij}(x_i, x_j)$. How can we estimate $P(k, N)$ efficiently?

A popular approach is to formulate this as a "maximum entropy" problem (Jaynes, 1957), where the pairwise marginal distributions P_{ij} serve as the observed moments while the remaining degrees of freedom are determined by maximizing the entropy of the joint distribution $P(\mathbf{x})$. The dual of this problem is given by the maximum likelihood problem of a Markov random field model with indicator features $f_{ij,kl}(x_i, x_j) = \mathbb{I}[x_i = k, x_j = l]$ (Lebanon & Lafferty, 2002). The weights $w_{ij,kl}$ multiplying these features serve as Lagrange multipliers in the primal problem. Unfortunately, computing the optimal values for the weights is very hard because it requires averages of the features under the model. There have been many attempts to approximate this learning problem which have proven useful in a wide range of real world applications (Besag, 1977; Geman & Geman, 1984; Zhu et al., 1997; Lafferty, 1999; Zhu & Liu, 2002; Hinton, 2002; Hyvarinen, 2005; Tieleman, 2008).

Once, the weights have been learned we have access to the full joint (Gibbs) distribution P and we are ready to estimate quantities of interest. This may be achieved by first exhaustively sampling from the distribution and then computing the quantity of interest by averaging over these samples. However, at this stage another problem may surface. The painstakingly estimated Gibbs distribution may have many local modes in which our Markov chain Monte Carlo procedure may get stuck.

The question we ask ourselves in this paper is if there is perhaps a shortcut that sidesteps the learning phase and directly generates samples with only the moments as input? While we will insist that our samples will reproduce the observed moments we may relax the requirement of maximum entropy (which acts as our inductive bias for the purpose of generalization). There is some indication in the literature that the two phase *learning→inference* approach may be suboptimal. For instance, the "unified propagation and scaling" algorithm of (Teh & Welling, 2002) combines learning and inference into one problem that avoids the two separate phases. This algorithm was further improved recently by (Ganapathi et al., 2008). Unfortunately, even these approximate methods cannot estimate a global quantity such as $P(k, N)$ because it depends on all variables in the problem jointly.

Learning MRF models can also be formulated as a problem in stochastic approximation theory (Younes, 1999; Yuille, 2004). It was first observed in (Neal, 1992) and later elaborated in (Tieleman, 2008) that one can view learning in MRF models as running a Markov chain that is periodically interrupted by a small parameter update. It was then noted that as long as this perturbation is small and the chain has sufficient time to equilibrate between perturbations, and

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009.

in addition to this the stepsize is decreased according to a certain schedule, one is guaranteed to eventually converge on the maximum likelihood value of the parameters.

We believe that this procedure is still unnecessarily inefficient if one is only interested in estimating quantities such as $P(k, N)$. More precisely, one can refrain from annealing the stepsize and let the parameters "dance around" instead. It is then not hard to see that averages over the samples from the periodically interrupted Markov chain will reproduce the moment constraints. Hence, we may be wasting resources if we are trying to nail down precise estimates for the parameters by annealing step-sizes. Parameters are now more like auxiliary variables necessary to define a sampling procedure. The system acts like a filter that transforms data (observed moments) directly into samples without first estimating parameter values. It is in this sense that one can think of it as a nonparametric approach to estimation.

As was also observed by (Tieleman, 2008), the sequence of samples from this periodically interrupted Markov chain tend to rapidly mix between different modes[1]. The reason is that the perturbations of the weights from the learning updates push the sampler away from regions that are over-explored. Therefore, also in this sense it is much more efficient to simply collect samples while running the "learning" updates with a fixed stepsize than to run a separate Markov chain *after* a point estimate has been found. The only downside of this shortcut is that we are no longer guaranteed to obtain samples from a maximum entropy distribution. Thus, although the moment constraints are still satisfied we implicitly put a different prior on the remaining degrees of freedom. Empirically, we have observed that the impact of this is small, and may in some cases even improve the estimates of interest.

As a final twist, we have discovered that there is in fact no need to run a periodically perturbed Markov chain at all. By taking the zero temperature limit of the corresponding maximum likelihood problem, we can replace sampling by maximization, resulting in a fully deterministic dynamical system. Perhaps surprisingly, pseudo-samples collected from the trajectories of this dynamical system are also guaranteed to satisfy the moment constraints. In addition they exhibit a high degree of "pseudo randomness" (and hence entropy) which is caused by the complexity of the underlying nonlinear dynamics (for an example see Figure 1). Converting to this deterministic system of equations has additional computational advantages in that we avoid exponentiation and random number generation. As such, the proposed system may be more suitable for hardware implementation.

The analysis of the proposed herding dynamics requires tools from nonlinear dynamical systems theory. For instance, an arbitrarily initialized set of parameters seems to converge consistently to an attractor set with a dimensionality that was numerically estimated to be fractal [2]. We can show that the parameters will never run away to infinity, but the issue whether there exists a *unique* invariant measure is still open. It is not even clear if the dynamical system is chaotic or not. All Lyapunov exponents are 0 but one can show that information will be lost at a very slow (polynomial) rate. On the other hand, by slowly decreasing the temperature to zero, we have observed bifurcation sequences which are indicative of chaos. We believe that the system operates "on the edge of chaos", but this needs to be further investigated.

Perhaps the most exciting perspective that this work has to offer is that by linking the theory of learning to that of complex nonlinear dynamical systems a whole new field may open up for further exploration.

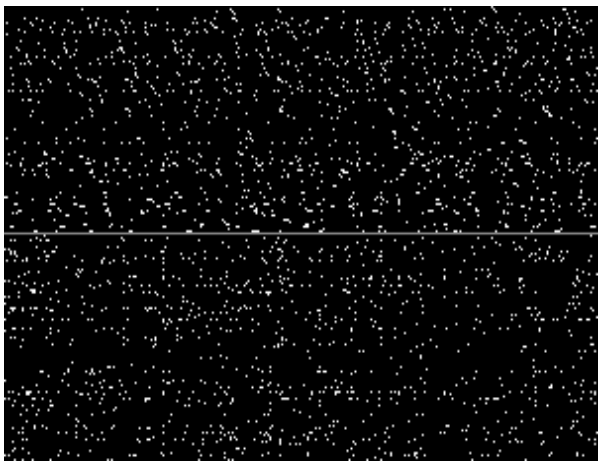

Figure 1. Top half: Sequence of 300 pseudo-samples generated from a herding algorithm for the "Newsgroup" dataset (section 6). White dots indicate the presence of certain word-types in documents (represented as columns). Bottom half: Newsgroup data (in random order). Data and pseudo-samples have the same first and second order statistics.

[1] Loosely speaking, these modes would be similar to the modes of the distribution obtained after the parameters have converged.

[2] Note that in this sense the algorithm is not dependent on its initial conditions.

2. Maximum Entropy and Maximum Likelihood

Define $\{x_i\}$ to be a set of D random variables over a discrete alphabet, $x_i \in \{1..K_i\}$. Groups of variables will be indexed using Greek indices, e.g. x_α. We define features over groups of variables by $g_\alpha(x_\alpha)$ while the expected value of g_α over data is denoted by $\bar{g}_\alpha$.

In a maximum entropy problem one is seeking a joint distribution P that satisfies a number of pre-specified expectation constraints (a.k.a. moments) while the remaining degrees of freedom are determined by requiring the distribution to have maximal entropy. We will include a temperature in the maximum entropy problem,

$$P = \arg\max_{\mathcal{P}} \quad T\mathcal{H}(\mathcal{P}) \quad \text{s.t.} \quad \mathbb{E}[g_\alpha]_{\mathcal{P}} = \bar{g}_\alpha, \ \forall\alpha \tag{1}$$

The dual of this formulation is in fact the well known maximum likelihood problem for Markov random fields without hidden variables (Lebanon & Lafferty, 2002). Here, one maximizes the following objective over $\mathbf{w}$ which are now interpreted as the Lagrange multipliers of the primal problem,

$$\ell_T(\mathbf{w}) = w_\alpha \bar{g}_\alpha - T \log \sum_{\mathbf{x}} \exp(\frac{1}{T} \sum_\alpha w_\alpha g_\alpha(x_\alpha)) \tag{2}$$

To learn the optimal values for the weights, we can take the gradients of the log-likelihood and apply gradient descent updates,

$$w_{\alpha,t+1} = w_{\alpha t} + \eta(\bar{g}_\alpha - \mathbb{E}[g_\alpha]_P) \tag{3}$$

This update is unfortunately intractable due to the fact that we can generally not compute the quantity $\mathbb{E}[g_\alpha]_P$ efficiently. Assuming for a moment that we have the optimal weight values, then the following Gibbs distribution is guaranteed to have the correct moments and maximum entropy on the remaining degrees of freedom,

$$P(\mathbf{x}) = \frac{1}{Z(\mathbf{w})} \exp\left[\sum_\alpha w_\alpha g_\alpha(x_\alpha)\right] \tag{4}$$

Quantities of interest can now be formulated as averages over this distribution. To compute those, we can draw samples from this distribution using Markov chain Monte Carlo techniques.

A fundamental question is whether we actually need to compute the gradient in Eqn.3 with high precision? Assume we only have very noisy estimates of the gradient, how could we use these to estimate quantities of interest? One answer is to run a Markov chain continuously which we periodically interrupt to update the weights using Eqn.3 (Younes, 1999; Neal, 1992; Tieleman, 2008). The samples are used to approximate the term $\mathbb{E}[g_\alpha]_P$ in Eqn.3. Importantly, in order to converge to point estimates for the parameters, the stepsize η needs to be decreased using some suitable annealing schedule. But let's assume for a moment we don't do that. We can then still collect the samples produced by this "periodically interrupted Markov chain". Interestingly, averaging the features over these samples will satisfy the moment constraints, as the following theorem proves.

Proposition 1: If $\forall\alpha \ \lim_{\tau\to\infty} \frac{1}{\tau} w_{\alpha\tau} = 0$, then $\forall\alpha \ \lim_{\tau\to\infty} \frac{1}{\tau} \sum_{t=1}^{\tau} g_\alpha(s_{\alpha t}) \to \bar{g}_\alpha$.

Proof:

$$\delta w_{\alpha t}/\eta = \bar{g}_\alpha - g_\alpha(s_{\alpha t}) \tag{5}$$

with $\delta w_{\alpha t} \triangleq w_{\alpha t} - w_{\alpha,t-1}$. Next, average left and right hand sides over t,

$$\frac{1}{\tau}\sum_{t=1}^{\tau} \delta w_{\alpha t}/\eta = \frac{1}{\tau}(w_{\alpha\tau} - w_{\alpha 0})/\eta = \bar{g}_\alpha - \frac{1}{\tau}\sum_{t=1}^{\tau} g_\alpha(s_{\alpha t}) \tag{6}$$

Using the premise that the weights grow slower than linearly we see that the left hand term vanishes in the limit $\tau \to \infty$ which proves the result.

What this says is that under the very general assumption that the weights don't grow out to infinity linearly (not that due to the finite stepsize they can now grow faster than linear), the moment constraints will be satisfied by the samples collected from the combined learning/sampling procedure. For many applications, these samples are all we need, and so it is a waste of resources to try to nail down a single point estimate for the weights by decreasing the stepsize. But we actually killed two birds with one stone. Sampling from P after convergence can easily get stuck in local modes. As observed before by (Tieleman, 2008), the weight updates actually help with mixing because they bias the chain away from over-explored regions of state space.

What is not clear is whether we also produce samples with high entropy which (presumably) generalize well. We conjecture this is the case, albeit that the entropy may not be maximal.

3. Tipi Functions and The Zero Temperature Limit

We now ask the question whether it was necessary to run the Markov chain in the first place? To answer that question we will take the zero temperature limit

of 2,

$$\ell_0(\mathbf{w}) = \sum_\alpha w_\alpha \bar{g}_\alpha - \max_{\mathbf{s}} \left[\sum_\alpha w_\alpha g_\alpha(s_\alpha) \right] \quad (7)$$

This function has a number of interesting properties that justify the name "Tipi function"(see Figure 2) which we discuss below[3]

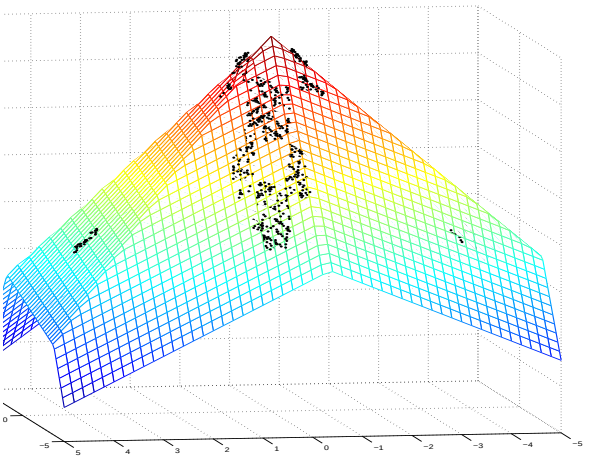

Figure 2. Two-dimensional Tipi-function for herding with features $f(x) = sin(x)$ and $g(x) = cos(x)$, $x = [-\pi, -\pi + 1, .., 2.86]$ and a uniform distribution $P(x)$ to compute $\bar{f}$ and $\bar{g}$. Small dots represent weights sampled during herding.

1. ℓ_0 *is continuous piecewise linear (C^0 but not C^1).* It is clearly linear in $\mathbf{w}$ as long as the maximizing state $\mathbf{s}^*$ doesn't change. However, changing $\mathbf{w}$ may in fact change the maximizing state in which case the gradient changes discontinuously.

2. ℓ_0 *is a concave, non-positive function of* $\mathbf{w}$ *with a maximum at* $\ell_0(0) = 0$. This is true because the first term represents the average $\mathbb{E}[\sum_\alpha w_\alpha g_\alpha]_P$ over some distribution P, while the second term is its maximum. Therefore, $\ell_0 \leqq 0$. If we furthermore assume that $P > 0$ everywhere then $\ell_0 < 0$ and the maximum at $\mathbf{w} = 0$ is unique. Concavity follows because the first term is linear and the second maximization term is convex.

3. ℓ_0 *is scale free.* This follows because $\ell_0(\beta\mathbf{w}) = \beta\ell_0(\mathbf{w})$ as can be easily checked. This means that the function has exactly the same structure at any scale of $\mathbf{w}$.

[3]These properties warrant the name "Tipi function" because in two dimensions it looks like a, possible crooked, native Indian Tipi dwelling.

We notice that in this limit optimization becomes a futile exercise: the maximum is at $\mathbf{w} = 0$ and there is no need to search for it. However, the procedure proposed in the previous section still makes sense. We can perform gradient ascent on this surface with a fixed (non decreasing) stepsize. The fixed stepsize will result in a perpetual overshooting of the maximum at $\mathbf{w} = 0$. Every flat face of the Tipi function is associated with a state and the state sequence obtained by following this gradient ascent procedure still has the property that its averages over features will reproduce the observed average feature values (the proof is identical to the one presented in the previous section).

Moreover, we notice that changing the stepsize will only change the scale at which the weights operate. However, because ℓ_0 is scale free, it has no effect on the actual state sequence. Hence, we can simply set $\eta = 1$.

Taken together, we now formulate our herding dynamics,

$$\mathbf{s}_t^* = \arg\max_{\mathbf{s}} \sum_\alpha w_{\alpha t} g_\alpha(\mathbf{s}_\alpha) \quad (8)$$

$$w_{\alpha,t+1} = w_{\alpha,t} + \bar{g}_\alpha - g_\alpha(s_{\alpha t}^*) \quad (9)$$

$$f_\beta^*(y_\beta) \leftarrow \left(\frac{t-1}{t}\right) f_\beta^*(y_\beta) + \left(\frac{1}{t}\right) f_\beta(s_{\alpha t}^*)\mathbb{I}[y_\alpha = s_{\alpha t}^*] \quad (10)$$

Note that this represents a collection of *deterministic* nonlinear update equations. In particular it is not a Markov chain Monte Carlo procedure and it does not require random number generation. In fact, it doesn't even require exponentiation which is computationally expensive relative to maximization.

The first update determines the state $\mathbf{s}$ necessary to compute the gradient of the Tipi function, i.e. it determines the locally flat region of the function on which we are currently located. Given this, the second update takes a gradient step upwards on 7. This process is repeated and while we sample states $\mathbf{s}_t$ we compute online averages for quantities of interest using the third update.

The herding updates represent two phases of a minimax problem with $\mathbf{s}$ minimizing the objective ℓ_0 and $\mathbf{w}$ maximizing it. The maximization over $\mathbf{s}$ can still be a very hard problem. In practice we perform a local coordinate ascent version of it and our empirical results show that this still works well. In fact, proposition 1 tells us that we simply have to monitor the L_2-norms of the weights and make sure that they do not grow to infinity linearly. If they don't, proposition 1 guarantees that the moments will be correct.

Conversely, if the moments will not be reproduced, for example because we specified inconsistent moments or because local optimization is stuck in some corner of state space, the weights will grow away to infinity linearly. Hence, the weight norms act as detectors for the algorithm going astray. One can imagine an adaptive version of the algorithm that spends more time on maximization over $\mathbf{s}$ if the weight norms become too large.

4. Recurrence

The premise to proposition 1 is that the norm of the weights do not grow to infinity. We will now prove that for the herding algorithm defined in the previous section this does not happen. The proof depends on our ability to identify the true maximizing state $\mathbf{s}^*$ which may not be satisfied in many applications. However, as we will discuss at the end of this section, the result may still be useful when we replace full maximization with local maximization.

In the following we will assume that $\bar{g}_\alpha = \mathbb{E}[g_\alpha]_P$ with $P(\mathbf{x}) > 0,\ \forall \mathbf{x}$ implying $\ell_0(\mathbf{w}) < 0$ everywhere except at the origin. We will also need the result that the gradient in the direction of w_α is always negative: $\sum_\alpha w_\alpha \nabla_{w_\alpha} \ell_0(\mathbf{w}) < 0$ everywhere except on the boundaries between the piecewise linear faces where the derivative is not defined. This result follows from concavity of ℓ_0, but can also be understood by observing that $\sum_\alpha w_\alpha \nabla_{w_\alpha} \ell_0(\mathbf{w}) = \ell_0(\mathbf{w})$.

Lemma 1: If $|g_\alpha(s_\alpha)| < \infty,\ \forall \mathbf{s}, \alpha$, then $\exists\ \mathcal{B}$ such that $||\nabla \ell_0||_2 < \mathcal{B}$.

Proof: $\nabla_{w_\alpha} \ell_0(\mathbf{w}) = \bar{g}_\alpha - g_\alpha(s^*_\alpha)$ with $\mathbf{s}^*$ the maximizing state. Since all $g_\alpha(s_\alpha)$ are finite for any value of $\mathbf{s}_\alpha$ the norm of the gradient must be bounded as well.

We now prove that there will be some radius $\mathcal{R}$ such that the herding algorithm will always decrease the norm $||\mathbf{w}||_2$.

Proposition 2: $\exists\ \mathcal{R}$ such that a herding update performed outside this radius, will always decrease the norm $||\mathbf{w}||_2$.

Proof: Write the herding update as $w'_\alpha = w_\alpha + \nabla_{w_\alpha} \ell_0$. Take the inner product with w'_α leading to, $||\mathbf{w}'||_2^2 = ||\mathbf{w}||_2^2 + 2\sum_\alpha w_\alpha \nabla_{w_\alpha} \ell_0 + ||\nabla_{w_\alpha} \ell_0||_2^2$, which leads to $\delta ||\mathbf{w}||_2^2 < 2\ell_0 + \mathcal{B}^2$. We now use the fact that 1) $\ell_0 < 0$ outside the origin, 2) $\mathcal{B}$ is constant (i.e. doesn't scale with $\mathbf{w}$) and 3) the scaling property $\ell_0(\beta \mathbf{w}) = \beta \ell_0(\mathbf{w})$ to argue that there is always some radius $\mathcal{R}$ for which $\delta ||\mathbf{w}||_2 < 0,\ \forall ||\mathbf{w}||_2 > \mathcal{R}$ (if not, increase β by a sufficient amount).

Corollary: $\exists\ \mathcal{R}'$ such that a herding algorithm initialized inside a ball with that radius will never generate weights $\mathbf{w}$ with norm $||\mathbf{w}||_2 > \mathcal{R}'$.

This follows because in the worst case we could still take one step radially outward starting somewhere on the surface of the ball at radius $\mathcal{R}$. Since the gradient is bounded in magnitude by $\mathcal{B}$ we have that $\mathcal{R}' \leqq \mathcal{R} + \mathcal{B}$.

These results may help to improve the versions of herding based on local optimization. For instance, we could derive a conservative radius $\mathcal{R}'' > \mathcal{R}'$ beyond which one does not *allow* the norm $||\mathbf{w}||_2$ to grow further. One can adapt the amount of effort spent at the maximization step of the herding algorithm (Eqn.8) to achieve this. The above results help identify the radius beyond which one can guarantee that there exists a state for which this is possible. Also, note that the result of proposition 1 still holds for such an adaptive herding variant.

5. Herding in Hopfield Networks

Hopfield networks (Hopfield, 1982) are recurrent neural networks defined by an energy function of the form, $\mathcal{E}(\mathbf{s}) = -(\sum_{ij} w_{ij} s_i s_j + \sum_i \theta_i s_i)$. Denote with $s_{it} \in \{0, 1\}$ the state of neuron i at time t. We will interpret $s = 1$ as a firing event. Groups of neurons will be labeled with α and their joint state denoted as $s_{\alpha t}$. The synaptic strength between neurons i and j at time t is denoted with $w_{ijt} \in \mathbb{R}$, while the bias for each neuron i is denoted with $\theta_{it} \in \mathbb{R}$.

We assume to have observed target frequencies p_i, p_{ij} for neuron i to be in state 1 and neurons i, j to be in state $(1, 1)$ jointly. Note that these two quantities are sufficient to determine the complete 2×2 table of joint probabilities for each pair of neurons. We usually (but not always) compute target frequencies from outside data sources which we denote with x_{in}. One can interpret this as the n'th measurement for neuron i. From this we then compute, $p_i = \frac{1}{N} \sum_{n=1}^N \mathbb{I}[x_{in} = 1]$ and $p_{ij} = \frac{1}{N} \sum_{n=1}^N \mathbb{I}[x_{in} = 1, x_{jn} = 1]$ where $\mathbb{I}[\cdot]$ is the indicator function.

In terms of these quantities we can now state the herding algorithm for the Hopfield network. We first initialize all weights $\mathbf{w}$ and states $\mathbf{s}$ arbitrarily. We then iterate the following equations:

$$s_{it} = \mathbb{I}[\theta_{it} + \sum_{j \in \mathcal{N}(i)} w_{ijt} s_{j,t-1} > 0] \quad \text{(until conv.)} \tag{11}$$

$$w_{ij,t+1} = w_{ijt} + p_{ij} - \mathbb{I}[s_{it} = 1, s_{jt} = 1] \tag{12}$$

$$\theta_{i,t+1} = \theta_{i,t} + p_i - \mathbb{I}[s_{it} = 1] \tag{13}$$

where $\mathcal{N}(i)$ denotes the set of all neighbors of i.

Update 11, $s_{i,t+1} = \mathbb{I}[\theta'_{it} + \sum_{j \in \mathcal{N}(i)} w'_{ijt} s_{jt} > 0]$, should be interpreted as a firing event if the argument evaluates to "true". Computationally, it maximizes the function $\sum_{(ij)} w_{ij} s_i s_j + \sum_i \theta_i s_i$ through local coordinate ascent.

The update equations for the weights (and biases) can be interpreted as follows. At every iteration the weights "recover" by an amount p_{ij}. But by the time the weight has grown large it becomes increasingly likely that the neurons on both sides of the weight fire, after which the weight "depresses" by an amount equal to 1[4]. There is an intriguing similarity between herding and dynamical weights as described in e.g. (Maass & Zador, 1998; Pantic et al., 2002; Levina et al., 2007). There, synaptic efficacy is depressed after a firing event in the presynaptic neuron. It is argued that the fast depression/recovery dynamics of the synapses drives the system to self criticality which in turn is useful for information processing.

6. Experiments

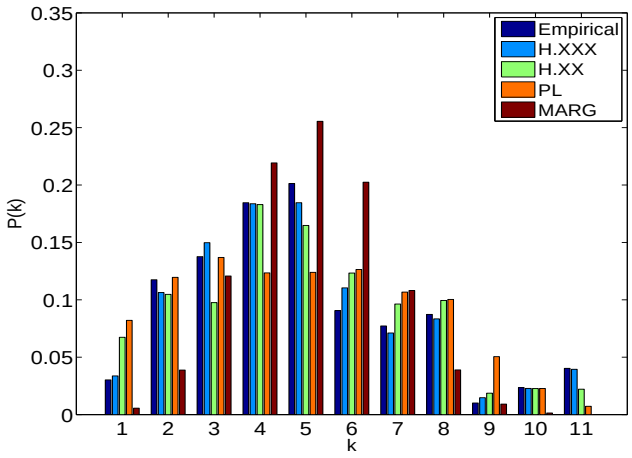

Figure 3. Estimates of $P(k)$ for the Bowling dataset. Each group of 5 bars represent the estimates for 1) ground truth, 2) herding with triples, 3) herding with pairs, 4) pseudo-likelihood, 5) marginals.

In the following experiments we will determine the ability of herding to convert information about average sufficient statistics into estimates of some quantities of interest. In particular the input to herding will be joint probabilities of pairs of variables (denoted H.XX) and sometimes triples of variables (denoted H.XXX) where all variables will be binary valued (which is easily relaxed).

[4]This is somewhat similar to an piston engine where the piston is pushed down after the fuel has combusted.

In experiment I we will consider the quantity $P(k) = \mathbb{E}[\mathbb{I}[\sum_i X_i = k - 1]]$ which is the distribution of the total number of 1's across all attributes. This quantity involves all variables in the problem and cannot be directly estimated from the input which consists of pairwise information only. This experiment measures the ability of herding to generalize from local information to global quantities of interest. In total 100K pseudo-samples were generated and used to estimate $P(k)$. The results were compared with the following two alternatives: 1) sampling 100K samples from the single variable marginals and using them to estimate $P(k)$ (denoted "MARG"), 2) learning a fully connected, fully visible Boltzman machine using the pseudo-likelihood method[5] (denoted PL), then sampling 200K samples from that model and using the last 100K to estimate $P(k)$.

In experiment II we will estimate a discriminant function for classifying one attribute (the label) given the values of other attributes. Our approach was simply to perform online learning of a logistic regression function after each pseudo-sample collected from herding. Again, local pairwise information is turned into a global discriminant function which is then compared with some standard classifiers learned directly from the data. In particular, we compared against Naive Bayes, 5-nearest neighbors, logistic regression and a fully observed, fully connected Boltzman machine learned with pseudo likelihood on the joint space of attributes and labels. The learned model's conditional distribution of label given the remaining attributes was subsequently used for prediction.

We have used the following datasets in our experiments.

A) The "Bowling Data" set[6]. Each binary attribute represents whether a pin has fallen during two subsequent bowls. There are 10 pins and 298 games in total. This data was generated by P. Cotton to make a point about the modeling of company default dependency. Random splits of 150 train and 148 test instances were used for the classification experiments.

B) Abalone dataset [7]. We converted the dataset into binary values by subtracting the mean from all (8) attributes and labels and setting all obtained values to 0 if smaller than 0 and 1 otherwise. For the classification task we used random subsets of 2000 examples for training and the remaining 2177 for testing.

[5]This method is close to optimal for this type of problem (Parise & Welling, 2005).

[6]Downloadable from:
http://www.financialmathematics.com/wiki/Code:tenpin/data

[7]Downloadable from UCI repository.

Table 1. **Abelone/Digits/NewsGroups:** KL divergence between true (data) distribution and the estimates from 1) herding algorithm using all triplets, 2) herding with all pairs, 3) samples from pseudo-likelihood model and 4) samples from single marginals.

DATASET	H.XXX	H.XX	PL	MARG
BOWLING	5E-3	4.1E-2	1.2E-1	4.3E-1
ABELONE	8E-4	2.5E-3	2.2E-2	1.8E0
DIGITS	–	6.2E-2	3.3E-2	4E-1
NEWS	–	2.5E-2	1.9E-2	5E-1

Table 2. Average classification results averaged over 5 runs.

DATA	H.XXY	PL	5NN	NB	LR
ABEL.	0.24± 0.004	0.24± 0.004	0.33± 0.1	0.27± 0.006	0.24± 0.004
BOWL.	0.23± 0.03	0.28± 0.06	0.32± 0.05	0.23± 0.03	0.23± 0.03
DIGIT	0.05± 0.01	0.06± 0.01	0.05± 0.01	0.09± 0.01	0.06± 0.02
NEWS	0.11± 0.005	0.04± 0.001	0.13± 0.006	0.12± 0.003	0.11± 0.004

C) "Newsgroups-small"[8] prepared by S. Roweis. It has 100 binary attributes and 16,242 instances and is highly sparse (4% of the values is 1). Random splits of 10,000 train and 6,242 test instances were used for the classification experiments.

D) Digits: 8×8 binarized handwritten digits. We used 1100 examples from the digit classes 3 and 5 respectively (a total of 2200 instances). The dataset contains 30% 1's. This dataset was split randomly in 1600 train and 600 test instances.

The results for experiment I are shown in table 1 and figure 3. Note that the herding algorithms are deterministic and repetition would have resulted in the same values.

We observe that herding is successful in turning local average statistics into estimates of global quantities. Providing more information such as joint probabilities over triplets does significantly improve the result (the triplet results for Digits and News took too long to run due to the large number of triplets involved). Also of interest is the fact that for the low dimensional data H.XX outperformed PL but for the high-D datasets the opposite was true while both methods seem to leverage the same second order statistics (even though PL needs the actual data to learn its model).

The results for the classification experiment are shown in table 2. On all tasks the online learning of a linear logistic regression classifier did just as well as running logistic regression on the original data directly. This implies that the herding algorithm generates the information necessary for classification and that the decision boundary can be learned online during herding. Interestingly, the PL procedure significantly outperformed all standard classifiers as well as herding on the Newsgroup data. This implies that a more sophisticated decision boundary is warranted for this data.

[8]Downloaded from:
http://www.cs.toronto.edu/ roweis/data.html

To see if the herding sequence contained the information necessary to estimate such a decision boundary we reran PL on the first 10,000 pseudo-samples generated by herding resulting in an error of 0.04, answering the question in the affirmative. A plot of the herding pseudo-samples as compared to the original data was shown in Figure 1.

7. Outlook

We have studied a new herding algorithm that converts average sufficient statistics into a sequence of pseudo-samples from which statistics of interest can be estimated. The setup is similar to the maximum entropy principle. However, herding bypasses the model fitting procedure completely resulting in a more efficient algorithm that is better suited for hardware implementation. Besides these computational considerations it may also shed some light on how to compute with dynamic synapses for which some empirical evidence exists in the neuroscience literature.

We emphasize that to the best of our knowledge, the sequence of weights generated through herding is not even approximately a sample from some meaningful Bayesian posterior distribution. Recall that by changing the stepsize we can change the size of the weights to any desirable scale, which can therefore not represent posterior uncertainty. In addition, Bayesian inference in undirected graphical models is even harder than maximum likelihood learning (Murray & Ghahramani, 2004; Welling & Parise, 2006) while we argue that herding is simpler computationally.

We only considered problems where variables received direct input from data. The logical next step is to consider "hidden units", or variables that only receive input indirectly through other variables. Another natural extension is a herding algorithm for highly structured conditional random fields.

Many questions remain open and require additional

study. For instance, I: Can we modify herding to handle inconsistent or noisy constraints? II: Can we formulate a neurally more plausible "directed" version of herding where signals flow in one direction only. III: Can we prove the existence of an invariant distribution for herding and gain insight from studying its properties? IV: Is the herding dynamics chaotic? Does the attractor set of the weight sequence $\{\mathbf{w}_t\},\ t = 1 : \infty$ have fractal dimension (i.e. is it a strange attractor)?

Acknowledgements

This material is based upon work supported in part by the National Science Foundation under Award Number IIS-0447903 and IIS-0535278 and by ONR-MURI under Grant No. 00014-06-1-073. We thank N. LeRoux, Y. Bengio and R. Palais for feedback.

References

Besag, J. (1977). Efficiency of pseudo-likelihood estimation for simple Gaussian fields. *Biometrika*, *64*, 616–618.

Ganapathi, V., Vickrey, D., Duchi, J., & Koller, D. (2008). Constrained approximate maximum entropy learning. *Proceedings of the Twenty-fourth Conference on Uncertainty in AI* (pp. 196–203).

Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *6*, 721–741.

Hinton, G. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, *14*, 1771–1800.

Hopfield, J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, *79*, 2554–2558.

Hyvarinen, A. (2005). Estimation of non-normalized statistical models using score matching. *Journal of Machine Learning Research*, *6*, 695–709.

Jaynes, E. (1957). Information theory and statistical mechanics. *Physical Review*, *106*, 620–630.

Lafferty, J. (1999). Additive models, boosting, and inference for generalized divergences. *COLT: Proceedings of the Workshop on Computational Learning Theory* (pp. 125–133).

Lebanon, G., & Lafferty, J. (2002). Boosting and maximum likelihood for exponential models. *Neural Information Processing Systems* (pp. 447–454).

Levina, A., Herrmann, J., & Geisel, T. (2007). Dynamical synapses causing self-organized criticality in neural networks. *Nature Physics*, *3*, 857 – 860.

Maass, W., & Zador, A. M. (1998). Dynamic stochastic synapses as computational units. *Advances in Neural Information Processing Systems* (pp. 903–917). MIT Press.

Murray, I., & Ghahramani, Z. (2004). Bayesian learning in undirected graphical models: approximate MCMC algorithms. *Proceedings of the 14th Annual Conference on Uncertainty in AI* (pp. 392–399).

Neal, R. (1992). Connectionist learning of belief networks. *Articial Intelligence*, *56*, 71–113.

Pantic, L., Torres, J., Kappen, H., & Gielen, C. (2002). Associative memory with dynamic synapses. *Neural Computation*, *14*, 2903–2923.

Parise, S., & Welling, M. (2005). Learning in markov random fields: An empirical study. *Proc. of the Joint Statistical Meeting*.

Teh, Y., & Welling, M. (2002). The unified propagation and scaling algorithm. *Neural Information Processing Systems* (pp. 953–960).

Tieleman, T. (2008). Training restricted boltzmann machines using approximations to the likelihood gradient. *Proceedings of the International Conference on Machine Learning* (pp. 1064–1071).

Welling, M., & Parise, S. (2006). Bayesian random fields: The Bethe-Laplace approximation. *Proc. of the Conf. on Uncertainty in Artificial Intelligence* (pp. 512–519).

Younes, L. (1999). On the convergence of Markovian stochastic algorithms with rapidly decreasing ergodicity rates. *Stochastics An International Journal of Probability and Stochastic Processes*, *65*, 177–228.

Yuille, A. (2004). The convergence of contrastive divergences. *Advances in Neural Information Processing Systems* (pp. 1593–1600).

Zhu, S., & Liu, X. (2002). Learning in Gibbsian fields: How accurate and how fast can it be? *IEEE Trans. on Pattern Analysis and Machine Intelligence*, *24*, 1001–1006.

Zhu, S., Wu, Z., & Mumford, D. (1997). Minimax entropy principle and its application to texture modeling. *Neural Computation*, *9*, 1627–1660.