NeurIPS 2026

AFRL

Bayesian Backprop as Belief Propagation: Single-Pass Predictive Uncertainty

Wen Dong

Air Force Research Laboratory

The views expressed are those of the authors and do not reflect the official guidance or position of the United States Government, the Department of Defense or of the United States Air Force. The content or appearance of hyperlinks does not reflect an official DoD, Air Force, Air Force Research Laboratory position or endorsement of the external websites, or the information, products, or services contained therein.

Current uncertainty practices

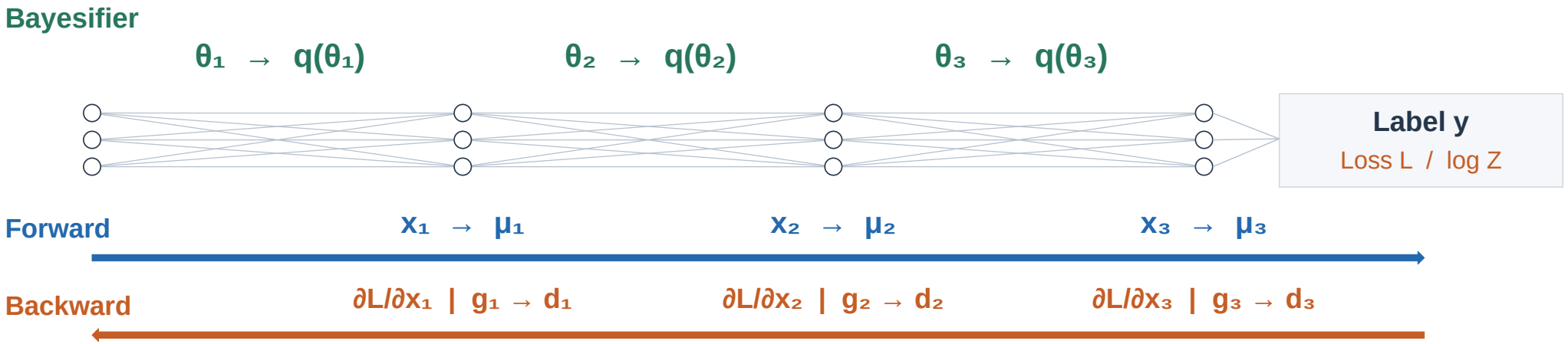
Practice	Learning and prediction
Temperature scaling	Recalibrate a trained model’s confidence scores.
Deep ensembles	Train several networks and combine their predictions.
MC dropout	Repeat prediction with different dropout masks.
VI, Laplace, SWAG	Approximate weight beliefs, often sampling for prediction.
Single-pass methods	Uncertainty-aware heads and existing moment propagation.

Repeated model evaluations can make uncertainty costly to serve.

Bayesify converts selected neural modules into moment-propagation modules and generates their belief-update rules.

Ordinary and Bayesified forward–backward propagation

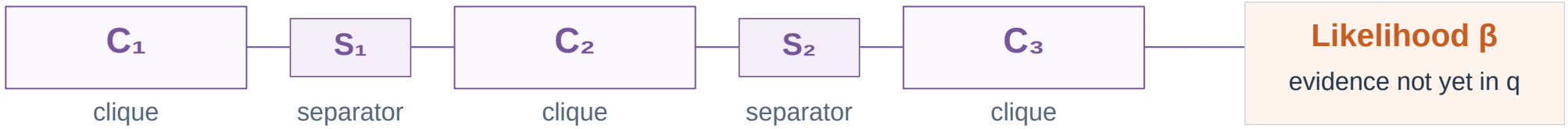
The same neural graph carries values or moments, and learns point weights or beliefs.



Operation	Ordinary network	Bayesified network
Learned state	Point weights θ	Parameter beliefs q(θ)
Forward pass	Activation values x	Moments μ (stored as mean / variance)
Backward pass	Loss derivatives	Moment derivatives g → corrections d
Prediction	One forward evaluation	One forward moment pass

Exact messages become posterior corrections

Theorem 3.1 · Fully represented exponential-family messages on a normalized junction tree



Combine and marginalize → forward moments μ + cached normalizers

$$Z = \mathbb{E}_q[\beta]$$

Differentiate log evidence → backward moment corrections d

Why the derivative is a posterior correction

$$\log Z = A_\beta(\eta) - A(\eta)$$

Z: local normalizer η: natural parameters
A: current log partition Aβ: reweighted log partition

$$d = Fg = \nabla_\eta \log Z = \mu^{\text{post}} - \mu$$

g = ∇μ log Z: ordinary moment derivative
F: Fisher matrix = covariance of retained statistics

One reverse sweep, followed by one moment correction

$$d_p = F_p g_p^{\text{loc}} + \sum_c F_p J_{c,p}^{\mathsf{T}} F_c^{-1} d_c$$

p: parent c: child J: derivative of the forward moment map
Local term: direct evidence / local-normalizer derivatives.

$$\mu_p^{\text{post}} = \mu_p + d_p$$

posterior = reference + correction
At a prior input, μ is the prior.

EP maps inference ingredients to backward computation

Theorem 3.2 · Selected moments of a local likelihood-reweighted belief

$$d = F\nabla_\mu \log Z = \mu^\star - \mu$$

$$q^\star \propto q\beta, \qquad Z = \mathbb{E}_q[\beta]$$

μ⋆: selected moments of the local target q⋆

Exact local Z: exact selected target moments. Approximate Z: corresponding approximate correction.

The tested diagonal-Gaussian implementation omits higher moments and correlations.

EP ingredient	What the reverse computation does
Reference / cavity	Choose μ and F after excluding the likelihood-site contribution being revised.
Label / observation head	Cached likelihood β + output belief → Z and the terminal correction.
Layer / moment projection	Differentiate the chosen forward moment map between Fisher conversions.
Skip / shared activation	Use one belief state; sum derivatives from all paths.
Prior / persistent update	Retain the prior once in EP bookkeeping; complete the scheduled, damped weight update.

One local backward contribution

$$d_{p\leftarrow c} = F_p\, J_{c,p}^{\mathsf{T}}\, F_c^{-1} d_c$$

Decode the output correction → apply the local chain rule → encode the input / weight correction.

Each module computes its own belief correction

Local cached moments + an incoming derivative carrying downstream evidence

Scalar connection z = wa: input a and weight w have independent Gaussian beliefs

Forward: calculate output moments

$$m_z = m_w m_a$$

$$v_z = m_w^2 v_a + v_w(m_a^2 + v_a)$$

$$g_{m_z} = \frac{\partial \ell}{\partial m_z}, \qquad g_{v_z} = \frac{\partial \ell}{\partial v_z}$$

ℓ: local EP log normalizer

Backward: generate weight corrections

$$d_{m_w} = v_w\left(m_a g_{m_z} + 2 m_w v_a g_{v_z}\right)$$

$$d_{s_w} = 2 m_w d_{m_w} + 2 v_w^2 (m_a^2 + v_a) g_{v_z}$$

m: mean v: variance s = m² + v
Above: g differentiates (m, v); d corrects (m, s).

MFVI: change the learning objective

Expected log likelihood − KL to the prior

$$g^{\mathrm{VI}} = \nabla_\mu \Psi + \eta_0 - \eta$$

$$\eta^+ = \eta + \rho\, g^{\mathrm{VI}}, \qquad \mu^+ = \nabla A(\eta^+)$$

Ψ: expected log likelihood; η_0, η: prior / current natural parameters
ρ: step size; ∇A converts natural parameters to moments.

Covariance blocks: retain dependence

Activation and weight correlations use
richer local moment maps and Fisher blocks.



c: covariance; richer blocks use more memory.

Near-ensemble uncertainty, less prediction time

Diagonal-Gaussian EP · WRN-28-10 · CIFAR-10 / CIFAR-100 OOD

32% lower calibration error

ECE: 2.5% → 1.7%, relative to temperature scaling

1.45× faster prediction than ensemble

5.52× faster than 20-sample MC dropout

Method	NLL ↓	ECE % ↓	OOD AUROC ↑	Training-step time ↓	Prediction time ↓
Deterministic + TS[1]	0.205	2.5	0.78	1.00×	1.00×
Ensemble (5)	0.180	1.5	0.88	≈5×[2]	4.77×
MC dropout (20)	0.190	2.0	0.82	≈1×[2]	18.16×
Bayesify (EP)	**0.186**	**1.7**	**0.87**	**3.72×**	**3.29×**

Quality: three-seed means. Timing: same hardware, prediction batch size 1; ensemble members run serially.

[1] Quality uses temperature scaling; time is relative to the ordinary deterministic model.

[2] Estimated training-step times. A training-step comparison does not measure total-training time.

Paper

One network learns beliefs by backward moment corrections and predicts uncertainty in one forward moment pass.