Weekly Report KARIMI 2021-07-23

My work this week has been focussing on three projects which should occupy me for the next 4 weeks.

1. Paddle paddle implementation of Bayesian Neural Networks.
2. Additional experiments for the STANLey method (EBM)
3. Experiments and Theory for the Quantized sEM under Federated settings.

1 Paddle Paddle: Bayesian Neural Networks

I have started to implement on paddle paddle Bayesian variants of deep neural networks. Apart from the fact that this implementation will eventually be needed for our MISSO paper, I am mainly focussing on the different classes allowing one to use random weights for neural networks.

Indeed, I believe having those classes (BNN, RandomWeights, Normal distribution) ready will facilitate the development of any research work towards Bayesian Deep Learning or even Random Graphs modeling where nodes are latent variables associated with random distributions.

For now I have constructed the main framework which is comprised of `BayesianNeuralNet` and `RandomWeights` classes, see the snippet below, as well as base distribution and `Normal` distribution object in order to at least be able to assign a Gaussian distribution to the weights.

The optimization part, starting with plain Variational Inference and moving to our MISSO method will be done in a second step.

```python
import numpy as np
import paddle

import paddle.fluid as fluid

from bayesian.framework.RandomWeights import RandomWeights
from bayesian.distributions import *

class RandomWeights(object):
    def __init__(self, bn, name, dist, observation=None, **kwargs):
        self._bn = bn
        self._name = name
        self._dist = dist
        self._dtype = dist.dtype
        self._n_samples = kwargs.get("n_samples", None)
        self._observation = observation
        super(RandomWeights, self).__init__()


class BayesianNeuralNet(paddle.nn.Layer):
    def __init__(self, observed=None):
        super(BayesianNeuralNet, self).__init__()
        self._nodes = {}
        self._cache = {}
        self._observed = observed if observed else {}
```

2 EBM Stanley for in-paintings tasks

As the experimental part of the STANLey paper we submitted at ICCV is the weak part of the contribution, Jianwen suggested to work on the image completion (also known as in-paintings) task. The first experiment I have done is on the CelebA dataset [1]. Figure 1 corresponds to random samples where STANLey has performed image completion. I am working on inferring similar images with baseline methods that we presented in the paper and in the rebuttal, i.e. MH, HMC and Langevin.

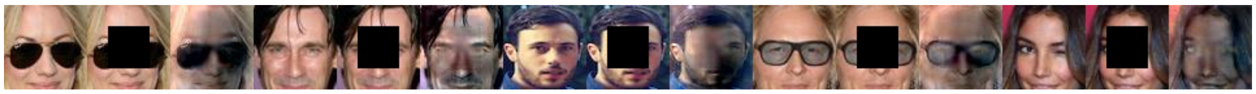

Figure 1: CelebA: Image in-paintings with STANLey

The improved STANLey paper will thus be improved in terms of baselines (3 additional ones presented in the rebuttal for CIFAR-10) and in terms of tasks: a) GMM toy example, 2) image generation on Flowers and CIFAR and 3) Image in-paintings.

3 Quantized and Compressed sEM for Federated Learning

The final method I am considering for solving federated optimization using the EM is the following:

Algorithm 1 Quantized and Compressed FL-SAEM with Periodic Statistics Averaging

```
1:  Input: Compression operator $\mathcal{C}(\cdot)$, number of rounds $R$, initial parameter $\theta_0$.
2:  for each $r = 1$ to $R$ do
3:      for parallel for device $i \in D^r$ do
4:          Set $\hat{\theta}_i^{(0,r)} = \hat{\theta}^{(r)}$.        ▷ Initialize each worker with current global model
5:          Draw M samples $z_{i,m}^{(r)}$ under model $\hat{\theta}_i^{(r)}$ via Quantized LD:        ▷ Local Quantized MCMC step
6:          for $k = 1$ to $K$ do
7:              Compute the quantized gradient of $\nabla \log p(z_i|y_i;\hat{\theta}_i^{(k)})$:
                    $$g_i(k,m) = \mathsf{C}_j^{(\ell)}\left(\nabla_j f_{\theta_t}(z_i^{(k-1,m)}), \xi_j^{(k)}\right) \quad \text{where} \quad \xi_j^{(k)} \sim \mathcal{U}_{[a,b]}$$
8:              Sample the latent data using the following chain:
                    $$z_i^{(k,m)} = z_i^{(k-1,m)} + \frac{\gamma_k}{2} g_i(k,m) + \sqrt{\gamma_k}\mathsf{B}_k,$$
                where $\mathsf{B}_t$ denotes the Brownian motion and $m \in [M]$ denotes the MC sample.
9:          end for
10:         Assign $\{z_i^{(r,m)}\}_{m=1}^M \leftarrow \{z_i^{(K,m)}\}_{m=1}^M$.
11:         Compute $\tilde{S}_i^{(r+1)}$ and its Top-$k$ variant $\ddot{S}_i^{(k+1)} = \mathcal{C}\left(\tilde{S}_i^{(k+1)}\right)$.        ▷ Compressed local statistics
12:         Workers send local statistics $\tilde{S}_i^{(k+1)}$ to server.        ▷ Single round of communication
13:     end for
14:     Server computes global model:        ▷ (Global) M-Step using aggregated statistics
                    $$\hat{\theta}^{(r+1)} = \overline{\theta}(\ddot{S}^{(r+1)})$$
        where $\ddot{S}^{(r+1)} = (\ddot{S}_i^{(r+1)}, i \in D_r)$ and send global model back to the devices.
15: end for each
```

Implementation of Algorithm 1 is underway and initial plots should be ready for next week's report.

References

[1] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15(2018):11, 2018.