Semi-supervised Learning with Deep Generative Models

Kingma et. al. (2014)

Tyler Brown

CS 7180

Motivating Question

How can we model data of increasing size when obtaining label information is difficult?

High-level Answer

We can estimate missing label information by using a probabilistic model.

Most Relevant Previous Work

Pitelis, N., Russell, C., and Agapito, L. (2014). Semi-supervised learning using an unsupervised atlas. *In Proceddings of the European Conference on Machine Learning (ECML)*, volume LNCS 8725, pages 565 – 580.

- Observing that high-dimensional datasets often lie on or near *manifolds* of locally low rank can help avoid the *curse of dimensionality*
- Experiments show how using unlabelled data to learn the underlying manifold improves classifier accuracy when trained on limited labelled data
    1. **Unsupervised learning of the underlying manifold:** Approximate the manifold of data on the original space by fitting an atlas of low-dimensional overlapping affine charts.
    2. **Supervised training of an SVM:** Proposed a new family of Mercer Kernels for SVM-based supervised learning which uses soft-assignment of datapoints to the underlying low-dimensional affine charts to generate the kernels

Specifying the Probabilistic Model for Missing Labels

Kingma et. al. (2014)

- Data appears as pairs $(\mathbf{X}, \mathbf{Y}) = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$ with the i-th observation $x_i \in \mathbb{R}^D$ and a corresponding class label $y_i \in \{1, ..., L\}$
  - Each pair of observations (x_i, y_i) has a corresponding latent variable z_i
  - Empirical distribution over the labelled and unabelled subsets is referred to as $\tilde{p}_l(\mathbf{x}, y)$ and $\tilde{p}_u(\mathbf{x})$
- We can estimate y_i for x_i in distribution $\tilde{p}_u(\mathbf{x})$ by finding the maximum probability of $p(y_i)$ by using a set of features related to z_i and a predictive model
  1. **Latent-feature discriminative model (M1)**
  2. **Generative semi-supervised model (M2)**
  3. **Stacked generative semi-supervised model (M1+M2)**

Bayes Rule is used when specifying M1 & M2

$$
\begin{aligned}
p(x,y) &= p(x)p(y|x) \\
&= p(y)p(x|y) \\
p(x|y) &= \frac{p(x)p(y|x)}{p(y)}
\end{aligned}
$$

for models M1 [1], $p(z|x)$, and M2 [2]; $p(y|x)$

[1] Kingma et. al. (2014) equation (1)
[2] Kingma et. al. (2014) equation (2)

(M1) Latent-feature discriminative model

$$y \Leftarrow p(z|x) = \frac{p(z)p(x|z)}{p(x)}$$

where

$p(z) = \mathcal{N}(z|0, I)$ Gaussian distribution of z given a missing label y

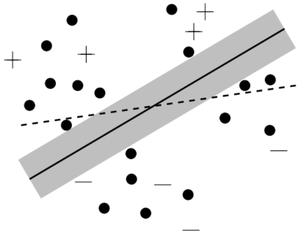
$p(x|z) = f(x; z, \theta)$ likelihood function, parameters θ of a set of z

$p(x) = \tilde{p_u}(x)$ unlabelled subset of $x_i \in \mathbb{R}^D$

Kingma et. al. (2014) eq. (1)

(M1) Predicting Class Labels y

Approximate samples from the posterior distribution over the latent variables $p(z|x)$ are used as features to train a classifier that predicts class labels y



(transductive) SVM[3] finds the largest margin w.r.t. the training **and** the test vectors

[3] See Figure 6.2 from Chapelle, O., B. Schölkopf, and A. Zien. "Semi-Supervised Learning." (2006).

(M2) Generative semi-supervised model

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})} \approx \frac{p_\theta(\mathbf{x}|y,\mathbf{z})p(y)}{p(\mathbf{x})}$$

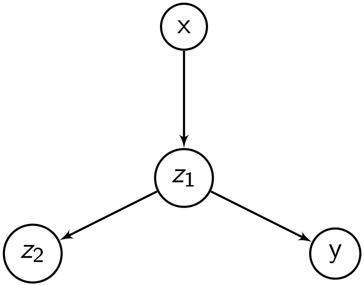
where

$p(y) = Cat(y\|\pi)$	multinomial distribution, y can be latent
$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}\|\mathbf{0},\mathbf{I})$	Gaussian distribution of z when missing y
$p_\theta(\mathbf{x}\|y,z) = f(\mathbf{x}; y, \mathbf{z}, \theta)$	likelihood function, nonlinear parameters
$p(x)$	all x in dist. of real numbers; $x \in \mathbb{R}^D$

Stacked generative semi-supervised model (M1 + M2)

Combine M1 and M2

1. Learn a new latent representation z_1 from M1
2. Use embeddings from z_1 instead of raw data x, to create a generative semi-supervised model M2



$$p_\theta(x, y, z_1, z_2) = p(y)p(z_2)p_\theta(z_1|y, z_2)p_\theta(x|z_1)$$

Scaling Up: Lower Bound Objective

Lower Bound Objective[4]: computation of the exact posterior distribution is intractable for models M1 and M2

$$\text{M1: } q_\phi(z|x) = \mathcal{N}(z|u_\phi(x), \text{diag}(\sigma^2_\phi(\mathbf{x}))) \tag{3}$$

$$\begin{aligned} \text{M2: } & q_\phi(\mathbf{z}|y, \mathbf{x}) = \mathcal{N}(\mathbf{z}|\mu_\phi(y, \mathbf{x}), \text{diag}(\sigma^2_\phi(\mathbf{x}))); \\ & q_\phi(y|\mathbf{z}) = \text{Cat}(y|\pi_\phi(x)), \end{aligned} \tag{4}$$

where

$\sigma_\phi(x)$	vector of standard deviations
$\pi_\phi(x)$	probability vector
$\mu_\phi(x), \sigma_\phi(x), \pi_\sigma(x)$	Maximum likelihood Priors (MLPs)

[4]Kingma et. al. (2014) equations (3), (4)

Scaling Up: M1 Model Objective

Variational bound $\mathcal{J}(x)$ on the marginal likelihood of a single data point is[5]

$$\log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - KL[q_\phi(z|x)||p_\theta(z)] = -\mathcal{J}(x)$$

Approximate posterior is used as a feature extractor for the labelled data set, and the features used for training the classifier

[5] Kingma et. al. (2014) Equation 5

Scaling Up: M2 Model Objective

When y_i is observed for the (x_i, y_i) data pair, extend from M1

$$\log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x,y)}[\log p_\theta(x|y,z) + \log p_\theta(y) + \log p(z) - q_\theta(z|x,y)]$$
$$= -\mathcal{L}(x,y)$$

In the case where y_i is missing,

$$\begin{aligned}\log p_\theta(x) &\geq \mathbb{E}_{q_\phi(y,z|x)}[\log p_\theta(x|y,z) + \log p_\theta(y) \\ &\qquad + \log p(z) - \log q_\phi(y,z|x)] \\ &= \sum_y q_\phi(y|x)(-\mathcal{L}(x,y)) + \mathcal{H}(q_\phi(y|x)) \\ &= -\mathcal{U}(x)\end{aligned}$$

The bound on the marginal likelihood for the entire dataset is now[6]

$$\mathcal{J} = \sum_{(x,y)\sim\tilde{p}_l} \mathcal{L}(x,y) + \sum_{x\sim\tilde{p}_u} \mathcal{U}(x)$$

[6] See Kingma et. al. (2014) equations 6-9

Optimization Techniques

- Bounds from M1 and M2 objective function equations provides for optimization of both θ and ϕ parameters
  - Optimization can be done *jointly* without using the EM Algorithm
- Use deterministic reparameterizations of the expectations in the objective function and *Monte Carlo* approximation
- Previous work refers to this as *stochastic gradient variational Bayes*[7] and *stochastic backpropagation*[8]

---

[7] Kingma, D. P. and Welling, M. (2014). Auto-encoding variational Bayes. *In Proceedings of the International Conference on Learning Representations (ICLR)*.

[8] Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. *In Proceedings of the International Conference on Machine Learning (ICML)*, volume 32 of JMLR WCP.

Optimization Algorithms

Algorithm 1 Learning in model M1

while generativeTraining() **do**
  $\mathcal{D} \leftarrow$ getRandomMiniBatch()
  $\mathbf{z}_i \sim q_\phi(\mathbf{z}_i|\mathbf{x}_i) \quad \forall \mathbf{x}_i \in \mathcal{D}$
  $\mathcal{J} \leftarrow \sum_n \mathcal{J}(\mathbf{x}_i)$
  $(\mathbf{g}_\theta, \mathbf{g}_\phi) \leftarrow (\frac{\partial \mathcal{J}}{\partial \theta}, \frac{\partial \mathcal{J}}{\partial \phi})$
  $(\boldsymbol{\theta}, \boldsymbol{\phi}) \leftarrow (\boldsymbol{\theta}, \boldsymbol{\phi}) + \boldsymbol{\Gamma}(\mathbf{g}_\theta, \mathbf{g}_\phi)$
end while
while discriminativeTraining() **do**
  $\mathcal{D} \leftarrow$ getLabeledRandomMiniBatch()
  $\mathbf{z}_i \sim q_\phi(\mathbf{z}_i|\mathbf{x}_i) \quad \forall \{\mathbf{x}_i, y_i\} \in \mathcal{D}$
  trainClassifier($\{\mathbf{z}_i, y_i\}$)
end while

Algorithm 2 Learning in model M2

while training() **do**
  $\mathcal{D} \leftarrow$ getRandomMiniBatch()
  $y_i \sim q_\phi(y_i|\mathbf{x}_i) \quad \forall \{\mathbf{x}_i, y_i\} \notin \mathcal{O}$
  $\mathbf{z}_i \sim q_\phi(\mathbf{z}_i|y_i, \mathbf{x}_i)$
  $\mathcal{J}^\alpha \leftarrow$ eq. (9)
  $(\mathbf{g}_\theta, \mathbf{g}_\phi) \leftarrow (\frac{\partial \mathcal{L}^\alpha}{\partial \theta}, \frac{\partial \mathcal{L}^\alpha}{\partial \phi})$
  $(\boldsymbol{\theta}, \boldsymbol{\phi}) \leftarrow (\boldsymbol{\theta}, \boldsymbol{\phi}) + \boldsymbol{\Gamma}(\mathbf{g}_\theta, \mathbf{g}_\phi)$
end while

Gradients w.r.t. generative parameters θ and variational parameters ϕ can be efficiently computed as expectations of simple gradients[9]

$$\nabla_{\{\theta,\phi\}} \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] = \mathbb{E}_{\mathcal{N}(\epsilon|0,I)}[\nabla_{\{\theta,\phi\}} \log p_\theta(x|\mu_\theta(x) + \sigma_\phi(x) \odot \epsilon)].$$

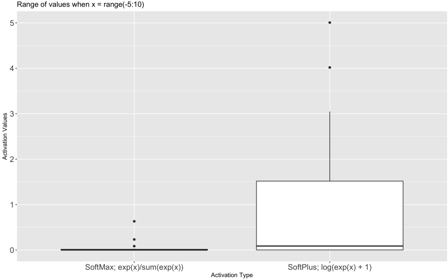
[9]Kingma et. al. (2014) equation 11, regarding M1

Results: Benchmark Classification

Table 1: Benchmark results of semi-supervised classification on MNIST with few labels.

N	NN	CNN	TSVM	CAE	MTC	AtlasRBF	M1+TSVM	M2	M1+M2
100	25.81	22.98	16.81	13.47	12.03	8.10 ($\pm$ 0.95)	11.82 ($\pm$ 0.25)	11.97 ($\pm$ 1.71)	**3.33** ($\pm$ 0.14)
600	11.44	7.68	6.16	6.3	5.13	–	5.72 ($\pm$ 0.049)	4.94 ($\pm$ 0.13)	**2.59** ($\pm$ 0.05)
1000	10.7	6.45	5.38	4.77	3.64	3.68 ($\pm$ 0.12)	4.24 ($\pm$ 0.07)	3.60 ($\pm$ 0.56)	**2.40** ($\pm$ 0.02)
3000	6.04	3.35	3.45	3.22	2.57	–	3.49 ($\pm$ 0.04)	3.92 ($\pm$ 0.63)	**2.18** ($\pm$ 0.04)

- Varying the size of labelled data from 100 to 3000
- SoftPlus Activation function



Results: Image Classification

Table 2: Semi-supervised classification on the SVHN dataset with 1000 labels.

KNN	TSVM	M1+KNN	M1+TSVM	M1+M2
77.93 ($\pm$ 0.08)	66.55 ($\pm$ 0.10)	65.63 ($\pm$ 0.15)	54.33 ($\pm$ 0.11)	**36.02** ($\pm$ 0.10)

Table 3: Semi-supervised classification on the NORB dataset with 1000 labels.

KNN	TSVM	M1+KNN	M1+TSVM
78.71 ($\pm$ 0.02)	26.00 ($\pm$ 0.06)	65.39 ($\pm$ 0.09)	**18.79** ($\pm$ 0.05)

- No comparative results in semi-supervised setting exists for SVHN and NORB image data sets
- Performed nearest-neighbor and TSVM classification with RBF kernels
- Compared performance on features generated by their latent-feature discriminative model to the original features

Discussion

You can use this model for a few things
TODO: just summarize their discussion

Any questions? Thanks!