Deep Learning for Complex Image Analysis

Kevin Liang, Yunchen Pu

Duke University

Adviser: Lawrence Carin, kevin.liang@duke.edu

Consortium for Verification Technology (CVT)

Introduction

- Image analysis plays a key role in detecting potential nuclear activity from a distance.
- Convolutional Neural Networks (CNNs) in recent years have proven themselves effective tools for image analysis tasks, such as image classification and object detection. Additionally, Recurrent Neural Networks (RNNs) have also been employed to generate image captions learned from CNN image features.
- Most previous work has been done in fully supervised settings. In practice though, most data are unlabeled. In such settings, unsupervised models, such as the recently developed Deep Generative Deconvolutional Model, can infer latent image features, but at a cost of more computation time.
- We propose a new variational autoencoder set-up for image analysis, using a CNN as an image encoder for the distribution of latent parameters for a DGDN decoder. The CNN encoder (also termed "recognition model") allows for much faster test-time evaluation than the DGDN alone. A Bayesian SVM and an RNN (using the latent features shared by the CNN and DGDN) utilizes any available image labels and captions, respectively, for learning; all VAE model parameters are learned jointly. This allows for a *semi-supervised* approach.

Model

Image Decoder: Deep Generative Deconvolutional Network (DGDN)

Consider N images $\{\boldsymbol{X}^{(n)}\}_{n=1}^{N}$, with $\boldsymbol{X}^{(n)} \in \mathbb{R}^{N_x \times N_y \times N_C}$

$$\begin{aligned}
&\text{Layer 2:} && \tilde{\mathbf{S}}^{(n,2)} = \textstyle\sum_{k_2=1}^{K_2} \mathbf{D}^{(k_2,2)} * \mathbf{S}^{(n,k_2,2)} && (1)\\
&\text{Unpool:} && \mathbf{S}^{(n,1)} \sim \text{unpool}(\tilde{\mathbf{S}}^{(n,2)}) && (2)\\
&\text{Layer 1:} && \tilde{\mathbf{S}}^{(n,1)} = \textstyle\sum_{k_1=1}^{K_1} \mathbf{D}^{(k_1,1)} * \mathbf{S}^{(n,k_1,1)} && (3)\\
&\text{Data Generation:} && \mathbf{X}^{(n)} \sim \mathcal{N}(\tilde{\mathbf{S}}^{(n,1)}, \alpha_0^{-1}\mathbf{I}) && (4)
\end{aligned}$$

Stochastic Unpooling

Partition $\boldsymbol{S}^{(n,k_1,1)}$ into contiguous $p_x^{(1)} \times p_y^{(1)}$ pooling blocks. Let $\boldsymbol{z}_{i,j}^{(n,k_1,1)} \in \{0,1\}^{p_x^{(1)} p_y^{(1)}}$ be a one-hot vector.

$$\boldsymbol{z}_{i,j}^{(n,k_1,1)} \sim \text{Mult}(1, \boldsymbol{\theta}^{(n,k_1,1)})$$
$$\boldsymbol{\theta}^{(n,k_1,1)} \sim \text{Dir}(1/(p_x^{(1)} p_y^{(1)}))$$

The non-zero element of pooling block (i,j) is set to $\tilde{\boldsymbol{S}}_{i,j}^{(n,k_1,1)}$, while the rest are set to zero.

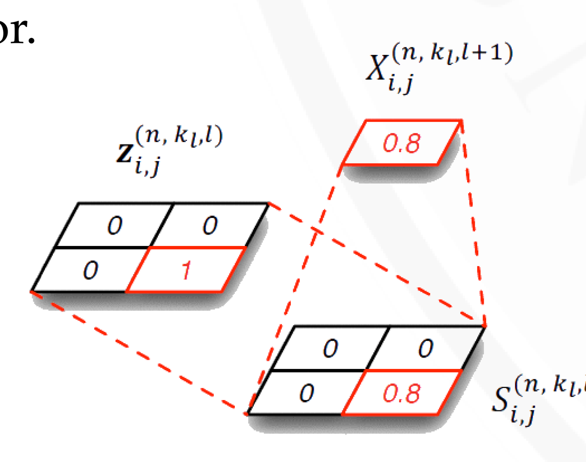

Figure 1: Stochastic unpooling example with 2 × 2 unpooling block

Image Encoder: Deep Convolutional Neural Network (CNN)

$$\begin{aligned}
&\text{Layer 1:} && \tilde{\mathbf{C}}^{(n,k_1,1)} = \mathbf{X}^{(n)} *_s \mathbf{F}^{(k_1,1)}\ ,\ k_1 = 1,\dots,K_1 && (5)\\
&\text{Pool:} && \mathbf{C}^{(n,1)} \sim \text{pool}(\tilde{\mathbf{C}}^{(n,1)}) && (6)\\
&\text{Layer 2:} && \tilde{\mathbf{C}}^{(n,k_2,2)} = \mathbf{C}^{(n,1)} *_s \mathbf{F}^{(k_2,2)}\ ,\ k_2 = 1,\dots,K_2 && (7)\\
&\text{Code Generation:} && s_n \sim \mathcal{N}\left(\boldsymbol{\mu}_\phi(\tilde{\mathbf{C}}^{(n,2)}), \text{diag}(\boldsymbol{\sigma}_\phi^2(\tilde{\mathbf{C}}^{(n,2)}))\right) && (8)
\end{aligned}$$

Stochastic Pooling

Partition $\tilde{\boldsymbol{C}}^{(n,k_1,1)}$ into contiguous $p_x^{(1)} \times p_y^{(1)}$ pooling blocks.

Each pooling block $\tilde{\boldsymbol{C}}_{i,j}^{(n,k_1,1)}$ is passed into a multi-layered perceptron to obtain a probability vector for a multinomial random variable, which then chooses which element within pooling block (i,j) to retain.

Labels and Captions

Generative Model for Labels: Bayesian SVM

Given labels $l_n \in \{1, \dots, C\}$, we design C one-versus-all binary SVM classifiers with $y_n^{(l)} \in \{-1,1\}$. If $l_n = l$, then $y_n^{(l)} = 1$, and $y_n^{(l)} = -1$ otherwise. The goal of the SVM is to find an $f(\boldsymbol{s})$ that minimizes:

$$\gamma \sum_{n=1}^{N} \max(1 - y_n f(\boldsymbol{s}_n), 0) + R(f(\boldsymbol{s}))$$

Generative Model for Captions: Recurrent Neural Network

A caption $\boldsymbol{Y}^{(n)} = \left(\boldsymbol{y}_1^{(n)}, \dots, \boldsymbol{y}_{T_n}^{(n)}\right)$ is generated sequentially by an RNN, with the probability of a particular caption being:

$$p(\boldsymbol{Y}^{(n)} | \boldsymbol{s}_n) = p\left(\boldsymbol{y}_1^{(n)} \middle| \boldsymbol{s}_n\right) \prod_{t=2}^{T_n} p(\boldsymbol{y}_t^{(n)} | \boldsymbol{y}_{<t}^{(n)}, \boldsymbol{s}_n)$$

Inference

Variational Learning of Model Parameters

The parameters $\{\boldsymbol{\phi}, \boldsymbol{\alpha}, \boldsymbol{\psi}\}$ of the image encoder $q_{\boldsymbol{\phi}}(\boldsymbol{s}, \boldsymbol{z}|\boldsymbol{X})$, the image decoder $p_{\boldsymbol{\alpha}}(\boldsymbol{X}|\boldsymbol{s}, \boldsymbol{z})$, and the caption/label model $p_{\boldsymbol{\psi}}(\boldsymbol{Y}|\boldsymbol{s})$ are optimized by maximizing the variational lower bound:

$$\mathcal{L}_{\phi,\alpha,\psi}(\mathbf{X}, \mathbf{Y}) = \xi\{\mathbb{E}_{q_\phi(s|\mathbf{X})}[\log p_\psi(\mathbf{Y}|s)]\} + \mathbb{E}_{q_\phi(s,z|\mathbf{X})}[\log p_\alpha(\mathbf{X}, s, z) - \log q_\phi(s, z|\mathbf{X})]$$

Diagram

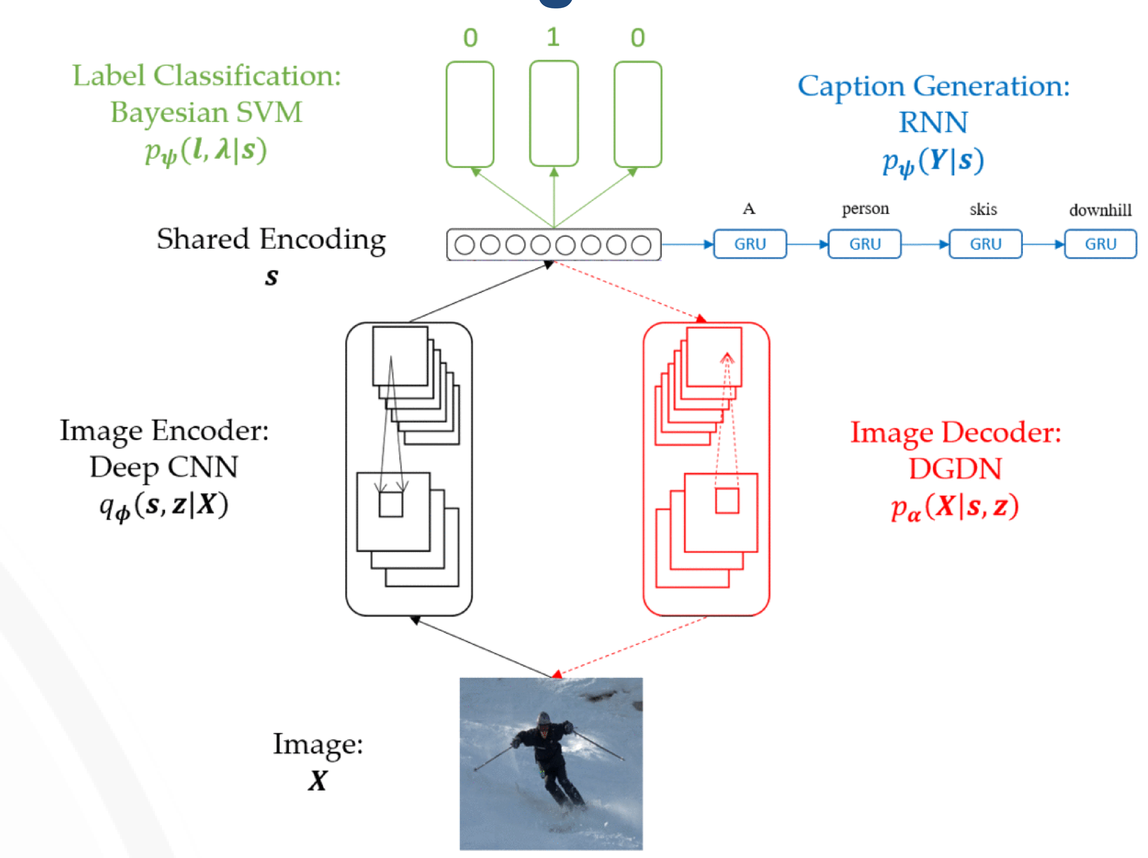

Figure 2: Variational Auto-Encoder set up

Experiments

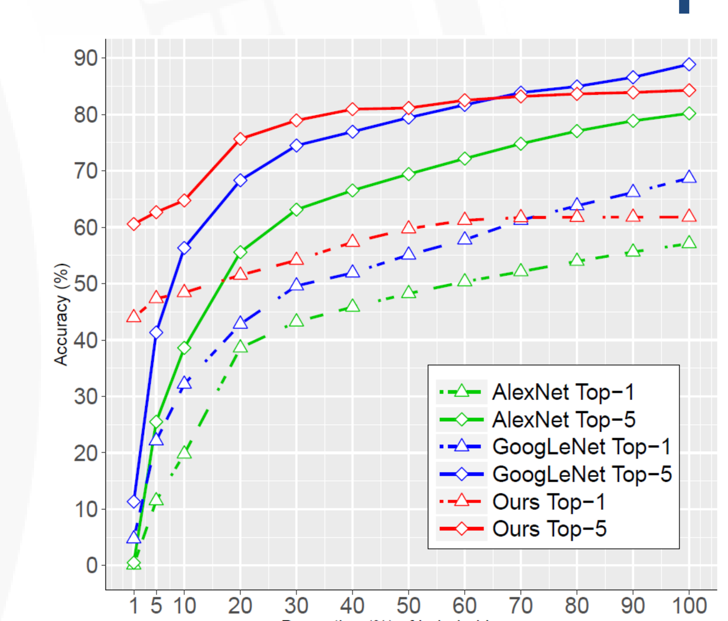

Figure 3: Semi-supervised classification accuracy on the validation set of ImageNet2012

Figure 4: Examples of generated captions from unseen images on the validation set of ImageNet2012

Table 1: Classification error (%) and testing time (ms per image) on benchmarks.

	MNIST		CIFAR-10		CIFAR-100		Caltech 101		Caltech 256	
	Test error	Test time	Test error	Test time	Test error	Test time	Test error	Test time	Test error	Test time
Gibbs [10]	0.37	3.1	8.21	10.4	34.33	10.4	12.87	50.4	29.5	52.3
MCEM [10]	0.45	0.8	9.04	1.1	35.92	1.1	13.51	8.8	30.13	8.9
VAE (Ours)	0.38	**0.007**	8.19	**0.02**	35.01	**0.02**	11.99	**0.3**	29.33	**0.3**

	ImageNet 2012			ImageNet Pretrained for			
				Caltech 101		Caltech 256	
	top-1 error	top-5 error	test time	test error	test time	test error	test time
MCEM [10]	37.9	16.1	14.4	6.85	14.1	22.10	14.2
VAE (Ours)	38.2	15.7	**1.0**	6.91	**0.9**	22.53	**0.9**

Table 2: Semi-supervised classification error (%) on MNIST. N is the number of labeled images per class.

N	TSVM	Deep generative model [27]		Ladder network [28]		Our model	
		M1+TSVM	M1+M2	Γ-full	Γ-conv	$\xi = 0$	$\xi = N_x/(C\rho)$
10	16.81	11.82± 0.25	3.33 ± 0.14	3.06 ± 1.44	**0.89**±0.50	5.83 ± 0.97	1.49 ± 0.36
60	6.16	5.72± 0.05	2.59 ±0.05	-	-	2.19 ± 0.19	**0.77** ± 0.09
100	5.38	4.24± 0.07	2.40 ±0.02	1.53 ± 0.10	-	1.75 ± 0.14	**0.63** ± 0.06
300	3.45	3.49± 0.04	2.18 ±0.04	-	-	1.42 ± 0.08	**0.51** ± 0.04

This work was funded in-part by the Consortium for Verification Technology under Department of Energy National Nuclear Security Administration award number DE-NA0002534