resit2022

52041DEL6Y Deep Learning 1 22/23 (Period 2) · 10 exercises · 100.0 points

Exam: MCQ #15880633

8 pts · Last updated 15 Mar, 2023, 14:00 · Saved in Deep Learning questions

2 PTS · MULTIPLE RESPONSE

Which of the following is NOT a common approach for self-supervised learning?

☐	Pretraining on a large dataset and fine-tuning on a smaller task-specific dataset	-1.0
☑	**Applying self-supervision to predict the weights of a supervised network**	1.0
☐	Using the data itself as the supervision signal	-1.0
☑	**Using hand-labeled data to train the model**	1.0

2 PTS · MULTIPLE RESPONSE

What statements are true about contrastive learning?

☑	**When the dataset is very large and very diverse, a higher temperature (τ) might be required.** Feedback This is true. When the dataset is diverse and large, this makes hard negatives more rare and therefore the task "too easy". To counter-act this, a higher temperature can help.	2.0
☐	When the dataset is very large and very diverse, a lower temperature (τ) might be required. Feedback see above	-1.0
☐	For any given sample, in self-supervised contrastive learning we define positive and negative samples. The positives are chosen in the dataloader by picking samples with the same class-label. Feedback Wrong. Self-supervised implies no usage of class-labels.	-1.0
☐	When training on multiple GPUs, one should stick to a normal BatchNorm implementation. Feedback False, batchnorm can leak information about from which GPU a particular sample came and thus make the contrastive task easier. Special implementations such as SyncBatchNorm or ShuffleBatchNorm or simply using other normalisation layers alleviates this.	-1.0

2 PTS · MULTIPLE RESPONSE

What statements about generative models are true?

- [ ] The improvement in GANs largely came from using larger datasets -1.0

  Feedback
  False, eg going from DCGAN to BigBiGAN or StyelGAN did not involve a change in dataset (all used ImageNet) eg

- [x] **Generative models can and are used for malicious purposes.** 1.0

  Feedback
  True. Eg deep fakes.

- [x] **Evaluating the quality of generative models objectively is difficult.** 1.0

  Feedback
  True. While measures like FID and IS exist, they are imperfect and not good enough for critical applications.

- [ ] The main goal of generative modelling is to learn a model that can accurately predict the value of a discrete variable -1.0

  Feedback
  False, the main goal is to learn a model that can generate new examples that are similar to the training data, potentially even providing probability estimates.

2 PTS · MULTIPLE CHOICE

What is a NeRF?

- (●) **A machine learning model for 3D reconstruction** 2.0

  Feedback
  correct

- ( ) A computer vision algorithm for object detection 0.0

  Feedback
  wrong

- ( ) A deep learning framework for image classification 0.0

- ( ) A MLP based approach to model 3D point clouds 0.0

Exam: Quadratic Layer #15880557

15 pts · Last updated 11 Apr, 2023, 14:07 · Saved in Deep Learning questions

TEXT BLOCK

For some input features $\mathbf{x} \in \mathbb{R}^M$ and output $\mathbf{y} \in \mathbb{R}^N$ let a quadratic layer be defined by:
$y_i = \lambda \mathbf{x}^\top \mathbf{A}^{(i)} \mathbf{x} + \mathbf{x}^\top \mathbf{b}^{(i)} + c^{(i)}$

Where the parameters are given by a symmetric square matrix $\mathbf{A}^{(i)} \in \mathbb{R}^{M \times M}$, a vector $\mathbf{b}^{(i)} \in \mathbb{R}^M$,

Hint: The number of degrees of freedom (i.e. unique numbers) in a symmetric square matrix is dete

2 PTS · MULTIPLE CHOICE

What type of module is obtained when $\lambda = 0$?

○ Convolutional layer	0.0
◉ **BatchNorm layer**	2.0
◉ **Linear activation**	2.0
○ Max pooling	0.0

2 PTS · MULTIPLE CHOICE

In the context of neural networks, how can the role of the scalar $c^{(i)}$ best be described?

◉ **Bias term**	2.0
○ Stride	0.0
○ Activation function	0.0
○ Ensures output stays large (in order to avoid small gradients)	0.0

2 PTS · MULTIPLE CHOICE

Why do the square matrices $\mathbf{A}^{(i)}$ have to be symmetric?

○ For the quadratic term to be well-defined (as this term is a scalar and it must equal its transpose).	0.0
◉ **To make the matrix unique (otherwise, during optimization, the parameters can get arbitrarily large).**	2.0
○ The matrix must have real eigenvalues, or it will crash the code (which is not always optimized for complex numbers).	0.0
○ None of the above	0.0

TEXT BLOCK

We wish to compare the number of trainable parameters in this quadratic layer to the traditional lir

3 PTS · MULTIPLE CHOICE

What is $\frac{P_Q}{P_L}$, the ratio of the number of parameters in the quadratic layer to that of the linear layer?

- ○ M^2N 0.0
- ○ N 0.0
- ◉ $\frac{1}{2}M + 1$ 3.0

  Feedback

  correct. the number of parameters in a square symmetric MxM matrix is 0.5*M(M+1) (ie the number of elements e.g. the upper-triangular part including the diagonal). A linear layer (for output dimension i) would be M+1.
  So in total:
  $P_Q = 0.5 * M(M+1) + M + 1$ and $P_L = M + 1$
- ○ M 0.0

TEXT BLOCK

A model consisting of a single quadratic layer together with a softmax output layer and categorical

3 PTS · OPEN · 1/8 PAGE

How many parameters does this model have?

exactly correct	100% (3 pts)
almost correct	66.67% (2 pts)
wrong	0% (0 pts)

1 PT · MULTIPLE CHOICE

Can a deep neural network consisting of quadratic layers learn to classify MNIST well?

- ◉ Yes 1.0
- ○ No 0.0
- ○ It depends on whether there are activation functions 0.0

2 PTS · MULTIPLE CHOICE

Below are the equations required during backpropagation for $\lambda = 1$:

$$\frac{\partial L}{\partial x_i} = \sum_{j,m} \frac{\partial L}{\partial y_j} \left(2A_{im}^{(j)} x_m + b_i^{(j)}\right)$$

$$\frac{\partial L}{\partial A_{ij}^{(k)}} = \Omega$$

$$\frac{\partial L}{\partial b_j^{(i)}} = \frac{\partial L}{\partial y_i} x_j$$

$$\frac{\partial L}{\partial c^{(i)}} = \frac{\partial L}{\partial y_i}$$

What is the unknown Ω in the equations above?

◉	$\boldsymbol{\frac{\partial L}{\partial y_k} x_i x_j}$	2.0
○	$\frac{\partial L}{\partial y_k} x_i^2$	-2.0
○	$\frac{\partial L}{\partial y_j} x_k x_i$	-2.0
○	$\frac{\partial L}{\partial y_k}$	-2.0

Exam: Graph Neural Networks #15880554

10 pts · Last updated 9 Mar, 2023, 11:54 · Saved in Deep Learning questions

2 PTS · OPEN · 1/5 PAGE

Which property should an aggregation function for graph neural networks have?

An aggregation function nee... 2 pts

An aggregation function needs to be **permutation invariant**: the order of the inputs does not change the output.

4 PTS · OPEN · 1/2 PAGE

Give an example of two graphs $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$, and initial node embeddings H_1 and H_2, such that for two nodes $v_1 \in V_1$ and $v_2 \in V_2$ with the same initial node embeddings $h_{v_1}^{(0)} = h_{v_2}^{(0)}$ the updated features $h_{v_1}^{(1)}$, $h_{v_2}^{(1)}$ are the same if we aggregate over their neighbors with:

1. *mean* and *sum*, but not the same if we aggregate with *max*.
2. *max* and *mean*, but not the same if we aggregate with *sum*.

1. In 1D, take the neighbor... 2 pts

1. In 1D, take the neighbors as -1, 1, and 0, 0 or as another example as -1, 1 and -1, -1, 1, 1. So, some variance, such that max deviates, but mean and sum add up to cancel out to zero (or any value).

2. In 1D, neighbors as e.g.... 2 pts

2. In 1D, neighbors as e.g. 1, 1 and 1, 1, 1. Namely, all neighbors have the same value, such that max and mean are the same, but a varying number of neighbors, such that the sum adds up differently.

4 PTS · OPEN · 2/5 PAGE

What is the connection between the Transformer Architecture and Graph Neural Networks? Talk about the adjacency, the GNN type and the keys and query matrices and the implied invariance/symmetry.

The transformer assumes a f... 1.25 pts

The transformer assumes a fully connected network, i.e., an adjacency matrix of all ones.

The attentional GNN (rather... 1.25 pts

The attentional GNN (rather than spectral or graph convolutional) is needed (or, with extra specifications, the Message Passing NN).

While the transformer uses ... 1.25 pts

While the transformer uses separate key, query, and value embedding matrices, the graph attention networks use the same embeddings for query and key.

Both can/do use multiple he... 0.5 pts

Both can/do use multiple heads for attention.

Both are permutation invari... 1.25 pts

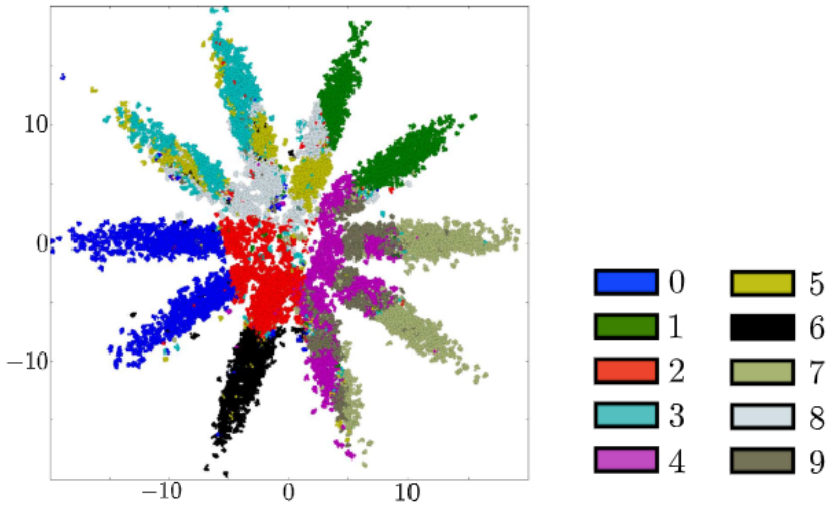
Both are permutation invariant. Ordering does not matter. Symmetry with regards to the permutation operator.

Exam: AAE #15880688

9 pts · Last updated 14 Feb, 2023, 13:39 · Saved in Deep Learning questions

TEXT BLOCK

For adversarial AutoEncoders (AAE), we are not limited to plain Gaussian priors. For instance, the l



1 PT · OPEN · 1/5 PAGE

Why might there be an overlap between regions that reconstruct different digits within a single Gaussian?

Because in the absence of l... 1 pt

Because in the absence of label information, the network might find it easier to dedicate different parts of feature space to similar digits like 3 and 8. On the other hand, for more sophisticated numbers like 7, it can use different modes for different subcategories of 7.

TEXT BLOCK

When label information is available, this can help structure the latent space of AAEs. To this end, l

1 PT · OPEN · 1/5 PAGE

Why might dedicating one mode to each different digit be be more advantageous for rare categories during training time?

Because when we force one m... 1 pt

Because when we force one mode for each category, we prevent the model from focusing only on recognizing the more frequent categories by considering different subcategories. Meanwhile, we also prevent the model from merging two rare categories.

3 PTS · OPEN · 2/5 PAGE

How can label information be used in the training of an AAE?

To incorporate label inform... 3 pts

To incorporate label information in the hidden code, we can add a one-hot vector to the input of the discriminator to make the model associate each distribution mode with different categories. For now, we only add the label information to the code for the discriminator. Thus, during adversarial training, for fake samples, we provide the number of Gaussian components that the sample is drawn from to the discriminator, while for real samples we provide the true label to the discriminator.

2 PTS · OPEN · 3/10 PAGE

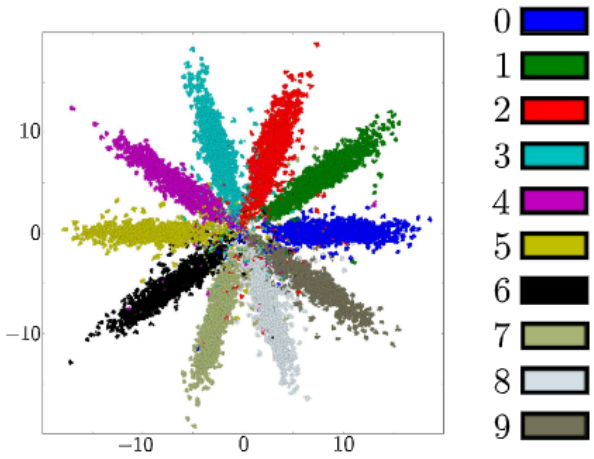
How can you train the AAE to benefit both from labeled and unlabelled data?

By adding an extra class fo... 2 pts

By adding an extra class for unlabelled data, every time we encounter an unlabelled sample, we can turn on this extra class. Meanwhile, for training the discriminator, we can sample the unlabelled examples from the whole mixture of Gaussian.

TEXT BLOCK

After training the network by incorporating label information for a prior distribution of a mixture of



2 PTS · OPEN · 1/2 PAGE

Explain how the mode allocation is different from unlabelled AAE.

As we can see from the diag... 2 pts

As we can see from the diagram, each mode has been associated with a different color this time. In other words, adding the label information aid the encoder in encoding the images from different classes to different modes.

Exam: Classification metrics #15880689

6 pts · Last updated 14 Feb, 2023, 13:43 · Saved in Deep Learning questions

TEXT BLOCK

Recall and accuracy are two different metrics used to evaluate the performance of a model. Recall i

2 PTS · OPEN · 1/5 PAGE

There are several common loss functions that are used for training a classification neural network. Very often we use the cross-entropy loss. Even in use cases where we care about having a high recall, we still use the cross-entropy loss, and not recall as the optimization metric. Why is this the case?

Recall is not a differentia... 2 pts

Recall is not a differentiable function. This means that we cannot calculate the gradients required for training our network.

2 PTS · OPEN · 1/5 PAGE

Suppose you have N classes, but you care about the k-th class having a high recall. How would you modify the cross-entropy loss to potentially improve the recall of the k-th class? Write down explicitly the modified version of the loss function, assuming you have a set of weights w_j for every class j. What are the appropriate values of the weight w_k? Note that weights w_j are not trainable parameters, but rather scalars used to modify the loss function.

Hint: make use of the weighted sum.

The modified version of the... 1 pt

The modified version of the loss should have a similar form to $L = \sum_i w_i \cdot t_i \log p_i$, where t_i is equal to 1 if the true class label is i, and 0 otherwise.

Weight $$w_k$$ should be la... 1 pt

Weight w_k should be larger than other weights to give more penalty for the k-th class.

2 PTS · OPEN · 3/10 PAGE

Now consider the following limit where the ratio $w_k/w_j \to \infty, \forall j \neq k$. In this limit, what is the final accuracy of the model? What is the final recall of this model? Express your answer in terms of the number of data points belonging to each class (for example, let N_j denote the number of data points belonging to the j-th class).

The model will most likely ... 1 pt

The model will most likely predict all datapoints to belong to the k-th class, in order to minimize the loss.

If only the $$k$$-th class ... 1 pt

If only the k-th class is correctly classified, then the final accuracy of the model is:

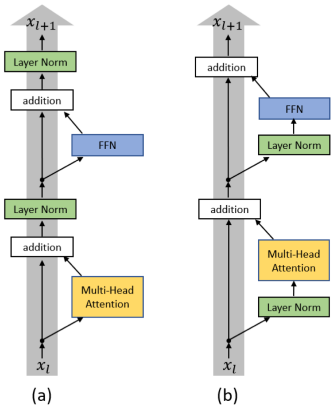
accuracy$= \frac{N_k}{\sum_i N_i}$

Exam: Transformer/Gradients #15880693

13 pts · Last updated 14 Feb, 2023, 13:39 · Saved in Deep Learning questions

TEXT BLOCK

In this question, we will take a closer look at the Transformer architecture, specifically its interactio



(figure credit - Xiong et al., 2020; you don't need to know the paper for this question)

2 PTS · OPEN · 1/4 PAGE

For the original Transformer architecture (a), consider an input $x_l \in \mathbb{R}^{B\times T\times d_x}$ with batch size B, T time steps, and d_x features. Assuming that each input element is sampled from a standard Gaussian, i.e. $[x_l]_{ijk} \sim \mathcal{N}(0,1)$, which distribution do we expect the elements/ features of x_{l+1} to follow?

$$x_{l+1}\sim\mathcal{N}(0,... 2 pts

$x_{l+1} \sim \mathcal{N}(0,1)$ - comes directly out of LayerNorm. Or at least sigma=1, mean=0

3 PTS · OPEN · 2/5 PAGE

Now consider the alternative architecture in (b). Which distributions do we expect x_{l+1} to follow for this architecture?

You can take the following assumptions:

- The FFN and MHA block are both initialized such that the output features have the same standard deviation as the inputs that came into the FFN and MHA block respectively
- The outputs of the FFN and MHA are statistically independent of their inputs

$$x_{l+1}\sim \mathcal{N}(0... 4 pts

$x_{l+1} \sim \mathcal{N}(0,3)$ - consider the intermediate LayerNorms

4 PTS · OPEN · 2/5 PAGE

Besides the feature variance, we are also interested in the scale of the gradients we obtain for each feature vector. Consider that the gradients of x_{l+1}, i.e. g_{l+1}, are sampled from $[g_{l+1}]_{ijk} \sim \mathcal{N}(0,1)$, and the original features $x_l \sim \mathcal{N}(0,1)$. Which distribution do we expect the gradients of x_l to follow for the original Transformer architecture (a)?

You can take the following assumptions:

- The gradients through the normalization statistics μ and σ in the LayerNorm can be neglected/considered to be zero.
- The FFN and MHA have been designed such that the gradients with respect to their inputs follow the same distribution (same mean and std) as the gradients with respect to their outputs.
- The gradients with respect to the inputs of the FFN and MHA block are statistically independent of the gradients of their outputs.

$$g_l\sim N(0,1)$$ - LayerN... 4 pts

$g_l \sim N(0,1)$ - LayerNorms divide by 2, but blocks multiply by 2

4 PTS · OPEN · 2/5 PAGE

Repeat the same analysis of the previous subquestion for the architecture in (b). Take the same assumption to answer what the gradient distribution of x_l is.

1 pt

Exam: MLP-code #15880695

9 pts · Last updated 31 Mar, 2023, 15:03 · Saved in Deep Learning questions

TEXT BLOCK

An assignment similar to our assignment 1, was due, yet Erin had little time to work on it. Charlie, t

```python
class MyMLP(nn.Module):

    def __init__(self, input_size=3072, output_size=10):
        super().__init__()
        self.layers = nn.Sequential(
            nn.Linear(input_size, 512),
            nn.Tanh(),
            nn.Dropout(0.1),
            nn.Linear(512, 256),
            nn.Tanh(),
            nn.Dropout(0.1),
            nn.Linear(256, 10),
            nn.Tanh()
        )
        self.softmax = nn.LogSoftmax(dim=-1)

        self._reset_parameters()

    def _reset_parameters(self):
        for n, p in self.named_parameters():
            if n.endswith("bias"):
                p.data.fill_(0)
            else:
                p.data.normal_(std=0.01)

    def forward(self, imgs):
        imgs = imgs.flatten(start_dim=1)
        logits = self.layers(imgs)
        logits = self.softmax(logits)
        return logits
```

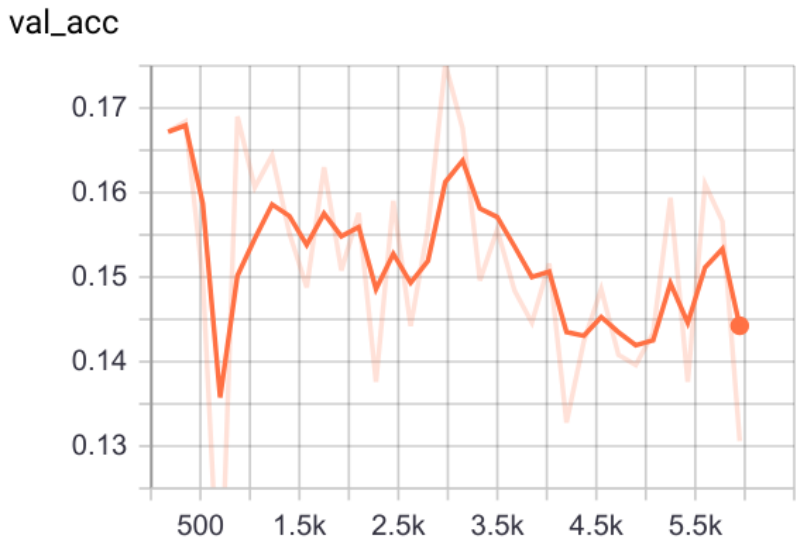
```python
model = MyMLP()
optimizer = torch.optim.Adam(model.parameters(), lr=0.1, betas=(0.9,0.999))
```

TEXT BLOCK

The validation accuracy over training iterations is plotted below.

IMAGE BLOCK 1/5 PAGE

Engineering_Validation_Performance.png



Validation performance over training iterations

4 PTS · OPEN · 1/2 PAGE

Looking at the code above, name at least 2 reasons why the model might be underperforming. Shortly explain why, and propose a solution to solve it.
Note: You can assume that the training and test loop is coded without an error, and uses the NLL loss function. The mistakes we are looking for are not specific to PyTorch. For example, are all hyperparameters in a reasonable range?

Expected answer length: two sentences for each reason.

Initialization with constan...	2 pts
Initialization with constant init. Better: kaiming/Xavier	
Tanh activation function be...	2 pts
Tanh activation function before softmax, stops the model from achieving “100%” accuracies.	
Adam uses a very high learn...	2 pts
Adam uses a very high learning rate (0.1). Common lr for Adam are 1e-3 or lower (sometimes 1e-2 but only for very tiny models, not CIFAR10).	

5 PTS · OPEN · 1/2 PAGE

Erin and Charlie discuss the residual connections in ResNet architectures. Charlie: “In most networks I have seen, the residual connection always adds the input to the output of a layer block, i.e. $x_{l+1} = x_l + F(x_l)$. But couldn’t we subtract as well instead of summing, i.e. $x_{l+1} = x_l - F(x_l)$?“
Comment on Charlie's suggestion. Are we able to replace the summation with subtraction in a standard ResNet with ReLU activations on the residual connection, and train it to a similar performance than the standard ResNet? Explain your answer.

Expected answer length: 3-4 sentences.

Wrong answer: By subtractin...	0 pts
Wrong answer: By subtracting instead of adding we amplify the vanishing gradient problem!	
It does not change the resu...	5 pts
It does not change the result if instead of adding we subtract!	

Exam: VAE - ELBO and β-VAE #15880696

15 pts · Last updated 2 Mar, 2023, 15:04 · Saved in Deep Learning questions

TEXT BLOCK

We have seen that when training a VAE we optimize the ELBO:
$\mathcal{L}_{ELBO} = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x} \mid \mathbf{z})\right] - KL\left(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})\right)$

3 PTS · OPEN · 1/2 PAGE

As seen above, $\mathcal{L}_{ELBO}$ consists out of two terms. What are these two terms typically called?

Left term: Reconstruction l... 3 pts

Left term: Reconstruction loss
Right term: Regularization loss

4 PTS · OPEN · 1/2 PAGE

Briefly explain why the names of the two terms (max 5 sentences per term) provided in part a are appropriate.

In the reconstruction loss ... 2 pts

In the reconstruction loss (left term) our data $\mathbf{x}$ is first encoded into $\mathbf{z}$ and then decoded to a distribution over the input variable x, the log likelihood then tells us how well this distribution matches x and thus how well samples from this distribution would be similar to the input. In other words, how well is the input sample $\mathbf{x}_n$ reconstructed.

In the regularization loss,... 2 pts

In the regularization loss, we minimize the difference between the distribution $q\left(z\right)$ and a prior $p\left(z\right)$, such that - on average - the distributions over encodings match the prior. Thus, we restrict the space of the latent variables. As these are just per-sample parameters, and restricting the values a parameter can take is called regularization, we can think of this term as a regularization term.

TEXT BLOCK

Another variation of VAEs is the β-VAE in which the Kullback-Leibler divergence term is weighted
$\mathcal{L}_\beta = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x} \mid \mathbf{z})\right] - \beta KL\left(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})\right)$
Note that when $\beta = 1$ is it is identical to the original VAE formulation.

4 PTS · OPEN · 1/2 PAGE

Given that we choose the following prior: $p\left(\mathbf{z}\right) = \mathcal{N}\left(0, I_D\right)$
Briefly explain (max. 5 sentences) what kind of effect a high value of β would have on the trained model?

A high value of $$\beta$$ p... 2 pts

A high value of β places considerable emphasis on the KL term in the ELBO. This applies a stronger constraint on the encoding model to be similar to the prior!

In this case, forces the la... 2 pts

In this case, forces the latent representation to be conditionally independent or disentangled.

4 PTS · FREE FORMATTED

Assume $p(x)$ is a data distribution which we wish to approximate with a parameterized distribution $q_\theta(x)$. Since the Kullback-Leibler divergence is asymmetric in its input distributions, the there are two possible expressions: $KL(p(x)||q_\theta(x))$ and $KL(q_\theta(x)||p(x))$.

Using the definition of the KL divergence, explain which of the above is referred to as mode-seeking and which as mean-seeking.

Mode-seeking the latter!	2 pts
Mode-seeking the latter!	
Mean seeking the former!	2 pts
Mean seeking the former!	

Exam: Optimization #18661868

15 pts · Last updated 14 Feb, 2023, 13:49 · Saved in Deep Learning questions

TEXT BLOCK

Convergence Rate is defined as $max\frac{||X_{t+1}-X^*||}{||X_t-X^*||}$ in which X^* is the optimization solution.

3 PTS · OPEN · 3/10 PAGE

For general quadratic form, which is $f(x)=\frac{1}{2}\,x^TAx$ calculate gradient decent for one step (learning rate is η).

$$x_{t+1}=(I-\eta A)x_t$$ 3 pts

$x_{t+1}=(I-\eta A)x_t$

5 PTS · OPEN · 1/2 PAGE

Using the value you find in the previous section, calculate the convergence rate for $f(x)$.

Hint: You can use $A=\sum_{i=1}^{n}\lambda_i u_i u_i^T$ in which λ_is are eigenvalues of matrix A and u_is are its eigenvectors

$$\text{Convergence Rate}=m... 5 pts

Convergence Rate$=max_x\frac{||(I-\eta A)x||}{||x||}=max_x\frac{||(I-\eta\sum_{i=1}^{n}\lambda_i u_i u_i^T)x||}{||x||}=max_x\frac{||\sum_{i=1}^{n}(1-\eta\lambda_i)u_i u_i^T x||}{||\sum_{i=1}^{n}u_i u_i^T x||}=max_i|1-\eta\lambda_i|=ma$

3 PTS · OPEN · 3/10 PAGE

What is the optimal learning rate for optimizing convergence rate (in terms of eigenvalues of the matrix A)?

We simply minimize the rate... 3 pts

We simply minimize the rate that we have in the previous question; it is obvious that the optimal rate happens when:

$1-\eta\lambda_{min}=\eta\lambda_{max}-1\rightarrow\eta=\frac{2}{\lambda_{min}+\lambda_{max}}$

2 PTS · OPEN · 1/5 PAGE

We define the condition number for matrix A as $\kappa=\frac{\lambda_{max}}{\lambda_{min}}$, what is the optimized convergence rate of function f in terms of κ?

If we input the optimal lea... 2 pts

If we input the optimal learning rate from previous part into convergence rate, we will have:

Optimal Convergence Rate$=1-\frac{2}{\lambda_{min}+\lambda_{max}}\,\lambda_{min}=\frac{\lambda_{max}-\lambda_{min}}{\lambda_{max}+\lambda_{min}}=\frac{\kappa-1}{\kappa+1}$

Obviously same results if we consider

Optimal Convergence Rate$=\eta\lambda_{max}-1$

2 PTS · OPEN · 1/5 PAGE

It can be proven that if we use momentum instead of gradient descent, the optimized convergence rate for the quadratic function will be $\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}$.

For $\kappa = 4$, if the number of required steps for gradient descent to reach an error of δ is n, calculate the number of steps required for Momentum to reach the same error.

$${{{1\over 3}}}^{steps}=... 2 pts

$(\frac{1}{3}^{steps}) = (\frac{3}{5})^n \rightarrow steps = n(\frac{\log 5}{\log 3} - 1) \approx 0.46n (31 steps) = (53)n \rightarrow steps = n(log3log5 - 1) \approx 0.46n$