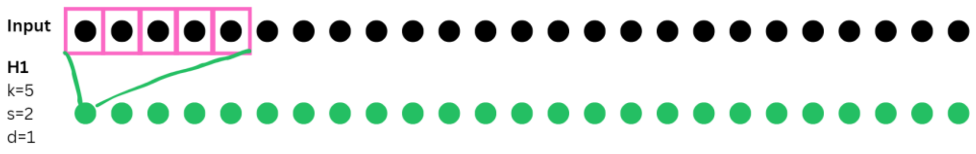
Apr 14, 2024

AI6126 – Advanced Computer Vision Assignment # 2

Reinelle Jan Bugnot
G2304329L

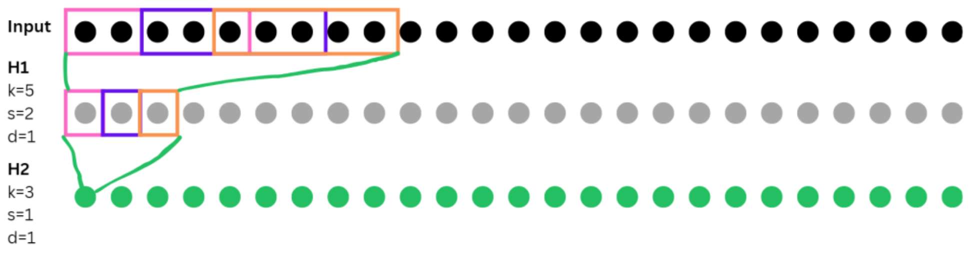
Question 1: Consider a convolutional network with 1D input x. The first hidden layer H_1 is computed using a convolution with kernel size five, stride two, and dilation one. The second hidden layer H_2 is computed using a convolution with kernel size three, stride one, and dilation one. The third hidden layer H_3 is computed using a convolution with kernel size five, stride one, and dilation two. What are the receptive field sizes at each hidden layer?

H1: The receptive field of hidden layer H1 is equal to the kernel size, **5.**



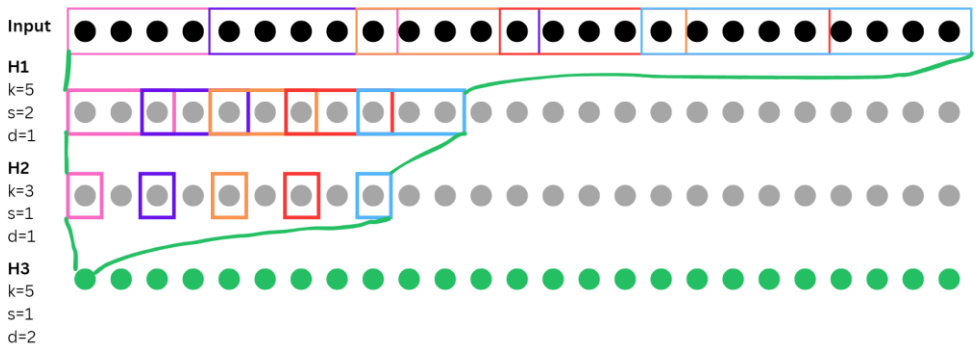
H2: The receptive field of hidden layer H2 is:

$$RF_{H_1} + (K_2 - 1)\, S_1 = 5 + (3 - 1) \times 2 = \mathbf{9}$$



H3: The receptive field of hidden layer H3 is:

$$RF_{H_2} + (K_3 - 1) \prod_{j=1}^{2} S_j\, D_3 = 9 + (5 - 1)(2 \times 1)(2) = \mathbf{25}$$



Question 2: Segmentation-related questions.

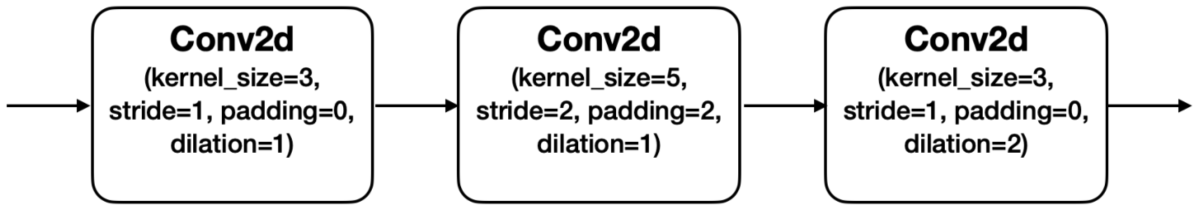
i) What is the motivation for using downsampling and upsampling in the Fully Convolutional Network for semantic segmentation?

Downsampling and upsampling are motivated by the need to balance between **contextual understanding** and **localization**. The first half of the FCN model for semantic segmentation involves downsampling, where the spatial resolution of the image is reduced while developing complex feature mappings—a.k.a, encoding. This helps the network extract and interpret context from the images.

Afterwards, the model performs upsampling which increases the spatial resolution of the feature maps—a.k.a. decoding. This upsampling process allows for localization, enabling the network to make precise, pixel-level predictions.

On top of these, FCNs for semantic segmentation also employs skip connections in order to recover the fine-grained spatial information lost in the downsampling process. These skip connections combine semantic information from a deep, course layer, with appearance information from shallow, fine layers, to produce accurate segmentations.

ii) Given the following network, calculate the receptive field.


Conv2d
(kernel_size=3, stride=1, padding=0, dilation=1)
Conv2d
(kernel_size=5, stride=2, padding=2, dilation=1)
Conv2d
(kernel_size=3, stride=1, padding=0, dilation=2)

Assuming we're calculating for the receptive field of each layer with respect to the layer before the first Conv2d (left):

Layer 1: receptive field is equal to kernel size, **3**

Layer 2: receptive field is calculated as:

$$RF_{L1} + (K_2 - 1)\, S_1 = 3 + (5 - 1) \times 1 \times 1 = \mathbf{7}$$

Layer 3: receptive field is calculated as:

$$RF_{L2} + (K_3 - 1) \prod_{j=1}^{2} S_j\, D_3 = 7 + (3 - 1)(2 \times 1)(2) = \mathbf{15}$$

iii) What are the advantages of dilated convolution over standard convolution for the semantic segmentation task?

The primary benefits of dilated convolution for semantic segmentation tasks is it **increases the size of the receptive field** to incorporate context **without reducing the resolution of the input**. A larger receptive field is especially advantageous for semantic segmentation tasks as it allows deeper layers to incorporate more contextual information. Meanwhile, preserving the original resolution of the input is imperative for segmentation tasks since we are assigning a class label to every pixel; which means the resolutions of our output and input images must match.

iv) Given a transposed convolution kernel as follows, whose stride=1, padding=0, dilation=1,

$$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 5 \end{bmatrix}$$

and input,

$$\begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}$$

what's the output matrix after the transposed convolution?
(Hint: the output should be a 4x4 matrix.)

Answer:

$$\begin{bmatrix} 1 & 2 & 3 & 0 \\ 2 & 3 & 4 & 0 \\ 3 & 4 & 5 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 2 & 4 & 6 \\ 0 & 4 & 6 & 8 \\ 0 & 6 & 8 & 10 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 \\ 0 & 2 & 3 & 4 \\ 0 & 3 & 4 & 5 \end{bmatrix} = \begin{bmatrix} 1 & 4 & 7 & 6 \\ 2 & 8 & 12 & 11 \\ 3 & 12 & 16 & 14 \\ 0 & 3 & 4 & 5 \end{bmatrix}$$

Question 3: Write out the KL divergence term in Variational Autoencoder between univariate q(z|x) and p(z), where q(z|x) = N(μ, σ2) and p(z) is a standard Gaussian N(0, 1).

The KL divergence term in a VAE between q(z|x) = N(μ, σ2) and p(z) = N(0, 1) is:

$$KL(q,p) = E_q[\log q(z) - \log p(z)]$$

Since,

$$q(z|x) = N(\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{z-\mu}{\sigma}\right)^2}$$

And,

$$p(z) = N(0,1) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}z^2}$$

Then, substituting to the KL divergence term above,

$$KL(q,p) = E_q\left[\log\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{1}{2}\left(\frac{z-\mu}{\sigma}\right)^2 - \log\left(\frac{1}{\sigma\sqrt{2\pi}}\right) + \frac{1}{2}z^2\right]$$

$$= E_q\left[\log\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \log\left(\frac{1}{\sqrt{2\pi}}\right)\right] + E_q\left[-\frac{1}{2}\left(\frac{z-\mu}{\sigma}\right)^2\right] + E_q\left[\frac{1}{2}z^2\right]$$

The three expectation terms can be simplified as follows:

The first expectation term does not include the variable z, so we can simply drop the expectation operator and simplify the logarithm.

$$E_q\left[\log\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \log\left(\frac{1}{\sigma\sqrt{2\pi}}\right)\right] = E_q\left[-\log\left(\frac{\sigma\sqrt{2\pi}}{\sqrt{2\pi}}\right)\right]$$
$$= E_q[-\log(\sigma)]$$
$$= -\log(\sigma)$$
$$= -\frac{1}{2}\log(\sigma^2)$$

For the second term, we know that $E[(x-\mu)^2]$ is the expectation equation for variance σ^2,

$$E_q\left[-\frac{1}{2}\left(\frac{z-\mu}{\sigma}\right)^2\right] = -\frac{1}{2\sigma^2}E_q[(z-\mu)^2]$$
$$= -\frac{1}{2\sigma^2}\sigma^2$$
$$= -\frac{1}{2}$$

For the third term, we know the equivalent expectation equation for variance,

$$\sigma^2 = E[x^2] - E[x]^2$$

And since $E[x] = \mu$,

$$E_q[z^2] = \sigma^2 + E_q[z]^2$$
$$= \sigma^2 + \mu^2$$
$$\frac{1}{2}E_q[z^2] = E_q\left[\frac{1}{2}z^2\right] = \frac{1}{2}(\sigma^2 + \mu^2)$$

Hence, we can represent the KL Divergence term as:

$$KL(q,p) = E_q\left[\log\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \log\left(\frac{1}{\sqrt{2\pi}}\right)\right] + E_q\left[-\frac{1}{2}\left(\frac{z-\mu}{\sigma}\right)^2\right] + E_q\left[\frac{1}{2}z^2\right]$$
$$= -\frac{1}{2}\log(\sigma^2) - \frac{1}{2} + \frac{1}{2}(\sigma^2 + \mu^2)$$
$$= \boldsymbol{\frac{1}{2}(-\log(\sigma^2) - 1 + \sigma^2 + \mu^2)}$$

Question 4: What is the key difference between VQVAE and VAE?

The key difference between VQVAE and VAE is that VQVAE models the latent space as a discrete distribution through a vector quantization layer; while VAE models the latent space as a continuous distribution by enforcing a constraint that the latent variables follow a standard normal distribution. The table below summarizes the implications of this key difference between VQVAE and VAE.

Table I. Difference between VQVAE and VAE

	VAE	**VQ-VAE**
Latent Representation	Continuous	Discrete
Reconstruction Quality	Less precise, often blurry	More precise
Interpolation	Smooth interpolation between data	Interpolation is difficult in discrete latent space
Learning	Learns a distribution of data	Learns a codebook of the data

Question 5: Briefly describe what will be generated if Mode Collapse happens in the GAN training and give one technique to alleviate the issue.

Mode collapse in Generative Adversarial Networks (GANs) occurs when the generator focuses solely on generating examples from a specific class, neglecting the diversity present in the training dataset. This limits the GAN's ability to produce a wide range of realistic samples, compromising its overall performance.

To address this issue, Wasserstein GANs (WGANs) have been proposed as an extension of traditional GANs. WGANs utilize the Wasserstein distance as the loss function instead of the traditional cross-entropy. This modification enables WGANs to offer a more stable and informative training signal, leading to smoother convergence and mitigating mode collapse.

Question 6: Briefly explain why the latent dimension is usually lower than the data dimension in Autoencoders.

If the latent dimension has the same or higher dimensionality as the input data, the autoencoder could potentially learn to just copy the input to the output without learning any meaningful representation of the data. The main purpose of autoencoders are to learn a compressed representation of the data; i.e., learn the most important features of the data that could facilitate its reconstruction. This could only happen if the latent dimension is lower that the data dimension.

References

[1] Chen-Change Loy, *Convolutional Neural Network II*, Week – 3 Lecture notes, Nanyang Technological University, Singapore.