Exercise 4.3: backpropagation through a convolutional layer

(a)

$$\nabla E_A(x) = \frac{\partial E}{\partial g} \frac{\partial g}{\partial x}$$

$$\frac{\partial E}{\partial g} = \frac{\partial}{\partial g} g(Ax) = \nabla g(Ax)$$

$$\frac{\partial g}{\partial x} = \frac{\partial}{\partial x} Ax = A$$

$$\nabla E_A(x) = \nabla g(Ax)\ A$$

(b)

The backwardpass of a convolution "restores" the input size of a layer given a (mostly) smaller output size. This upsampling process could be described as a convolution as well by adding zeros to the input of this operation. Altough a better solution is the convolution transpose operation:
Arithmetically a convolution describes a matrix multiplication. Thus the backpropagation of error derivatives through a convolutional layer can be achieved by multiplying the transpose of the matrix defined by the convolution. This operation is also needed when reconstructing visible units from hidden units in the network. The resulting restoration is often a desired effect in encoder-decoder networks constructed with convolutions.

Exercise 4.4: PCA vs. auto-encoder

$V \in \mathbb{R}^{d \times d}$, columns are eigenvectors q_i
$x \in \mathbb{R}^d$

$$a_i = q_i^T\ x,$$

a_i being the projected values in the new principal component basis projection of the vector x onto subspace spanned by the first K principal components in $\mathbb{R}^d$:

$$\hat{x} = \sum_{i=1}^{K} a_i\, q_i$$