

# Context Encoders

---

Niharika Vadlamudi 2018122008

Sravya Vardhani S 2019702008

Dhawal Sirikonda 2019201089

# Contents

**Abstract**

**Introduction**

**Context encoders for  
image generation**

**Loss function**

**Evaluation**

**Timeline**

# Abstract

Visual feature learning algorithm based on context based pixel prediction (surrounding based).

Two main parts

- The content of the image should be extracted.
- Plausible hypothesis should be provided for the missing parts.

Better results are obtained when a reconstruction loss plus an adversarial loss is used over standard pixel wise reconstruction loss.

# Introduction

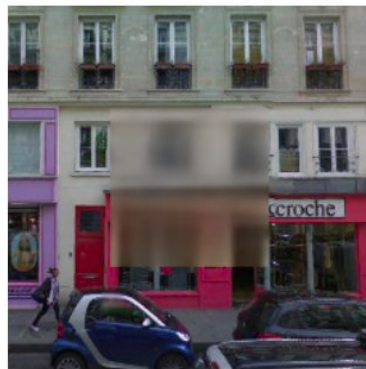
- Exploring if state-of-the-art computer vision algorithms can make a sense of the structure as humans do.
- In this paper they learn and predict the structure using convolutional neural networks.( shown below)



(a) Input context



(b) Human artist

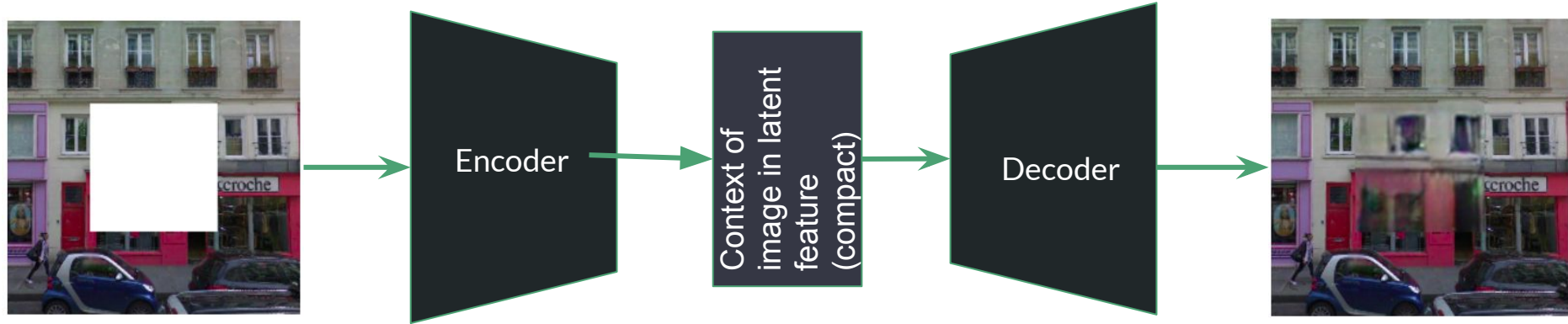


(c) Context Encoder  
( $L_2$  loss)



(d) Context Encoder  
( $L_2 + \text{Adversarial loss}$ )

# Model



In autoencoders the feature we get is likely to be just the compressed version of the image without learning any meaningful representation.

Denoising autoencoders corrupt the input image and try to undo it , but there's no need to understand the semantic meaning of the scene to do this as these corruptions occur at localized and low-level.

Here we need to fill large missing parts of image, we require deeper semantic understanding.

This task, however, is inherently multi-modal as there are:

- multiple ways to fill the missing region while.
- maintaining coherence with the given context.

This problem is decoupled in the loss function by jointly training our context encoders to minimize both a reconstruction loss and an adversarial loss.

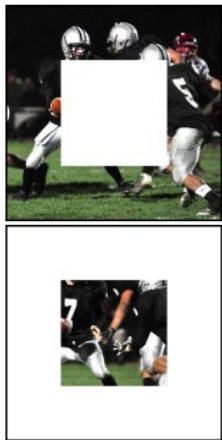
**Encoder** - Context of the image patch is extracted and nearest neighbour contexts produces patches which are semantically similar to the original patch.

**Fine tune** -they validated the quality of encoder features on variety of image understanding tasks (classification, object recognition).

**Decoder** - this method is often able to fill in realistic image content.

# Region Masks

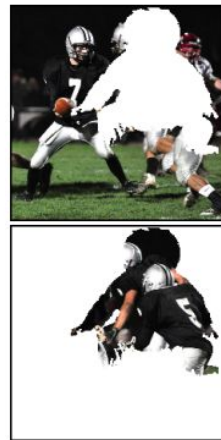
The input to a context encoder is an image with one or more of its regions “dropped out”; i.e., set to zero, assuming zero-centered inputs. The removed regions could be of any shape, we present three different strategies here:



(a) Central region

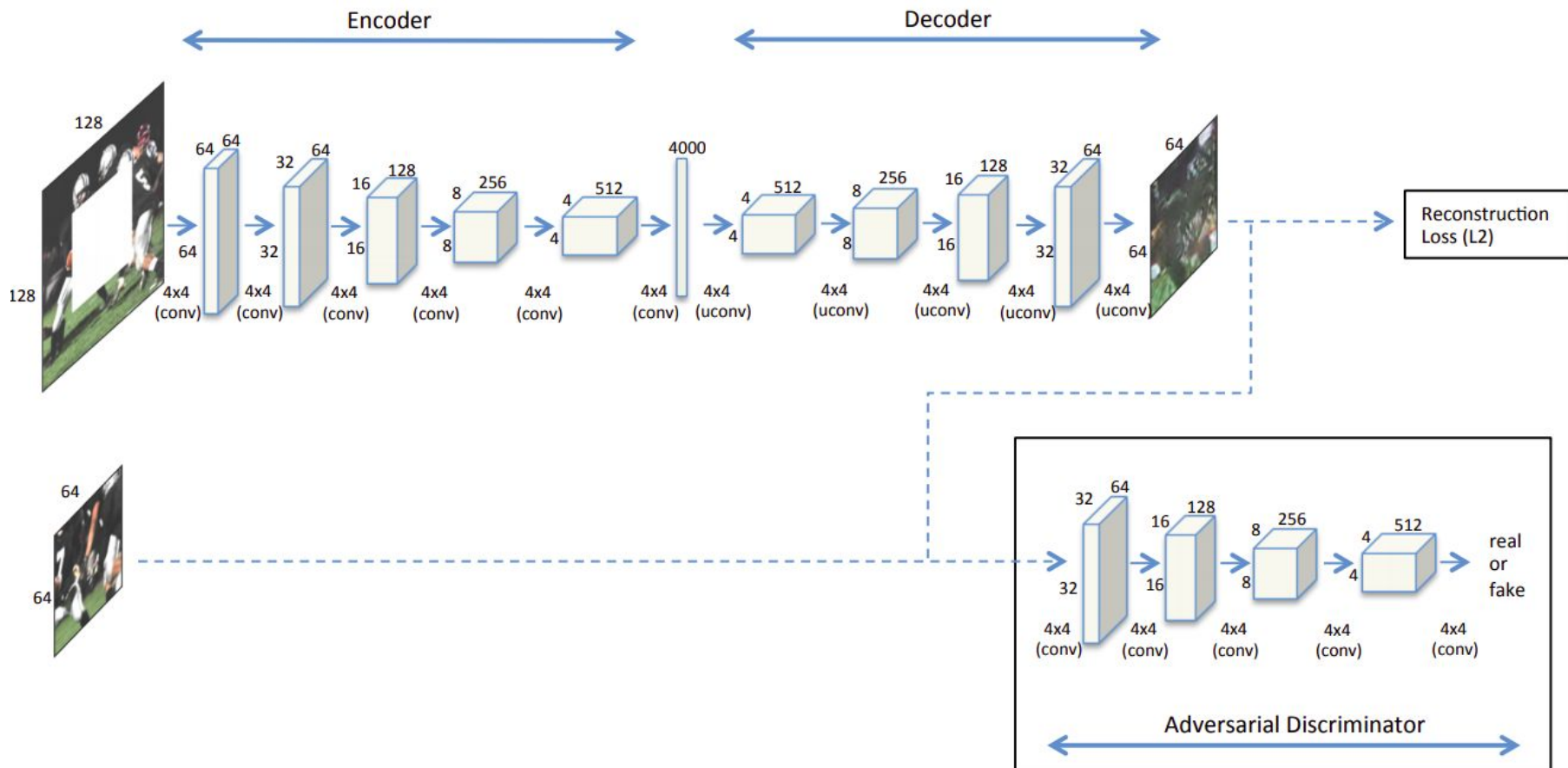


(b) Random block



(c) Random region

# Current network





# Context encoders for image generation

---

# Encoder-decoder pipeline

- Simple encoder-decoder pipeline.
- Encoder takes an input image with missing regions and produces a latent feature representation of that image
- Decoder takes this feature representation and produces the missing image content.
- Encoder and decoder should be connected with a channel wise fully connected layer, so decoder can reason with the whole image content.

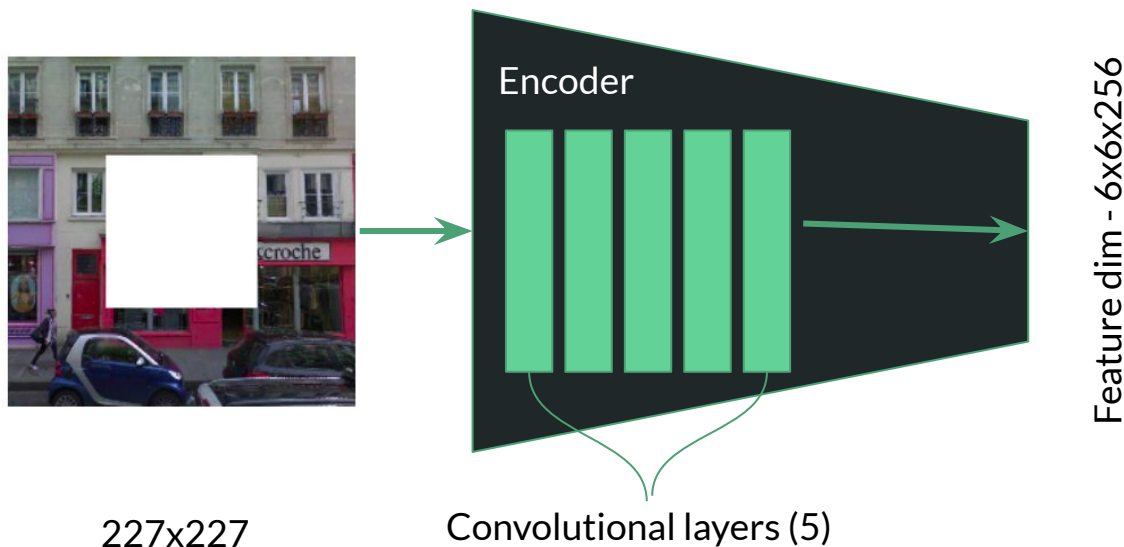
**Pool-free encoders** - They experimented with replacing all pooling layers with convolutions of the same kernel size and stride.

The overall stride of the network remains the same, but it results in finer inpainting.

Intuitively, there is no reason to use pooling for reconstruction based networks.

Original AlexNet architecture (with pooling) for all feature learning results is used.

# Encoder



Encoder is derived from the AlexNet architecture. Image of size  $227 \times 227$  ( $32 \times 32$ ), we use the first five convolutional layers and the following pooling layer to compute an abstract  $6 \times 6 \times 256$  (1024) dimensional feature representation.

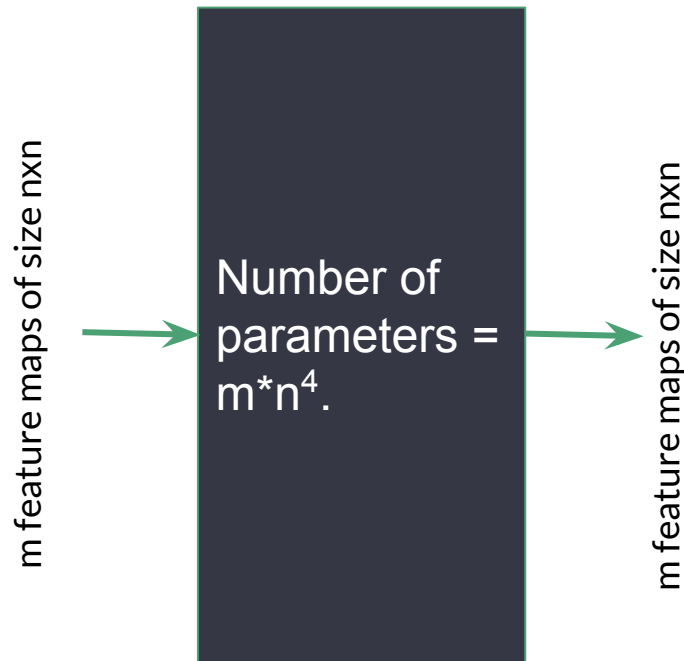
If only convolutional layers are used then all feature maps will be connected together but no connections within a specific map.

To handle this we use fully connected layer.

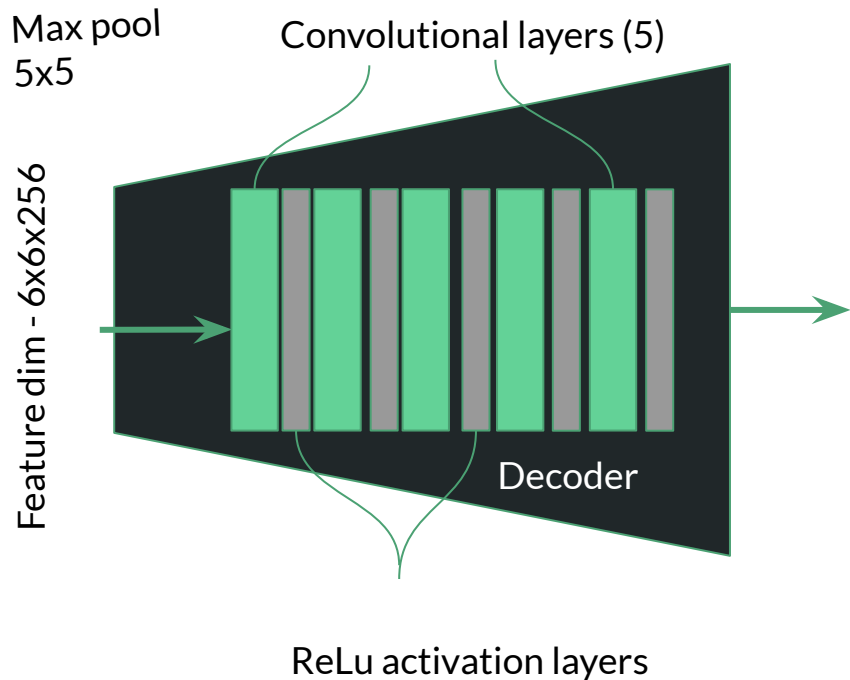
# Channel-wise fully-connected layer

This layer is essentially a fully-connected layer with groups, intended to propagate information within activations of each feature map.

However, unlike a fully-connected layer, it has no parameters connecting different feature maps and only propagates information within feature maps.



# Decoder



It can be understood as upsampling followed by convolution, or convolution with fractional stride.

The intuition behind this is straightforward **non-linear weighted upsampling** of the feature produced by the encoder until we roughly reach the original target size.

# Loss Function

---

# Loss function

**Reconstruction Loss** - Normalised L2 distance , it encourages the decoder to produce a rough outline of the predicted object, but does not indicate detail.

$$\mathcal{L}_{rec}(x) = \|\hat{M} \odot (x - F((1 - \hat{M}) \odot x))\|_2^2,$$

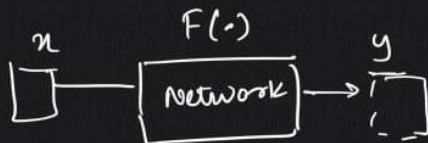
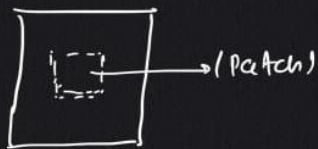
**Adversarial Loss** - Based on GANs To learn a generative model G of a data distribution and learn an adversarial discriminative model D to provide loss gradients to the generative model.

G and D are parametric functions (e.g., deep networks) where  $G : Z \rightarrow X$  maps samples from noise distribution Z to data distribution X .



•  $x = \text{Image}$

$$\hat{M} = \begin{cases} 1, & \text{pixels } \in \text{Patch} \\ 0, & \text{else} \end{cases}$$

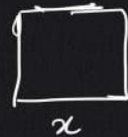


L2 loss:

Dot prod @ every pixel position.

(Start)  $\Rightarrow$

$$(1 - \hat{m}) \odot x =$$



(Patch will be 0)

Predict opt

$$\odot M \leq x - F((1 - \hat{m}) \odot x) \leftarrow \text{FC}$$

(changed pixels)

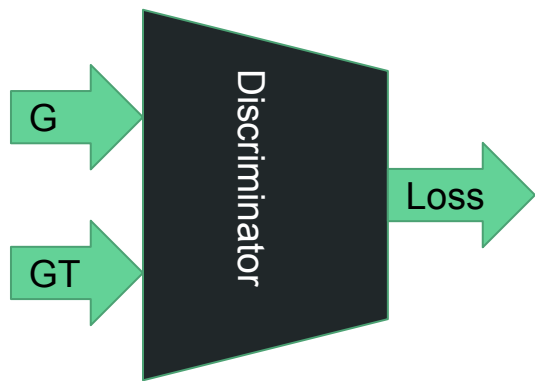


$\Uparrow$  END opt.



$\hat{y} = \text{Image Reconstructed.}$

The learning procedure is a two-player game where an adversarial discriminator  $D$  takes in both the prediction of  $G$  and ground truth samples, and tries to distinguish them, while  $G$  tries to confuse  $D$  by producing samples that appear as “real” as possible. (  $G$  - Generated image,  $GT$  - Ground Truth ).



The objective for discriminator is logistic likelihood indicating whether the input is real sample or predicted one:

$$\min_G \max_D \mathbb{E}_{x \in \mathcal{X}} [\log(D(x))] + \mathbb{E}_{z \in \mathcal{Z}} [\log(1 - D(G(z)))]$$

To customize GANs for this task, one could condition on the given context information; i.e., the mask  $\hat{M} \odot x$ .

However, conditional GANs don't train easily for context prediction task as the adversarial discriminator  $D$  easily exploits the perceptual discontinuity in generated regions and the original context to easily classify predicted versus real samples.

We thus use an alternate formulation, by conditioning only the generator (not the discriminator) on context. Modified adversarial loss is:

$$\mathcal{L}_{adv} = \max_D \mathbb{E}_{x \in \mathcal{X}} [\log(D(x)) + \log(1 - D(F((1 - \hat{M}) \odot x)))],$$

where, in practice, both  $F$  and  $D$  are optimized jointly using alternating SGD. Note that this objective encourages entire output of the context encoder to look realistic, not just the missing regions as before

# Joint loss

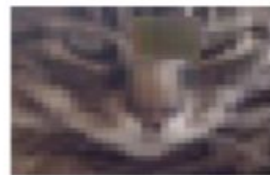
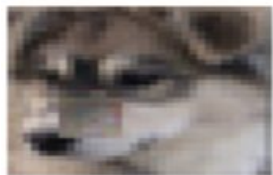
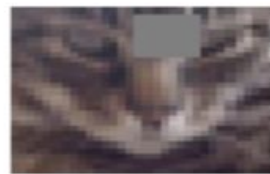
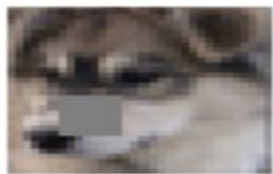
We define the overall loss function as

$$\mathcal{L} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{adv}\mathcal{L}_{adv}.$$

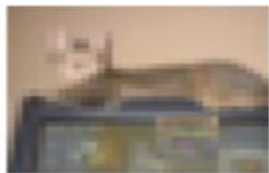
# Results

---

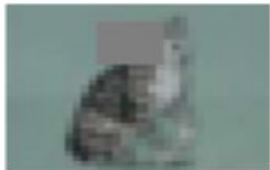
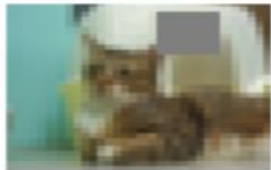
Iteration :200



Iteration :800

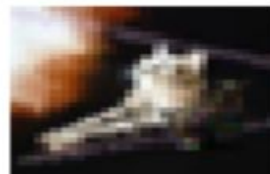
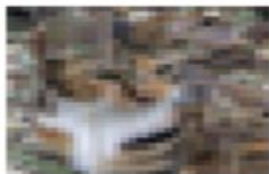
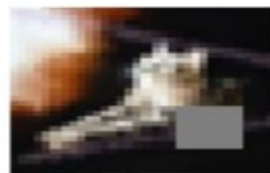
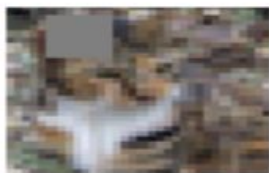
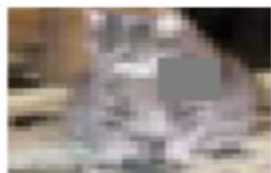


Iteration :1700

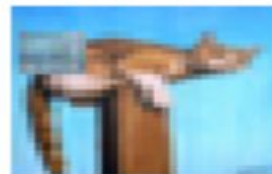
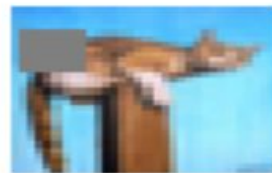
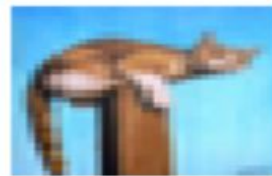




Iteration :5300



Iteration :10100



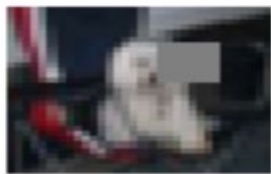
Iteration :15500



Iteration :20900

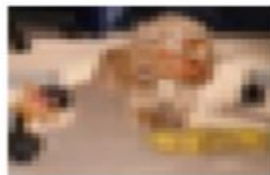
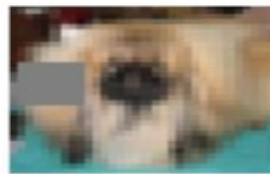
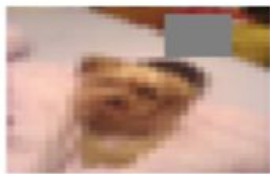


Iteration : 29300





Iteration : 29900

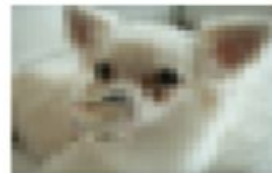
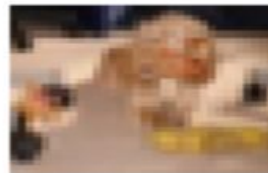
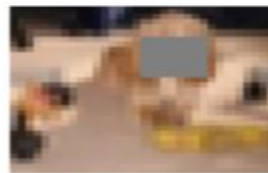
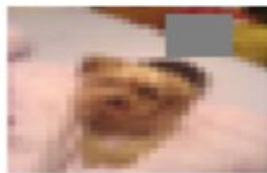


Fails !!

---

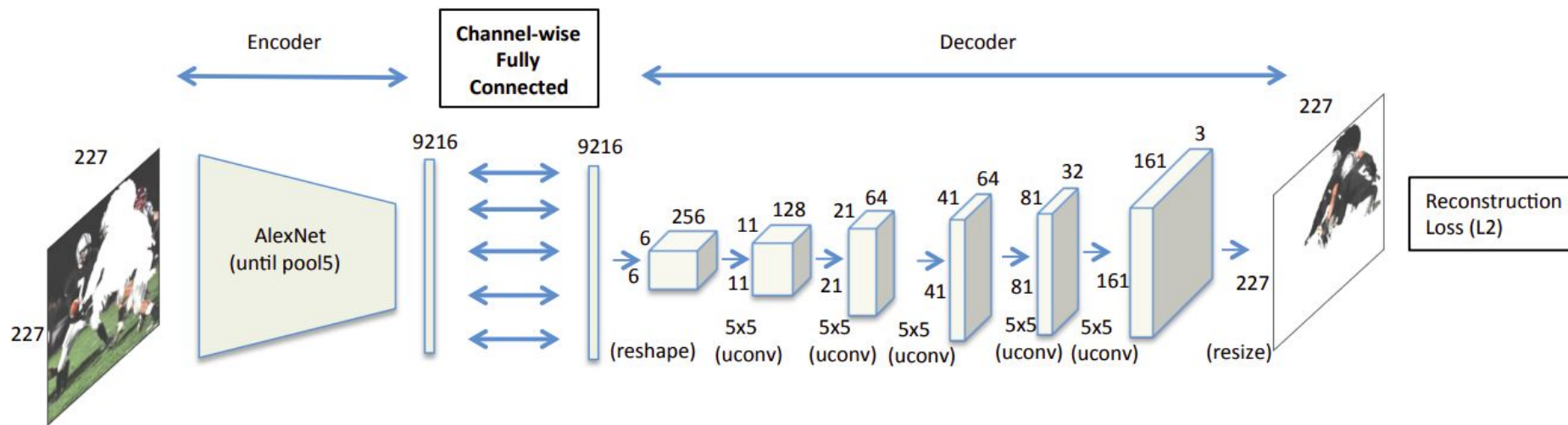
# Where the prediction fails ?

The prediction goes a little awry when the shaded block or the removed part of the image has mixed neighbours of both the animal and its background.





# Next phase



(b) Context encoder trained with reconstruction loss for feature learning by filling in *arbitrary region dropouts* in the input.

# Timeline

---

