PixelCNN Autoregressive Model

In [1]:
```
%load_ext autoreload
%autoreload 2

import warnings
warnings.filterwarnings("ignore")

import tensorflow as tf
from tensorflow.keras import (layers, models, callbacks, optimizers, datase

import numpy as np
import matplotlib.pyplot as plt
from IPython.display import DisplayObject, display, Image
```

```
2024-04-07 15:34:23.243349: I tensorflow/core/platform/cpu_feature_guard.c
c:210] This TensorFlow binary is optimized to use available CPU instructio
ns in performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebui
ld TensorFlow with the appropriate compiler flags.
2024-04-07 15:34:23.879816: W tensorflow/compiler/tf2tensorrt/utils/py_uti
ls.cc:38] TF-TRT Warning: Could not find TensorRT
```

In [2]:
```
IMAGE_SIZE=28
PIXEL_LEVELS = 4
N_FILTERS = 128
RESIDUAL_BLOCKS = 5
BATCH_SIZE = 128
EPOCHS = 25
```

In [3]:
```python
(x_train, _),(_, _) = datasets.fashion_mnist.load_data()

def preprocess(imgs_int):
    imgs_int = np.expand_dims(imgs_int, -1)
    imgs_int = tf.image.resize(imgs_int, (IMAGE_SIZE, IMAGE_SIZE)).numpy()
    imgs_int = (imgs_int / (256 / PIXEL_LEVELS)).astype(int)
    imgs = imgs_int.astype("float32")
    imgs = imgs / PIXEL_LEVELS
    return imgs, imgs_int

input_data, output_data = preprocess(x_train)
display(Image(input_data))

#plt.imshow(input_data)
```

```
2024-04-07 15:34:25.641564: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:984] could not open file to read NUMA node: /sys/bus/pci/
devices/0000:01:00.0/numa_node
Your kernel may have been built without NUMA support.
2024-04-07 15:34:26.081439: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:984] could not open file to read NUMA node: /sys/bus/pci/
devices/0000:01:00.0/numa_node
Your kernel may have been built without NUMA support.
2024-04-07 15:34:26.081489: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:984] could not open file to read NUMA node: /sys/bus/pci/
devices/0000:01:00.0/numa_node
Your kernel may have been built without NUMA support.
2024-04-07 15:34:26.106972: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:984] could not open file to read NUMA node: /sys/bus/pci/
devices/0000:01:00.0/numa_node
Your kernel may have been built without NUMA support.
2024-04-07 15:34:26.107030: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:984] could not open file to read NUMA node: /sys/bus/pci/
devices/0000:01:00.0/numa_node
Your kernel may have been built without NUMA support.
2024-04-07 15:34:26.107057: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:984] could not open file to read NUMA node: /sys/bus/pci/
devices/0000:01:00.0/numa_node
Your kernel may have been built without NUMA support.
2024-04-07 15:34:26.251771: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:984] could not open file to read NUMA node: /sys/bus/pci/
devices/0000:01:00.0/numa_node
Your kernel may have been built without NUMA support.
2024-04-07 15:34:26.251827: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:984] could not open file to read NUMA node: /sys/bus/pci/
devices/0000:01:00.0/numa_node
Your kernel may have been built without NUMA support.
2024-04-07 15:34:26.251835: I tensorflow/core/common_runtime/gpu/gpu_devic
e.cc:2019] Could not identify NUMA node of platform GPU id 0, defaulting t
o 0.  Your kernel may not have been built with NUMA support.
2024-04-07 15:34:26.251869: I external/local_xla/xla/stream_executor/cuda/
cuda_executor.cc:984] could not open file to read NUMA node: /sys/bus/pci/
devices/0000:01:00.0/numa_node
Your kernel may have been built without NUMA support.
2024-04-07 15:34:26.251889: I tensorflow/core/common_runtime/gpu/gpu_devic
e.cc:1928] Created device /job:localhost/replica:0/task:0/device:GPU:0 wit
h 2248 MB memory:  -> device: 0, name: NVIDIA GeForce GTX 1650, pci bus id
: 0000:01:00.0, compute capability: 7.5
```

```
In [4]: class MaskedConv2D(layers.Layer):
            def __init__(self, mask_type, **kwargs):
                super(MaskedConv2D, self).__init__()
                self.mask_type = mask_type
                self.conv = layers.Conv2D(**kwargs)

            def build(self, input_shape):
                self.conv.build(input_shape)
                kernel_shape = self.conv.kernel.shape
                self.mask = np.zeros(shape=kernel_shape)
                self.mask[:, kernel_shape[0] // 2, ...] = 1.0
                self.mask[kernel_shape[0] // 2, : kernel_shape[1] // 2, ...] = 1.0
                if self.mask_type == "B":
                    self.mask[kernel_shape[0] // 2, kernel_shape[1] // 2, ...] = 1.

            def call(self, inputs):
                self.conv.kernel.assign(self.conv.kernel * self.mask)
                return self.conv(inputs)

            def get_config(self):
                cfg = super().get_config()
                return cfg

        class ResidualBlock(layers.Layer):
            def __init__(self, filters, **kwargs):
                super(ResidualBlock,  self).__init__(**kwargs)
                self.conv1 = layers.Conv2D(filters=filters // 2, kernel_size=1, act
                self.pixel_conv = MaskedConv2D(mask_type = "B", filters = filters /
                self.conv2 = layers.Conv2D(filters=filters, kernel_size=1, activati

            def call(self, inputs):
                x = self.conv1(inputs)
                x = self.pixel_conv(x)
                x = self.conv2(x)
                return layers.add([inputs, x])

            def get_config(self):
                cfg = super().get_config()
                return cfg
```

In [5]:
```
inputs = layers.Input(shape=(IMAGE_SIZE, IMAGE_SIZE, 1))
x = MaskedConv2D(mask_type="A", filters=N_FILTERS, kernel_size=3, activatio

for _ in range(RESIDUAL_BLOCKS):
    x = ResidualBlock(filters=N_FILTERS)(x)

for _ in range(2):
    x = MaskedConv2D(mask_type="B", filters=N_FILTERS, kernel_size=1, strid

out = layers.Conv2D(filters = PIXEL_LEVELS, kernel_size=1, strides=1, activ

pixel_cnn = models.Model(inputs, out)
pixel_cnn.summary()
```

```
Model: "functional_1"

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━
┃ Layer (type)                       ┃ Output Shape             ┃       Param #
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━
│ input_layer (InputLayer)           │ (None, 28, 28, 1)        │             0
├────────────────────────────────────┼──────────────────────────┼───────────────
│ masked_conv2d (MaskedConv2D)       │ (None, 28, 28, 128)      │         1,280
├────────────────────────────────────┼──────────────────────────┼───────────────
│ residual_block (ResidualBlock)     │ (None, 28, 28, 128)      │        53,504
├────────────────────────────────────┼──────────────────────────┼───────────────
│ residual_block_1                   │ (None, 28, 28, 128)      │        53,504
│ (ResidualBlock)                    │                          │
├────────────────────────────────────┼──────────────────────────┼───────────────
│ residual_block_2                   │ (None, 28, 28, 128)      │        53,504
│ (ResidualBlock)                    │                          │
├────────────────────────────────────┼──────────────────────────┼───────────────
│ residual_block_3                   │ (None, 28, 28, 128)      │        53,504
│ (ResidualBlock)                    │                          │
├────────────────────────────────────┼──────────────────────────┼───────────────
│ residual_block_4                   │ (None, 28, 28, 128)      │        53,504
│ (ResidualBlock)                    │                          │
├────────────────────────────────────┼──────────────────────────┼───────────────
│ masked_conv2d_6 (MaskedConv2D)     │ (None, 28, 28, 128)      │        16,512
├────────────────────────────────────┼──────────────────────────┼───────────────
│ masked_conv2d_7 (MaskedConv2D)     │ (None, 28, 28, 128)      │        16,512
├────────────────────────────────────┼──────────────────────────┼───────────────
│ conv2d_18 (Conv2D)                 │ (None, 28, 28, 4)        │           516
└────────────────────────────────────┴──────────────────────────┴───────────────

 Total params: 302,340 (1.15 MB)

 Trainable params: 302,340 (1.15 MB)

 Non-trainable params: 0 (0.00 B)
```

In [6]:
```
adam = optimizers.Adam(learning_rate=0.0005)
loss = losses.SparseCategoricalCrossentropy(tf.keras.losses.Reduction.SUM)
pixel_cnn.compile(loss=loss, optimizer=adam, metrics=["accuracy", losses.Sp
```

In [7]:
```
tensorfboad_callback = callbacks.TensorBoard(log_dir = "./logs")
model_checkpoint_callback = callbacks.ModelCheckpoint(filepath="./checkpoin
                                                       save_weights_only=True

class ImageGenerator(callbacks.Callback):
    def __init__(self, num_img):
        self.num_img = num_img

    def sample_from(self, probs, temperature):
        probs = probs ** (1 / temperature)
        probs = probs / np.sum(probs)
        return np.random.choice(len(probs), p=probs)

    def generate(self, temperature):
        generated_images = np.zeros(shape=(self.num_img,) + (pixel_cnn.inpu
        batch, rows, cols, channels = generated_images.shape
        for row in range(rows):
            for col in range(cols):
                for channel in range(channels):
                    probs = self.model.predict(generated_images, verbose=0)
                    generated_images[:, row, col, channel] = [self.sample_f
                    generated_images[:, row, col, channel] /= PIXEL_LEVELS
        return generated_images

    def on_epoch_end(self, epoch, logs=None):
        generated_images = self.generate(temperature=1.0)
        plt.figure(figsize=(5,5))
        for i in generated_images:
            plt.imshow(i)
        #plt.savefig(f"Generated_images_{i}", format="png")
image_generator_callback = ImageGenerator(num_img=10)
```

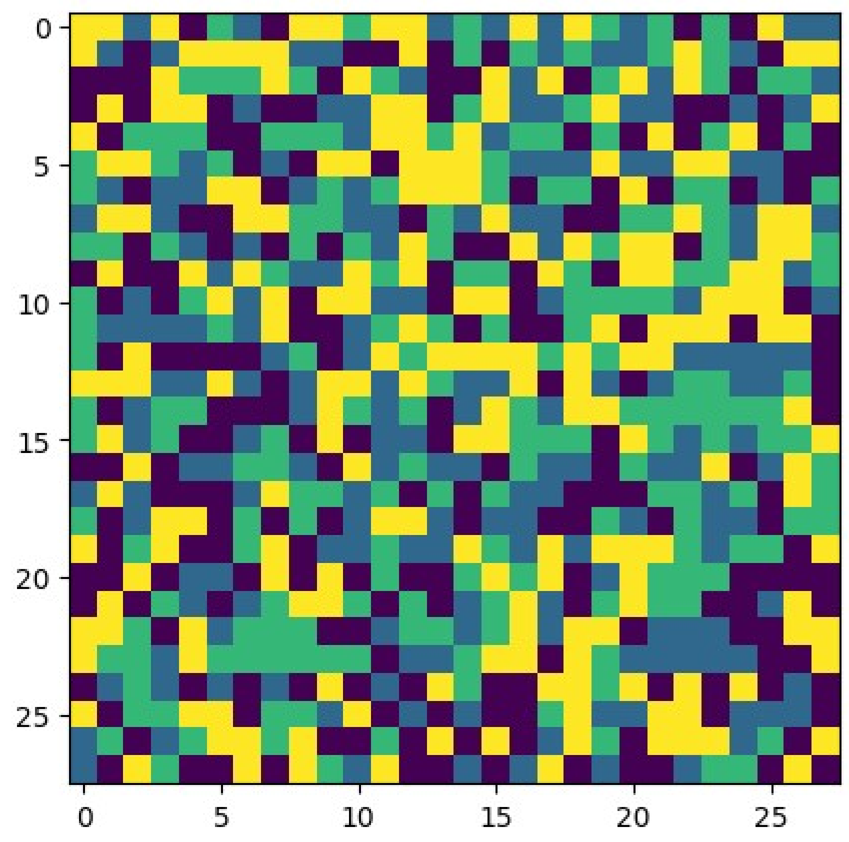
In [8]:
```python
# steps_per_epoch=1 to train faster but results are non accurate
pixel_cnn.fit(input_data, output_data, batch_size=BATCH_SIZE, epochs=EPOCHS
```

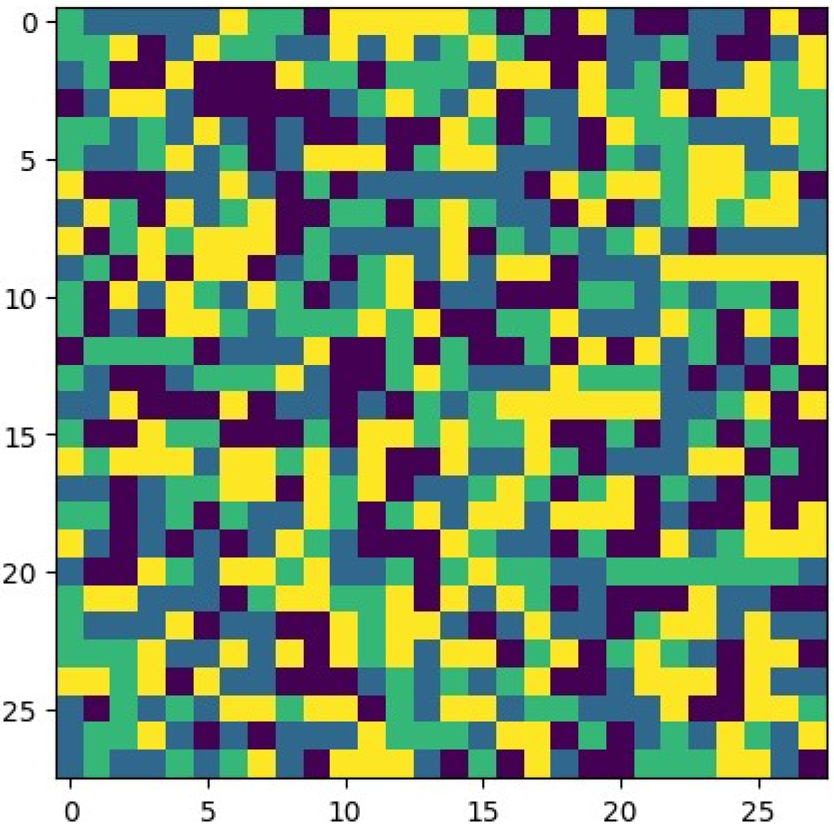
```
Epoch 1/25

WARNING: All log messages before absl::InitializeLog() is called are writt
en to STDERR
I0000 00:00:1712484273.469948  666377 service.cc:145] XLA service 0x7fa844
00ef20 initialized for platform CUDA (this does not guarantee that XLA wil
l be used). Devices:
I0000 00:00:1712484273.470016  666377 service.cc:153]   StreamExecutor dev
ice (0): NVIDIA GeForce GTX 1650, Compute Capability 7.5
2024-04-07 15:34:33.568880: I tensorflow/compiler/mlir/tensorflow/utils/du
mp_mlir_util.cc:268] disabling MLIR crash reproducer, set env var `MLIR_CR
ASH_REPRODUCER_DIRECTORY` to enable.
W0000 00:00:1712484273.750199  666377 assert_op.cc:38] Ignoring Assert ope
rator compile_loss/sparse_categorical_crossentropy/SparseSoftmaxCrossEntro
pyWithLogits/assert_equal_1/Assert/Assert
W0000 00:00:1712484273.977995  666377 assert_op.cc:38] Ignoring Assert ope
rator sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/
assert_equal_1/Assert/Assert
2024-04-07 15:34:34.231257: I external/local_xla/xla/stream_executor/cuda/
cuda_dnn.cc:465] Loaded cuDNN version 8902
I0000 00:00:1712484281.064228  666377 device_compiler.h:188] Compiled clus
ter using XLA!  This line is logged at most once for the lifetime of the p
rocess.

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 11s/step - accuracy: 0.6129 - loss: 1.3869 - s
parse_categorical_crossentropy: 1.3869

W0000 00:00:1712484281.875514  666374 assert_op.cc:38] Ignoring Assert ope
rator compile_loss/sparse_categorical_crossentropy/SparseSoftmaxCrossEntro
pyWithLogits/assert_equal_1/Assert/Assert
W0000 00:00:1712484281.883013  666374 assert_op.cc:38] Ignoring Assert ope
rator sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/
assert_equal_1/Assert/Assert
W0000 00:00:1712484283.635918  666376 assert_op.cc:38] Ignoring Assert ope
rator compile_loss/sparse_categorical_crossentropy/SparseSoftmaxCrossEntro
pyWithLogits/assert_equal_1/Assert/Assert
W0000 00:00:1712484283.639935  666376 assert_op.cc:38] Ignoring Assert ope
rator sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/
assert_equal_1/Assert/Assert

1/1 ━━━━━━━━━━━━━━━━━━━━ 56s 56s/step - accuracy: 0.6129 - loss: 1.3869 -
sparse_categorical_crossentropy: 1.3869 - val_accuracy: 0.7445 - val_loss:
1.3738 - val_sparse_categorical_crossentropy: 1.3738
Epoch 2/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 41s 41s/step - accuracy: 0.7470 - loss: 1.3736 -
sparse_categorical_crossentropy: 1.3736 - val_accuracy: 0.7725 - val_loss:
1.3602 - val_sparse_categorical_crossentropy: 1.3602
Epoch 3/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 43s 43s/step - accuracy: 0.7702 - loss: 1.3604 -
sparse_categorical_crossentropy: 1.3604 - val_accuracy: 0.7736 - val_loss:
1.3458 - val_sparse_categorical_crossentropy: 1.3458
Epoch 4/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 43s 43s/step - accuracy: 0.7761 - loss: 1.3464 -
sparse_categorical_crossentropy: 1.3464 - val_accuracy: 0.7734 - val_loss:
1.3295 - val_sparse_categorical_crossentropy: 1.3295
Epoch 5/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 43s 43s/step - accuracy: 0.7627 - loss: 1.3295 -
```

```
sparse_categorical_crossentropy: 1.3295 - val_accuracy: 0.7733 - val_loss:
1.3112 - val_sparse_categorical_crossentropy: 1.3112
Epoch 6/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 43s 43s/step - accuracy: 0.7655 - loss: 1.3128 -
sparse_categorical_crossentropy: 1.3128 - val_accuracy: 0.7733 - val_loss:
1.2906 - val_sparse_categorical_crossentropy: 1.2906
Epoch 7/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 42s 42s/step - accuracy: 0.7761 - loss: 1.2904 -
sparse_categorical_crossentropy: 1.2904 - val_accuracy: 0.7733 - val_loss:
1.2675 - val_sparse_categorical_crossentropy: 1.2675
Epoch 8/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 42s 42s/step - accuracy: 0.7892 - loss: 1.2630 -
sparse_categorical_crossentropy: 1.2630 - val_accuracy: 0.7731 - val_loss:
1.2407 - val_sparse_categorical_crossentropy: 1.2407
Epoch 9/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 43s 43s/step - accuracy: 0.7837 - loss: 1.2390 -
sparse_categorical_crossentropy: 1.2390 - val_accuracy: 0.7731 - val_loss:
1.2100 - val_sparse_categorical_crossentropy: 1.2100
Epoch 10/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 42s 42s/step - accuracy: 0.7728 - loss: 1.2095 -
sparse_categorical_crossentropy: 1.2095 - val_accuracy: 0.7731 - val_loss:
1.1747 - val_sparse_categorical_crossentropy: 1.1747
Epoch 11/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 43s 43s/step - accuracy: 0.7887 - loss: 1.1655 -
sparse_categorical_crossentropy: 1.1655 - val_accuracy: 0.7731 - val_loss:
1.1341 - val_sparse_categorical_crossentropy: 1.1341
Epoch 12/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 43s 43s/step - accuracy: 0.7890 - loss: 1.1246 -
sparse_categorical_crossentropy: 1.1246 - val_accuracy: 0.7730 - val_loss:
1.0877 - val_sparse_categorical_crossentropy: 1.0877
Epoch 13/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 44s 44s/step - accuracy: 0.7606 - loss: 1.0951 -
sparse_categorical_crossentropy: 1.0951 - val_accuracy: 0.7730 - val_loss:
1.0351 - val_sparse_categorical_crossentropy: 1.0351
Epoch 14/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 42s 42s/step - accuracy: 0.7585 - loss: 1.0492 -
sparse_categorical_crossentropy: 1.0492 - val_accuracy: 0.7729 - val_loss:
0.9772 - val_sparse_categorical_crossentropy: 0.9772
Epoch 15/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 40s 40s/step - accuracy: 0.7863 - loss: 0.9616 -
sparse_categorical_crossentropy: 0.9616 - val_accuracy: 0.7725 - val_loss:
0.9143 - val_sparse_categorical_crossentropy: 0.9143
Epoch 16/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 41s 41s/step - accuracy: 0.7767 - loss: 0.9076 -
sparse_categorical_crossentropy: 0.9076 - val_accuracy: 0.7723 - val_loss:
0.8491 - val_sparse_categorical_crossentropy: 0.8491
Epoch 17/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 39s 39s/step - accuracy: 0.7843 - loss: 0.8365 -
sparse_categorical_crossentropy: 0.8365 - val_accuracy: 0.7723 - val_loss:
0.7845 - val_sparse_categorical_crossentropy: 0.7845
Epoch 18/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 39s 39s/step - accuracy: 0.7664 - loss: 0.7854 -
sparse_categorical_crossentropy: 0.7854 - val_accuracy: 0.7723 - val_loss:
0.7242 - val_sparse_categorical_crossentropy: 0.7242
Epoch 19/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 41s 41s/step - accuracy: 0.7629 - loss: 0.7440 -
sparse_categorical_crossentropy: 0.7440 - val_accuracy: 0.7722 - val_loss:
0.6714 - val_sparse_categorical_crossentropy: 0.6714
Epoch 20/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 40s 40s/step - accuracy: 0.7817 - loss: 0.6465 -
```

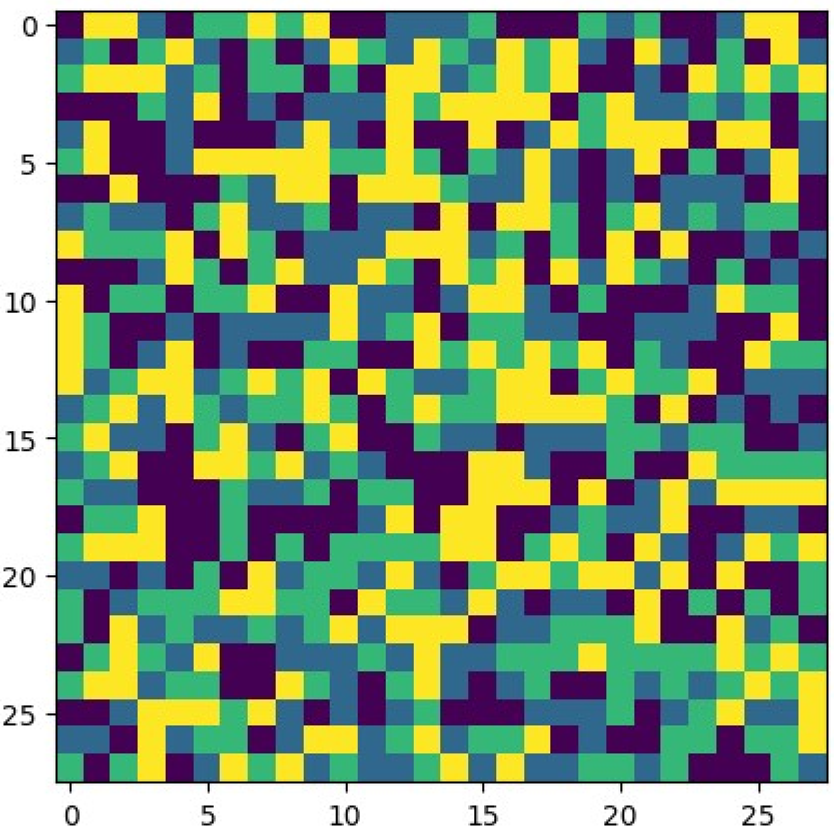
```
sparse_categorical_crossentropy: 0.6465 - val_accuracy: 0.7721 - val_loss:
0.6279 - val_sparse_categorical_crossentropy: 0.6279
Epoch 21/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 46s 46s/step - accuracy: 0.7679 - loss: 0.6383 -
sparse_categorical_crossentropy: 0.6383 - val_accuracy: 0.7720 - val_loss:
0.5936 - val_sparse_categorical_crossentropy: 0.5936
Epoch 22/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 41s 41s/step - accuracy: 0.7747 - loss: 0.6040 -
sparse_categorical_crossentropy: 0.6040 - val_accuracy: 0.7716 - val_loss:
0.5651 - val_sparse_categorical_crossentropy: 0.5651
Epoch 23/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 44s 44s/step - accuracy: 0.7739 - loss: 0.5610 -
sparse_categorical_crossentropy: 0.5610 - val_accuracy: 0.7711 - val_loss:
0.5411 - val_sparse_categorical_crossentropy: 0.5411
Epoch 24/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 41s 41s/step - accuracy: 0.7657 - loss: 0.5596 -
sparse_categorical_crossentropy: 0.5596 - val_accuracy: 0.7699 - val_loss:
0.5206 - val_sparse_categorical_crossentropy: 0.5206
Epoch 25/25
1/1 ━━━━━━━━━━━━━━━━━━━━ 42s 42s/step - accuracy: 0.7752 - loss: 0.5018 -
sparse_categorical_crossentropy: 0.5018 - val_accuracy: 0.7699 - val_loss:
0.5031 - val_sparse_categorical_crossentropy: 0.5031
```

```
Out[8]: <keras.src.callbacks.history.History at 0x7fa93477fc90>
```

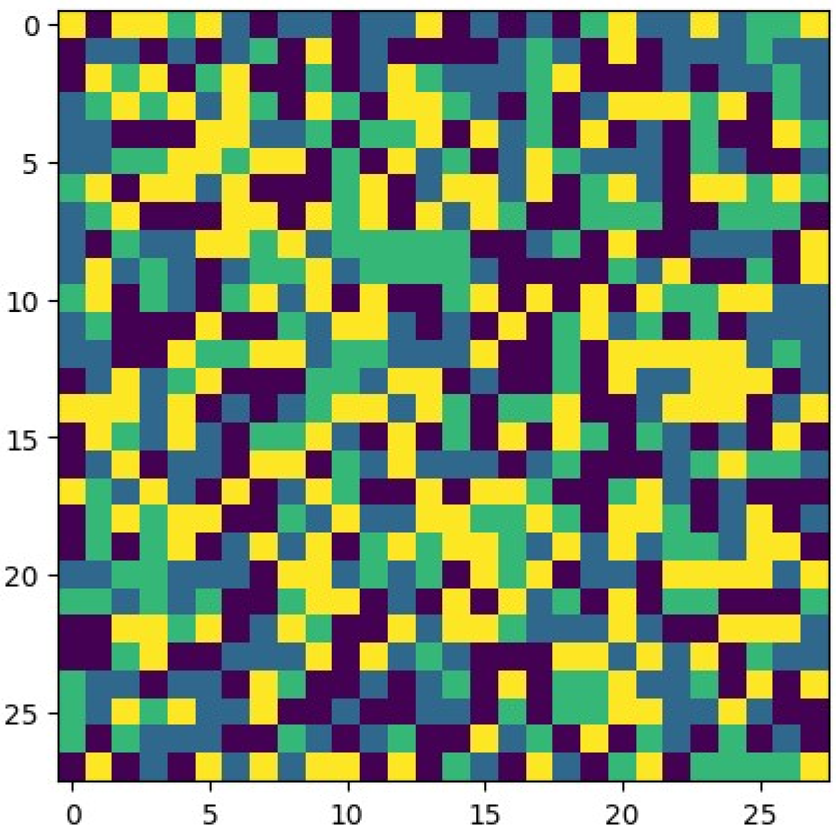




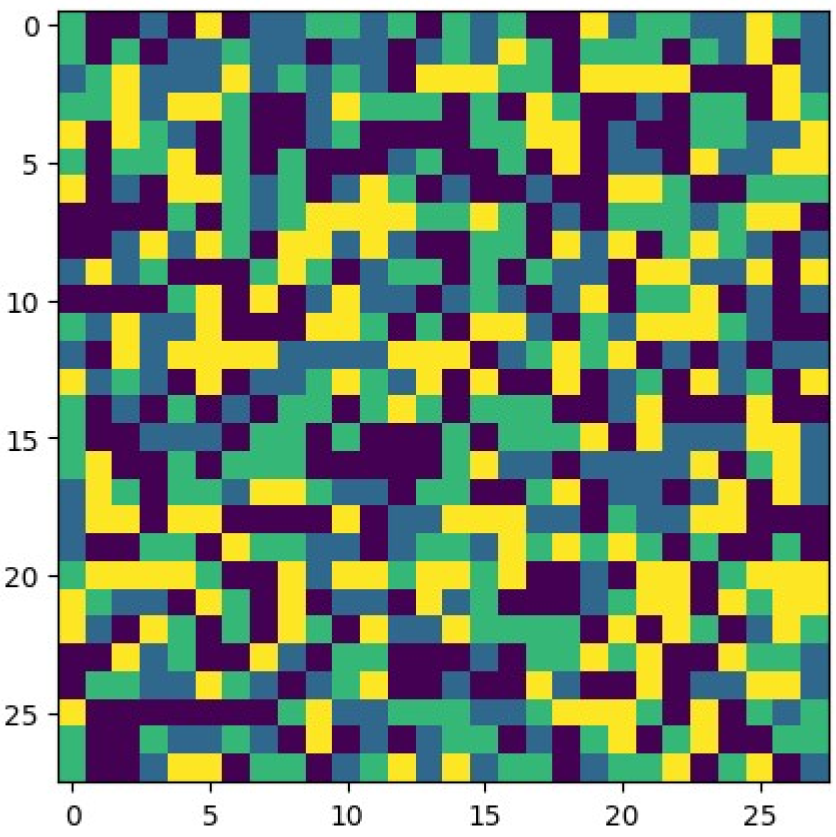
0
5
10
15
20
25



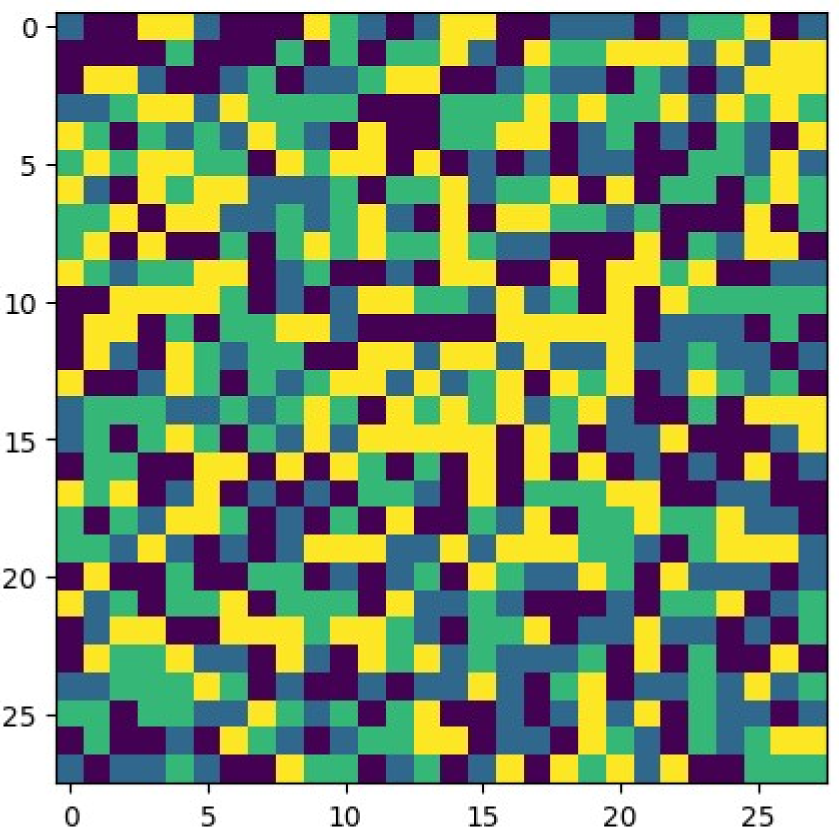
0
5
10
15
20
25

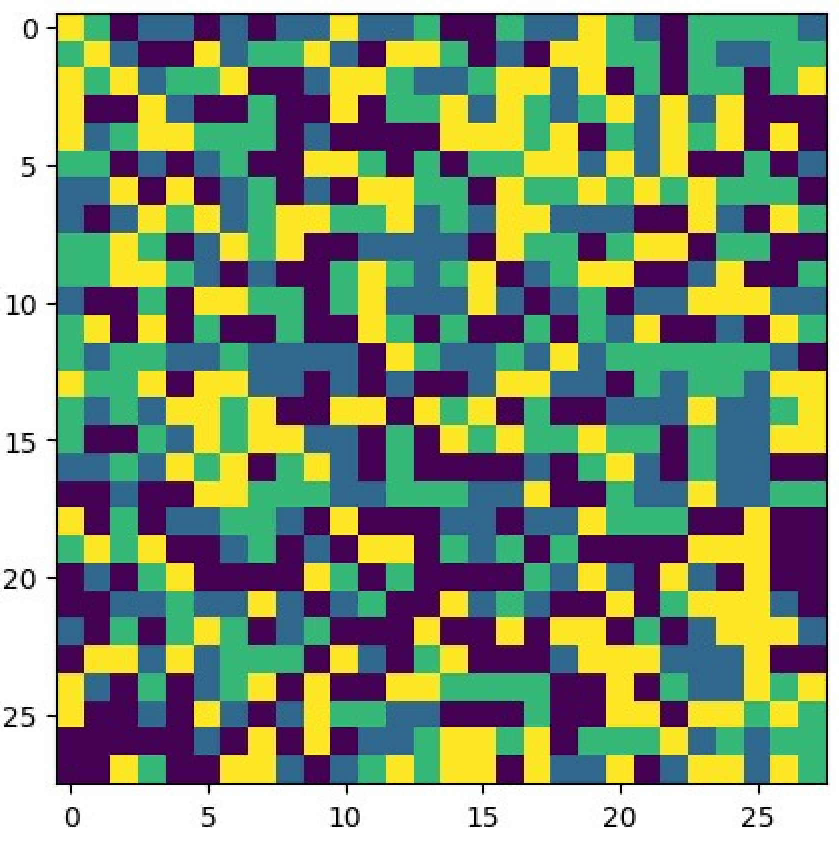


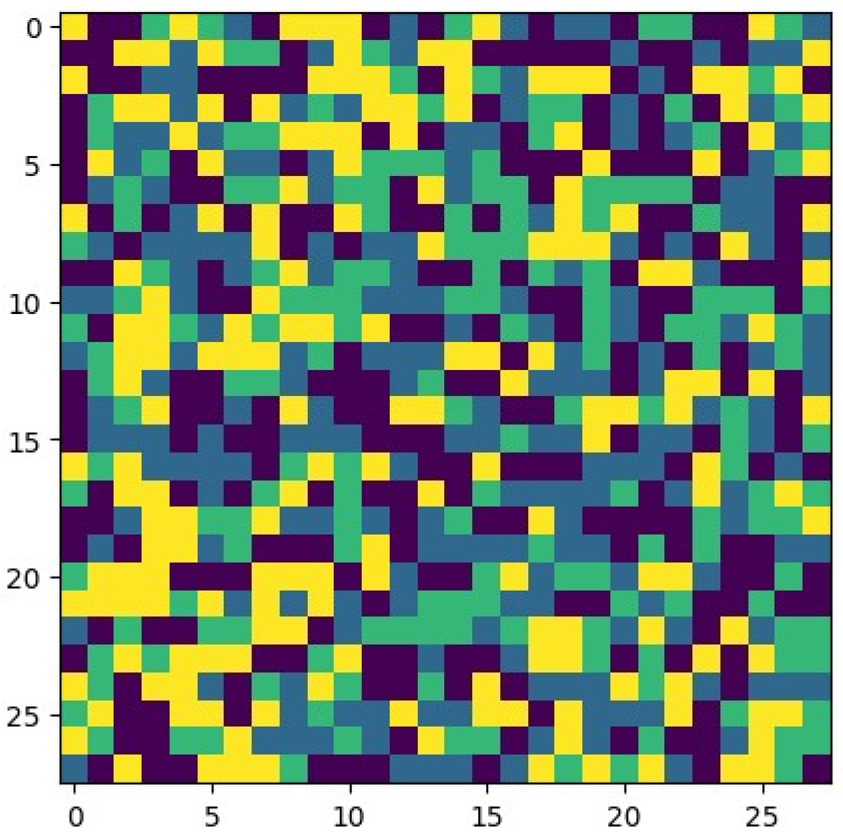
0
5
10
15
20
25

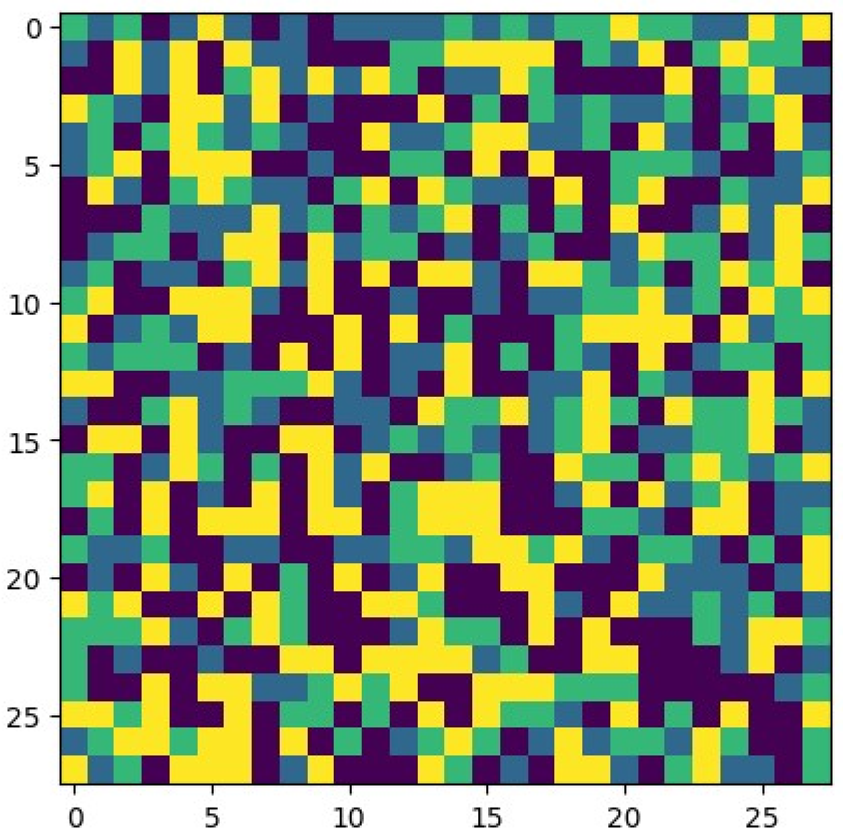


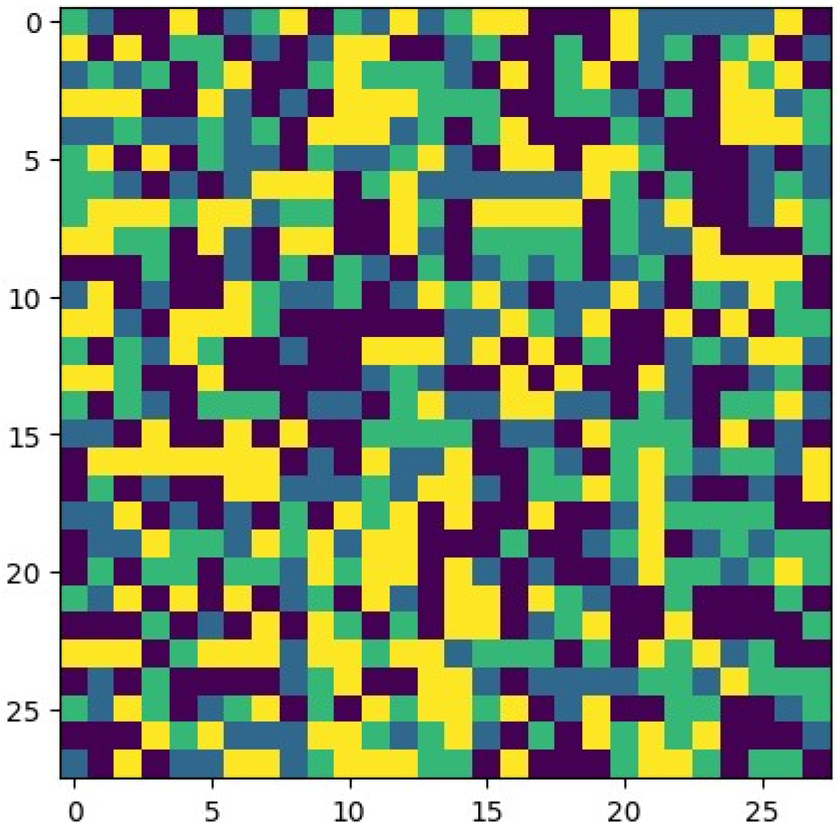
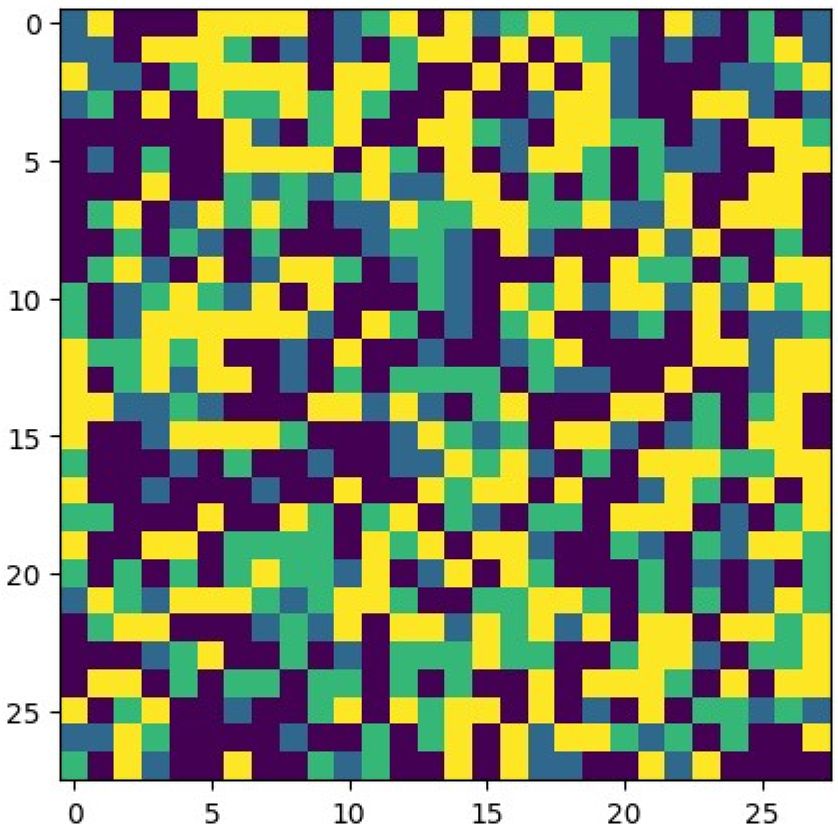
0
5
10
15
20
25

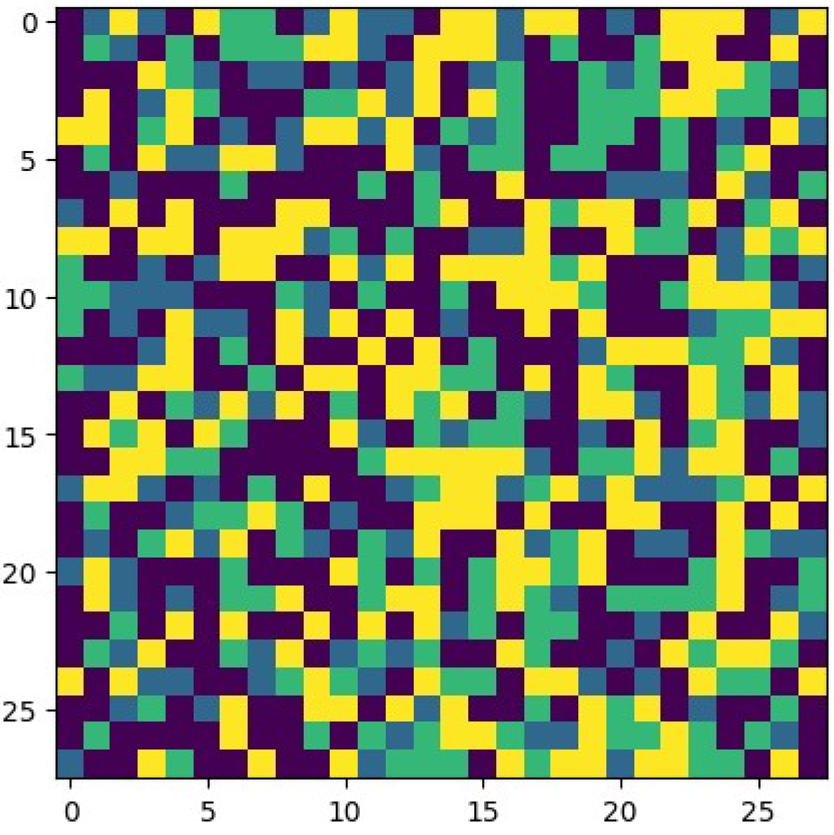

0
5
10
15
20
25
0
5
10
15
20
25

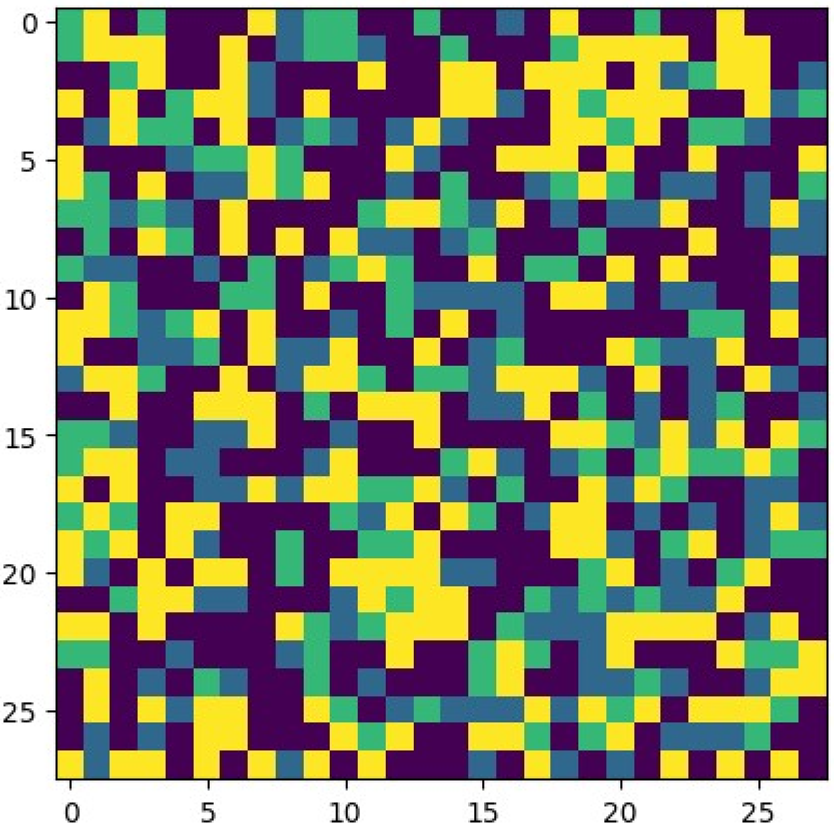

0
5
10
15
20
25
0
5
10
15
20
25

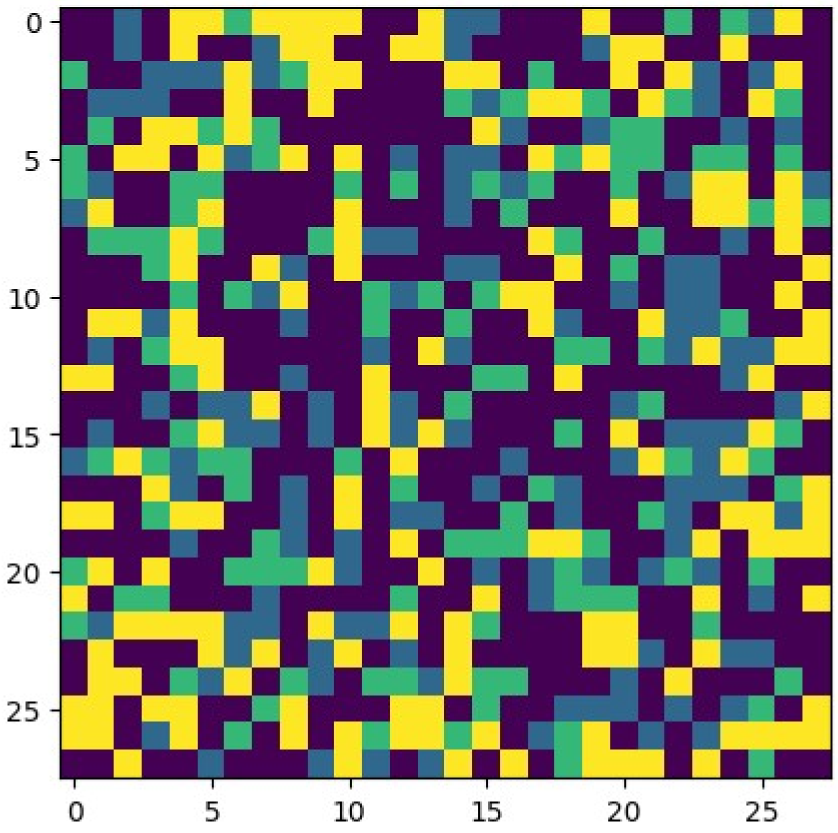
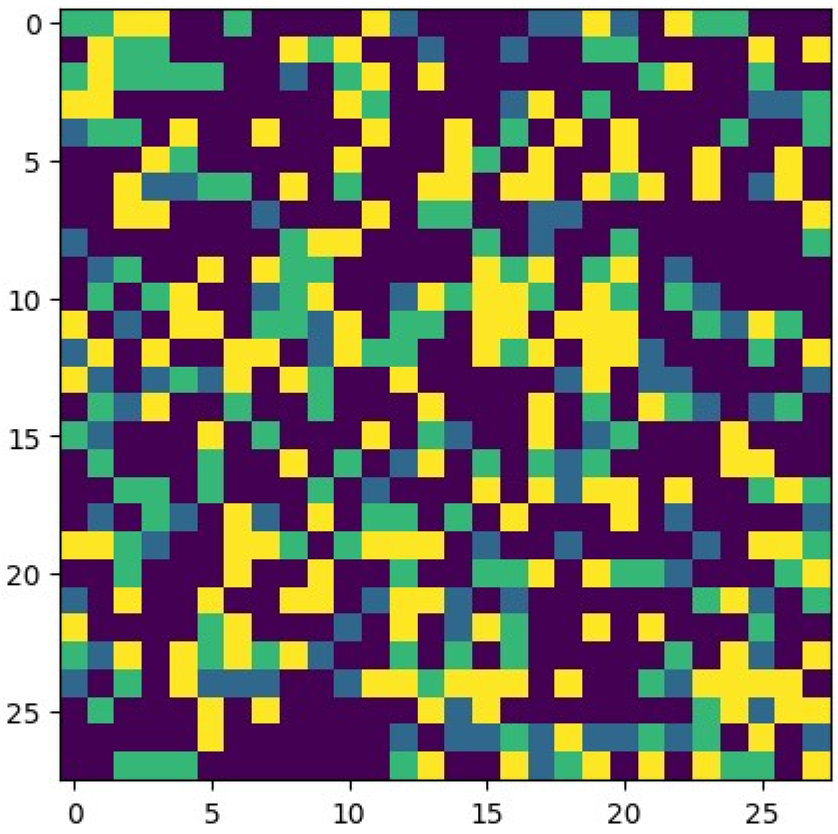

0
5
10
15
20
25
0
5
10
15
20
25

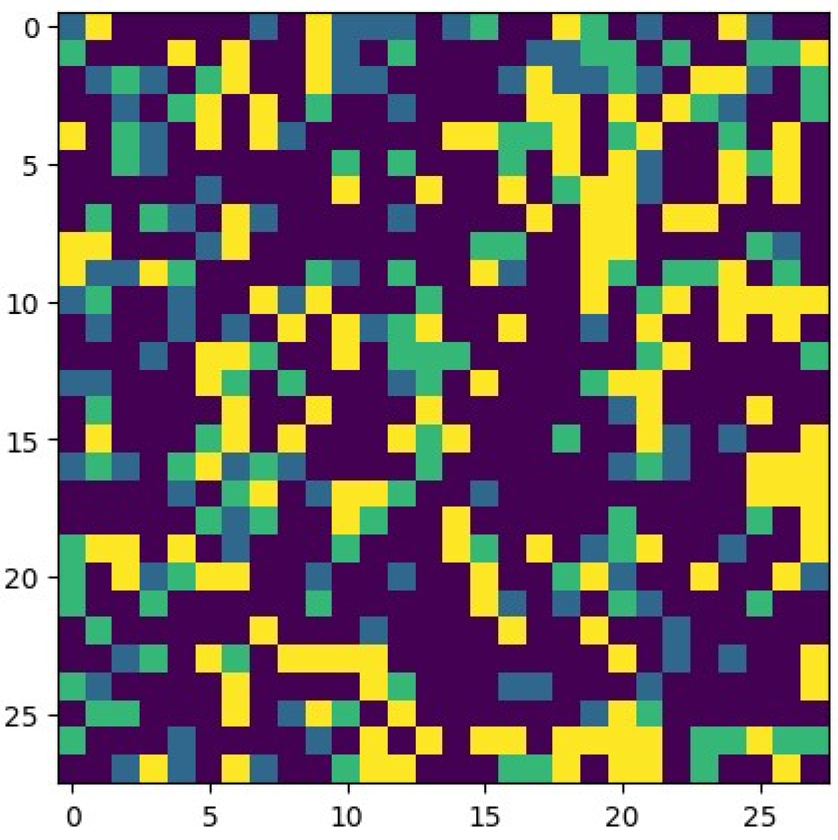

0
5
10
15
20
25
0
5
10
15
20
25

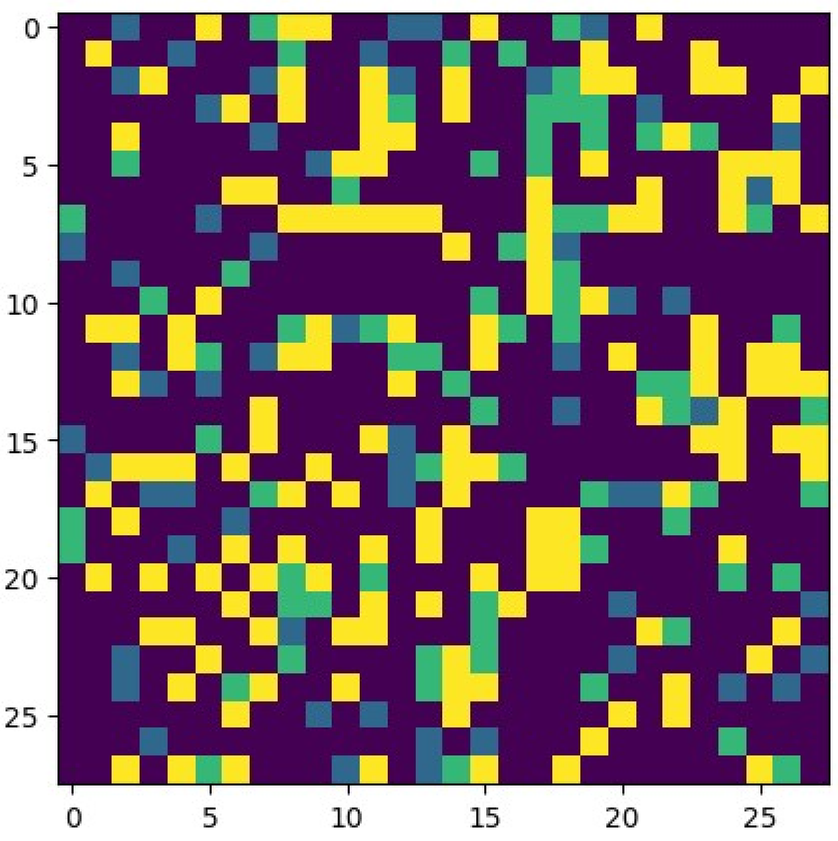

0
5
10
15
20
25
0
5
10
15
20
25

0
5
10
15
20
25
0
5
10
15
20
25

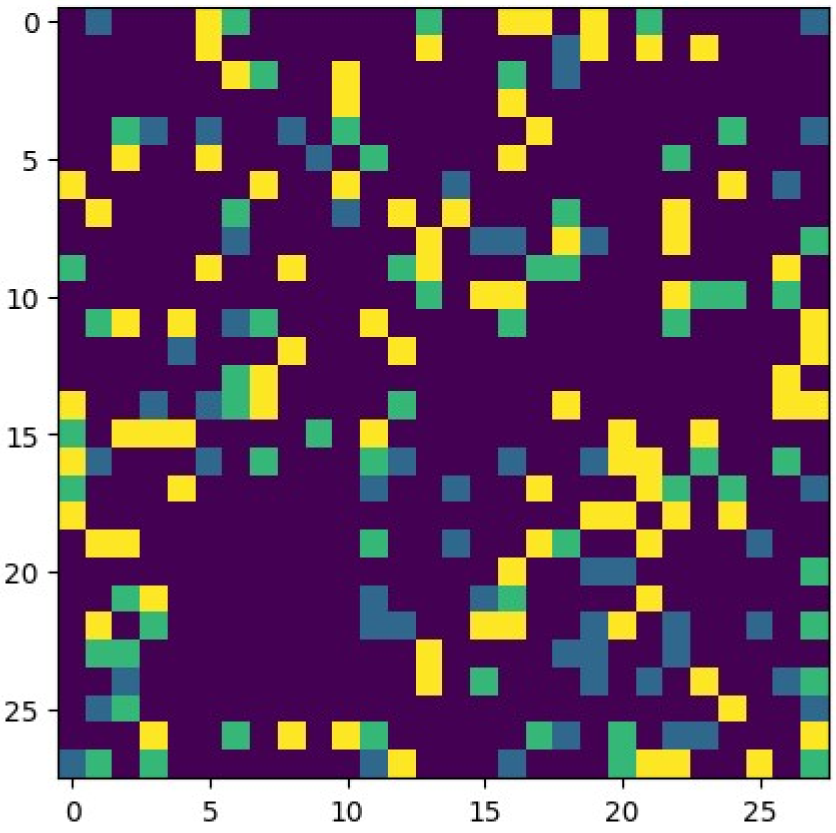

0
5
10
15
20
25
0
5
10
15
20
25

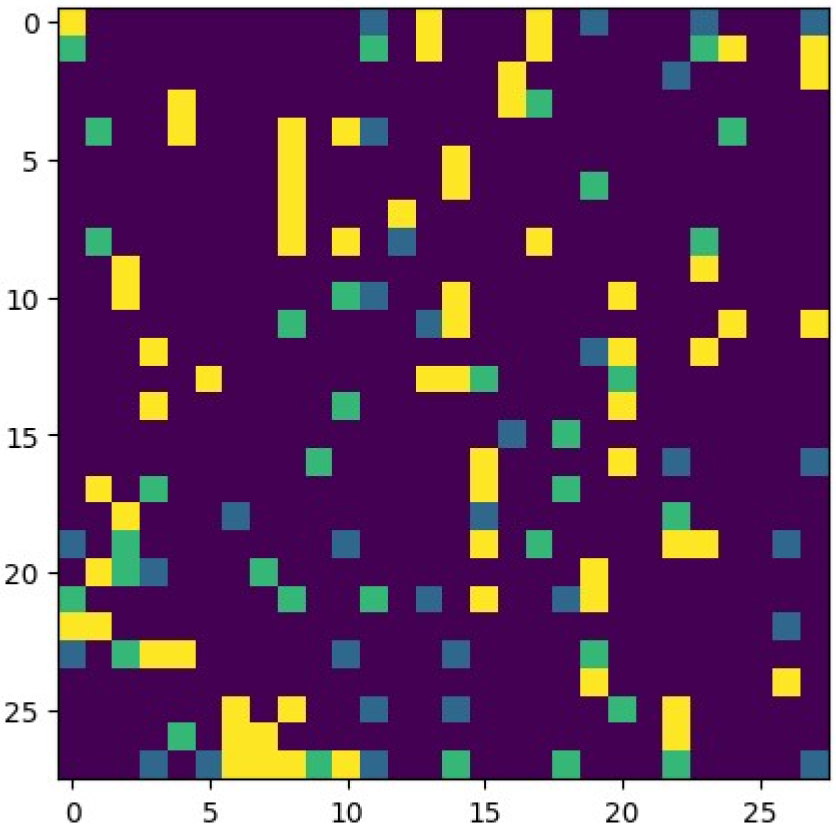

0
5
10
15
20
25
0
5
10
15
20
25

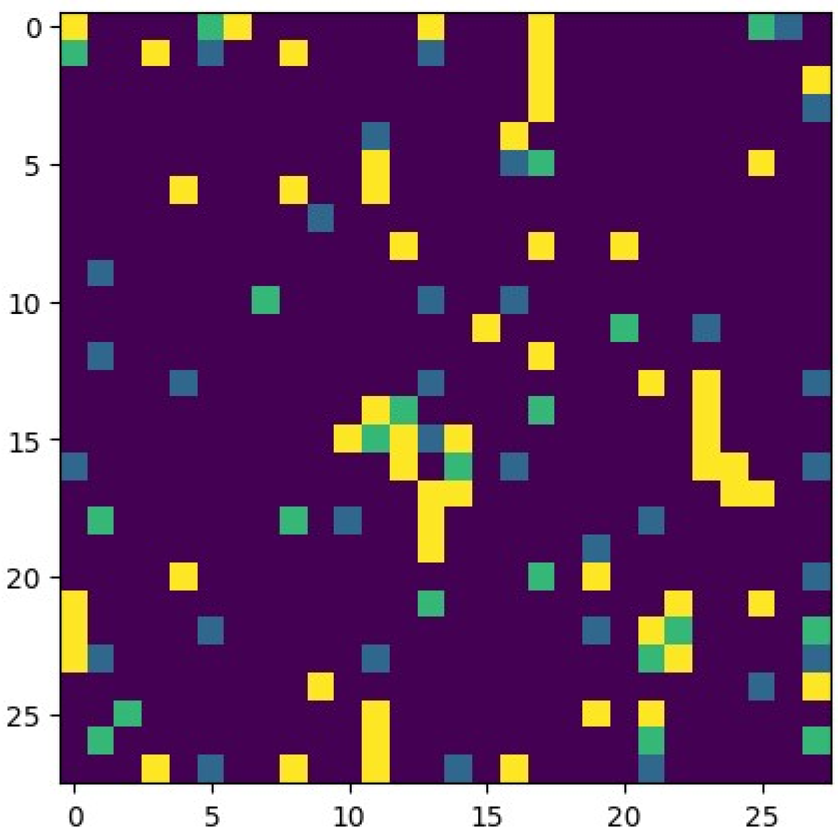

0
5
10
15
20
25
0
5
10
15
20
25

0
5
10
15
20
25
0
5
10
15
20
25

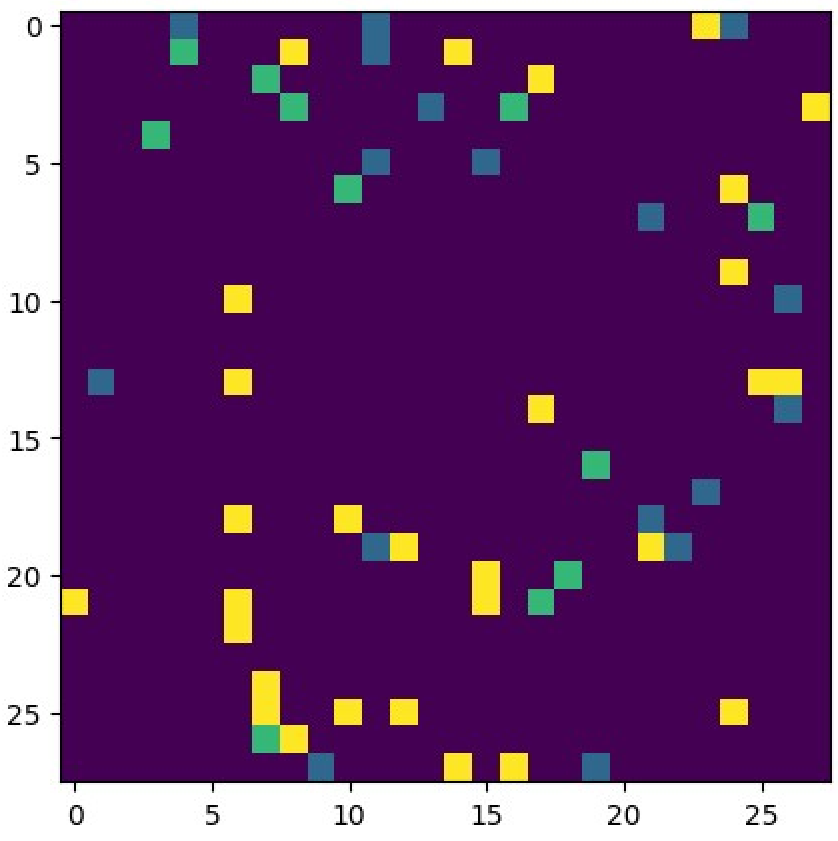

0
5
10
15
20
25
0
5
10
15
20
25

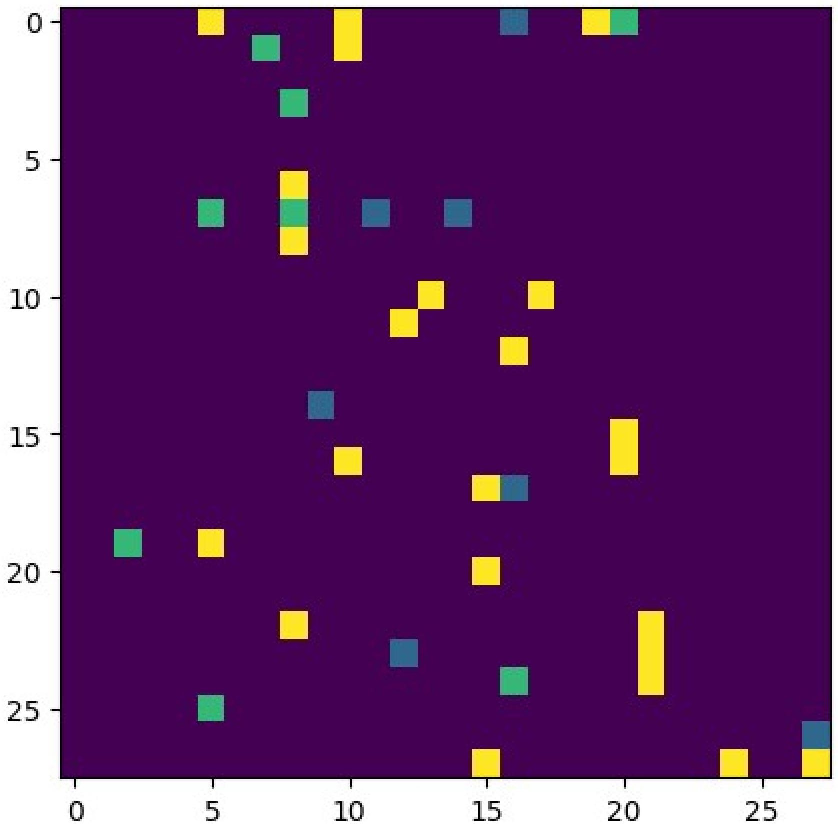

0
5
10
15
20
25
0
5
10
15
20
25

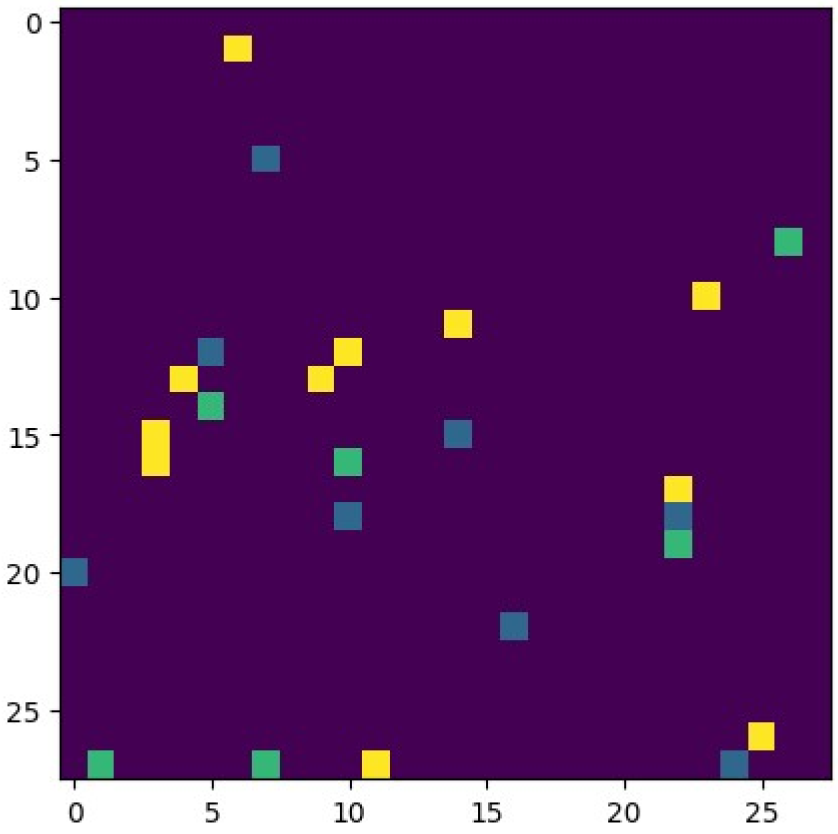

0
5
10
15
20
25
0
5
10
15
20
25

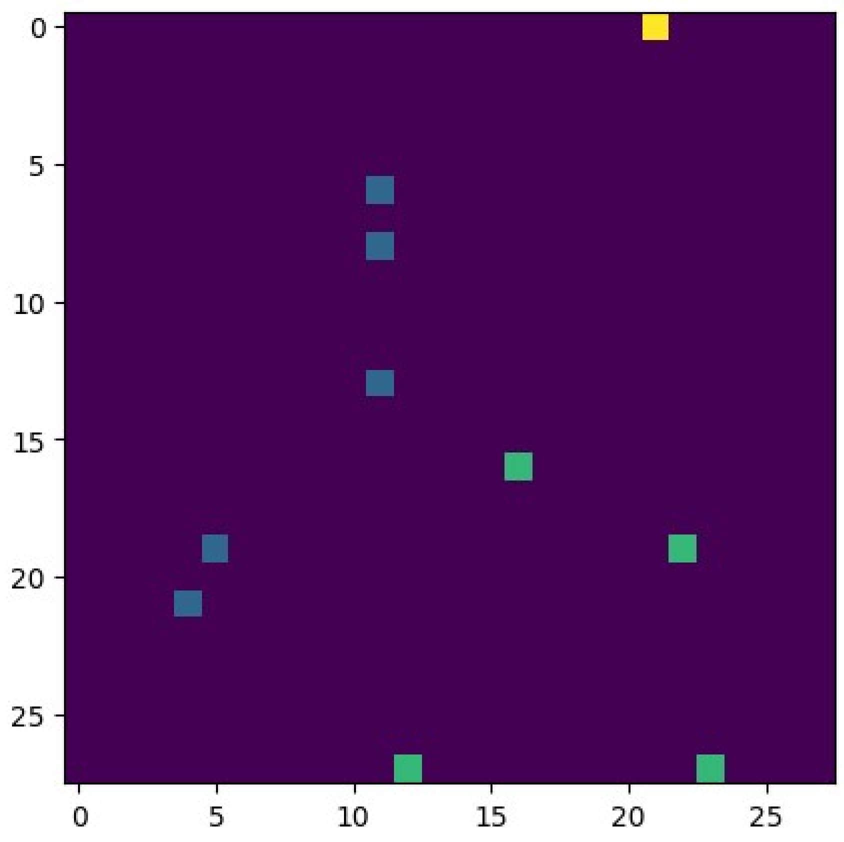

0
5
10
15
20
25
0
5
10
15
20
25

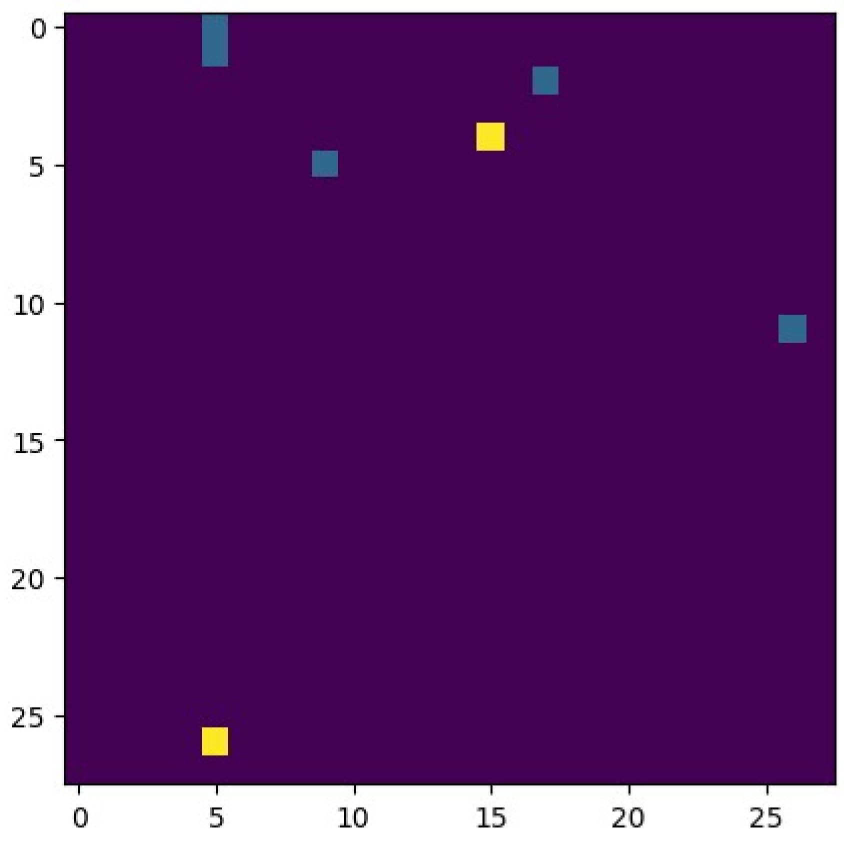

0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25





In [9]:
```
generated_images = image_generator_callback.generate(temperature=1.0)
display(Image(generated_images))
```

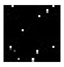
```
In [21]: try:
             pixel_cnn.save("./models/pixelcnn.keras", overwrite=True)
             pixel_cnn.save("./models/pixel_cnn.h5", overwrite=True)
         except Exception as e:
             print(str(e)[:-1])
             print("Model Name already exists")
```

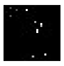
```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()`
or `keras.saving.save_model(model)`. This file format is considered legacy
. We recommend using instead the native Keras format, e.g. `model.save('my
_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.

Unable to synchronously create dataset (name already exists
Model Name already exists
```

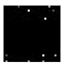
```
In [11]: display(Image(generated_images))
```

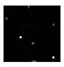
```
In [12]: pixel_cnn.history.history.keys()

Out[12]: dict_keys([])
```

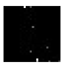
```
In [20]: def deprocess_image(x):
             x = np.stack((x, x, x), 2)
             x *= 255.0
             x = np.clip(x, 0, 255).astype("uint8")
             return x

         generated_images = image_generator_callback.generate(temperature=1.0)

         for i, pic in enumerate(generated_images):
             tf.keras.utils.save_img("./outputs/generative_outputs/pixelcnn_outputs/

         for i in range(0,9):
             display(Image(f"./outputs/generative_outputs/pixelcnn_outputs/generated
```











In [14]:
```
%timeit pixel_cnn.fit
```
```
33.7 ns ± 0.205 ns per loop (mean ± std. dev. of 7 runs, 10,000,000 loops
each)
```

In [15]:
```
pixel_cnn.evaluate(input_data, output_data)
```
```
W0000 00:00:1712485422.559829  666377 assert_op.cc:38] Ignoring Assert ope
rator compile_loss/sparse_categorical_crossentropy/SparseSoftmaxCrossEntro
pyWithLogits/assert_equal_1/Assert/Assert
W0000 00:00:1712485422.565238  666377 assert_op.cc:38] Ignoring Assert ope
rator sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/
assert_equal_1/Assert/Assert

1875/1875 ━━━━━━━━━━━━━━━━━━━━ 17s 9ms/step - accuracy: 0.7729 - loss: 0.4
974 - sparse_categorical_crossentropy: 0.4974
```

Out[15]:
```
[0.4993739128112793, 0.7715013027191162, 0.4993739128112793]
```

In [16]:
```
pixel_cnn.get_metrics_result()
```

Out[16]:
```
{'accuracy': 0.7715013027191162,
 'loss': 0.4993739128112793,
 'sparse_categorical_crossentropy': 0.4993739128112793}
```

In [17]:
```
pixel_cnn.history.history.keys()
```

Out[17]:
```
dict_keys([])
```

In [ ]:
```

```