ARIAN SHARIAT
MOHAMMAD MOHAMMADI

LICENSE PLATE RECOGNITION

WHY LICENSE PLATE RECOGNITION?

- We are amateurs in neural network
- Dataset availability
- Widely analyzed

INPUTS

- A set of pictures with labels (11.5K)

Train: 11K images

Test: 0.5K images

Width: 128 **X** Height: 64

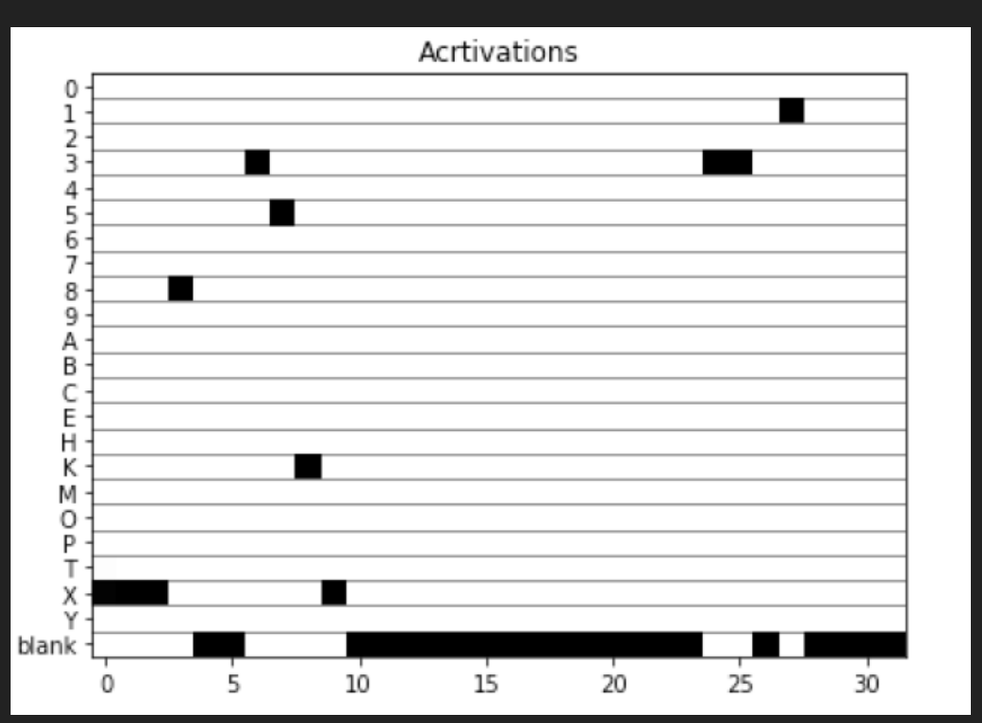
OUTPUTS

- Output is a set of strings

Generated from characters with the highest probability of presence

8 characters long

(0,1~9) and (A to Z)



Predicted: X835KX31
True: X835KX31

CHALLENGES

- Which neural network to use and why?
- Which loss function to use and why?
- Where to get data from?

IMPLEMENTATION

- Architecture
- Layers
- Loss Function

WE USE CNN AS NEURAL NETWORK

- CNN is a class of deep neural networks
- Most common neural network applied analyzing visual imagery
- CNN uses a variation of multilayer perceptrons
- CNN requires minimal preprocessing because of it’s various multilayer perceptrons design

LAYERS

- InputLayer
- Conv2D
- MaxPoolin2D
- Reshape
- Dense
- GRU (Deleted)
- Activation layer(Softmax)

WE USE CTC AS LOSS FUNCTION, WHY?

- Difference of different images beginning will be marked empty-space by CTC-blanks

- Occupancy of one character in multiple time-steps will be merged by CTC

RUNNING THE CODE USING ONE EPOCH WITH GRU

```
Using TensorFlow backend.
```

```
TensorFlow version: 1.12.0
Keras version: 2.2.4
Max plate length in "": 8
Max plate length in "": 8
{'Y', 'X', 'E', 'K', 'M', '5', '2', '3', 'T', '6', '0', 'B', 'C', 'P', '7', '9', '4', 'H', 'O', '1', 'A', '8'}
Letters in train and val do match
Letters: 0 1 2 3 4 5 6 7 8 9 A B C E H K M O P T X Y
Text generator output (data which will be fed into the neutral network):
1) the_input (image)
2) the_labels (plate number): A024AK54 is encoded as [10, 0, 2, 4, 10, 15, 5, 4]
3) input_length (width of image that is fed to the loss function): 30 == 128 / 4 - 2
4) label_length (length of plate number): 8
```

Layer (type)	Output Shape	Param #	Connected to
the_input (InputLayer)	(None, 128, 64, 1)	0	
conv1 (Conv2D)	(None, 128, 64, 16)	160	the_input[0][0]
max1 (MaxPooling2D)	(None, 64, 32, 16)	0	conv1[0][0]
conv2 (Conv2D)	(None, 64, 32, 16)	2320	max1[0][0]
max2 (MaxPooling2D)	(None, 32, 16, 16)	0	conv2[0][0]
reshape (Reshape)	(None, 32, 256)	0	max2[0][0]
dense1 (Dense)	(None, 32, 32)	8224	reshape[0][0]
gru1 (GRU)	(None, 32, 512)	837120	dense1[0][0]
gru1_b (GRU)	(None, 32, 512)	837120	dense1[0][0]
add_1 (Add)	(None, 32, 512)	0	gru1[0][0] gru1_b[0][0]
gru2 (GRU)	(None, 32, 512)	1574400	add_1[0][0]

RUNNING THE CODE USING ONE EPOCH WITH GRU (. . .CONTINUED)

```
gru2 (GRU)                      (None, 32, 512)      1574400     add_1[0][0]
gru2_b (GRU)                    (None, 32, 512)      1574400     add_1[0][0]
concatenate_1 (Concatenate)     (None, 32, 1024)     0           gru2[0][0]
                                                                 gru2_b[0][0]
dense2 (Dense)                  (None, 32, 23)       23575       concatenate_1[0][0]
softmax (Activation)            (None, 32, 23)       0           dense2[0][0]
==================================================================================================
Total params: 4,857,319
Trainable params: 4,857,319
Non-trainable params: 0
__________________________________________________________________________________________________
Epoch 1/1
10281/10281 [==============================] - 2428s 236ms/step - loss: 1.9692 - val_loss: 0.1442
Predicted: A036CT78
True: A036CT78
Predicted: A065CA30
True: A065CA30

C:\Users\Amu Chaei\Miniconda3\envs\license_plate\lib\site-packages\matplotlib\figure.py:2366: UserWarning: This figure includes
Axes that are not compatible with tight_layout, so results might be incorrect.
  warnings.warn("This figure includes Axes that are not compatible "

Predicted: A083YA10
True: A083YA10
Predicted: A128HB61
True: A128HB61
Predicted: A141AP01
True: A141AP01
Predicted: A160HE95
True: A160HE95
Predicted: A183KE05
True: A183KE05
Predicted: A225EP73
True: A225EP73
```

RESULTS

RUNNING THE CODE USING 4 EPOCHS WITHOUT GRU

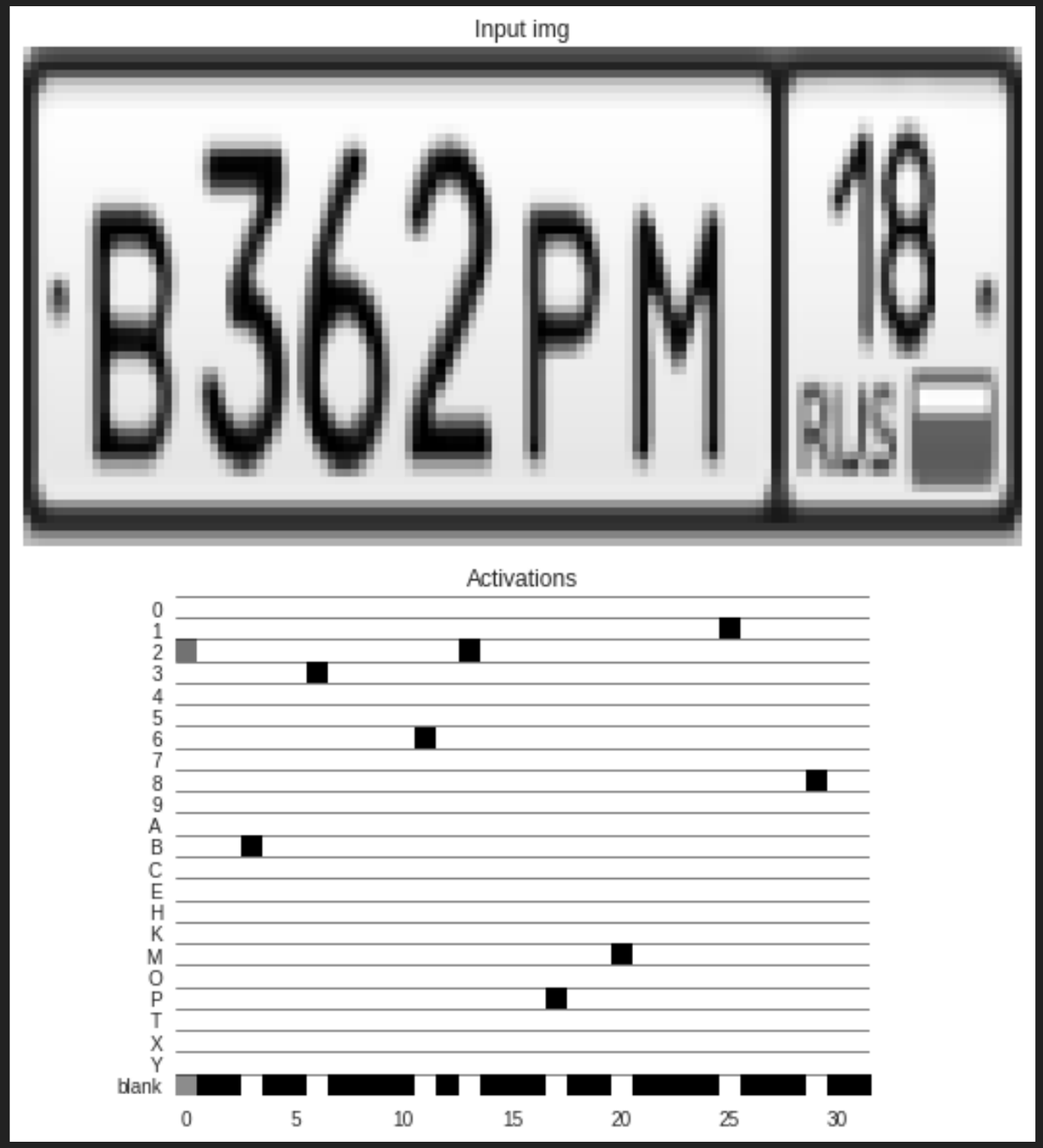
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
the_input (InputLayer)       (None, 128, 64, 1)        0
_________________________________________________________________
conv1 (Conv2D)               (None, 128, 64, 16)       160
_________________________________________________________________
max1 (MaxPooling2D)          (None, 64, 32, 16)        0
_________________________________________________________________
conv2 (Conv2D)               (None, 64, 32, 16)        2320
_________________________________________________________________
max2 (MaxPooling2D)          (None, 32, 16, 16)        0
_________________________________________________________________
reshape (Reshape)            (None, 32, 256)           0
_________________________________________________________________
dense1 (Dense)               (None, 32, 32)            8224
_________________________________________________________________
dense2 (Dense)               (None, 32, 23)            759
_________________________________________________________________
softmax (Activation)         (None, 32, 23)            0
=================================================================
Total params: 11,463
Trainable params: 11,463
Non-trainable params: 0
_________________________________________________________________
```

RUNNING THE CODE USING 4 EPOCHS WITHOUT GRU (. . . CONTINUED)

```
Epoch 1/4
10281/10281 [==============================] - 382s 37ms/step - loss: 0.3607 - val_loss: 1.5033e-04
Epoch 2/4
10281/10281 [==============================] - 386s 38ms/step - loss: 0.0045 - val_loss: 8.6403e-05
Epoch 3/4
10281/10281 [==============================] - 392s 38ms/step - loss: 0.0045 - val_loss: 8.4615e-05
Epoch 4/4
10281/10281 [==============================] - 391s 38ms/step - loss: 0.0045 - val_loss: 7.8257e-05
```

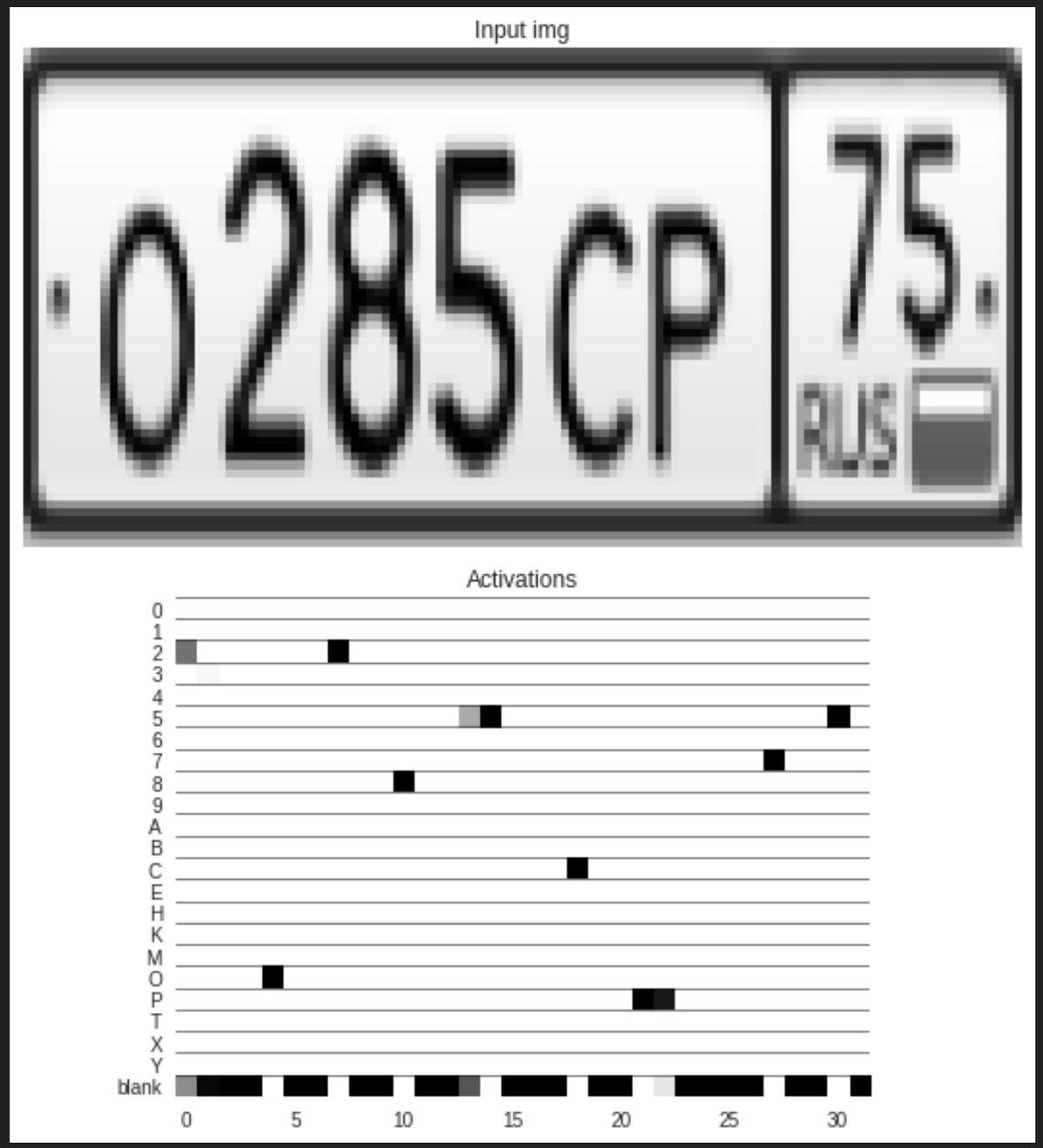
RUNNING THE CODE USING 4 EPOCHS WITHOUT GRU (. . . CONTINUED)

Predicted: B362PM18
True: B362PM18



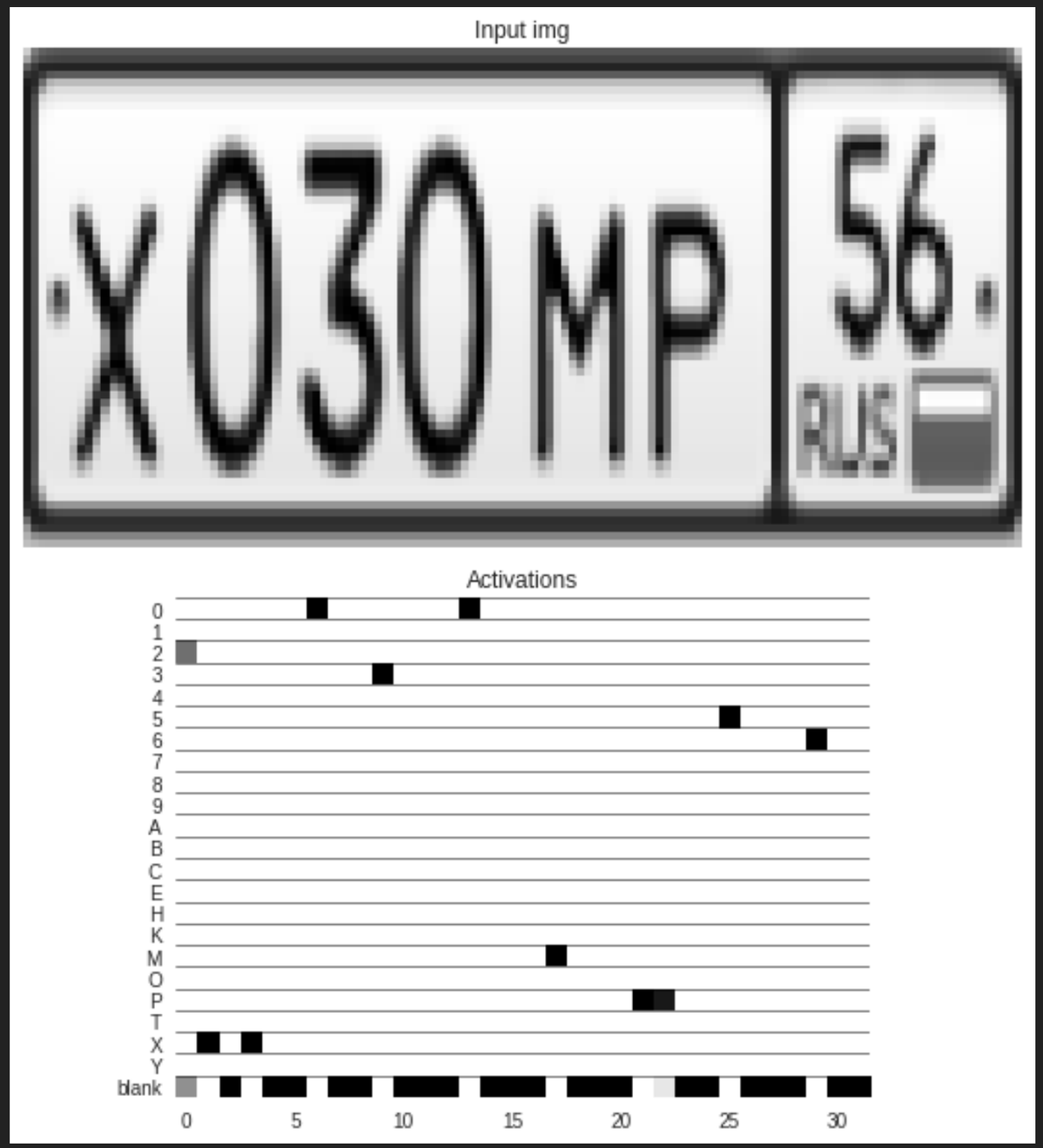
RUNNING THE CODE USING 4 EPOCHS WITHOUT GRU (. . . CONTINUED)

Predicted: O285CP75
True: O285CP75



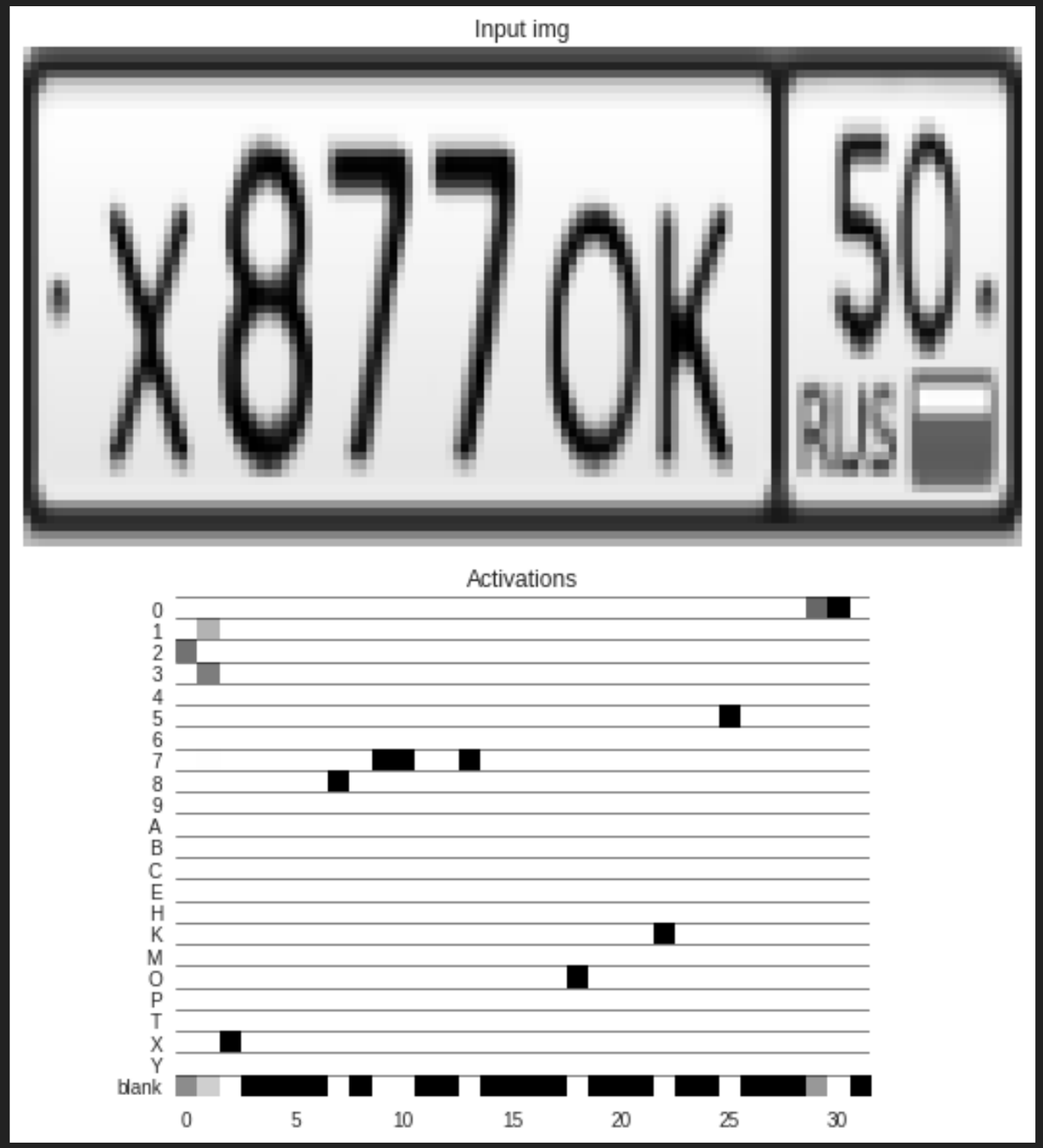
RUNNING THE CODE USING 4 EPOCHS WITHOUT GRU (. . . CONTINUED)

Predicted: X030MP56
True: X030MP56



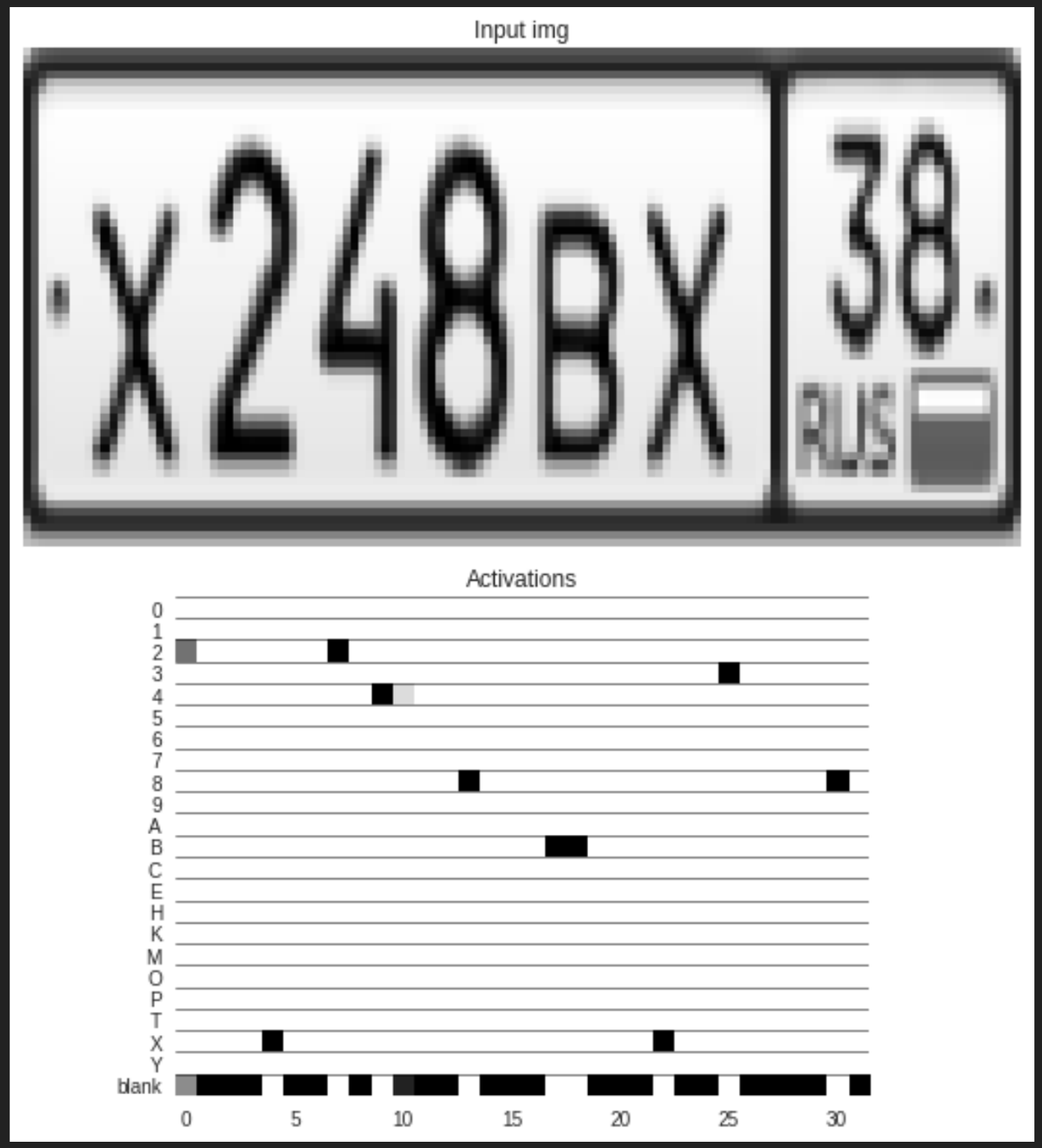
RUNNING THE CODE USING 4 EPOCHS WITHOUT GRU (. . . CONTINUED)

Predicted: X877OK50
True: X877OK50



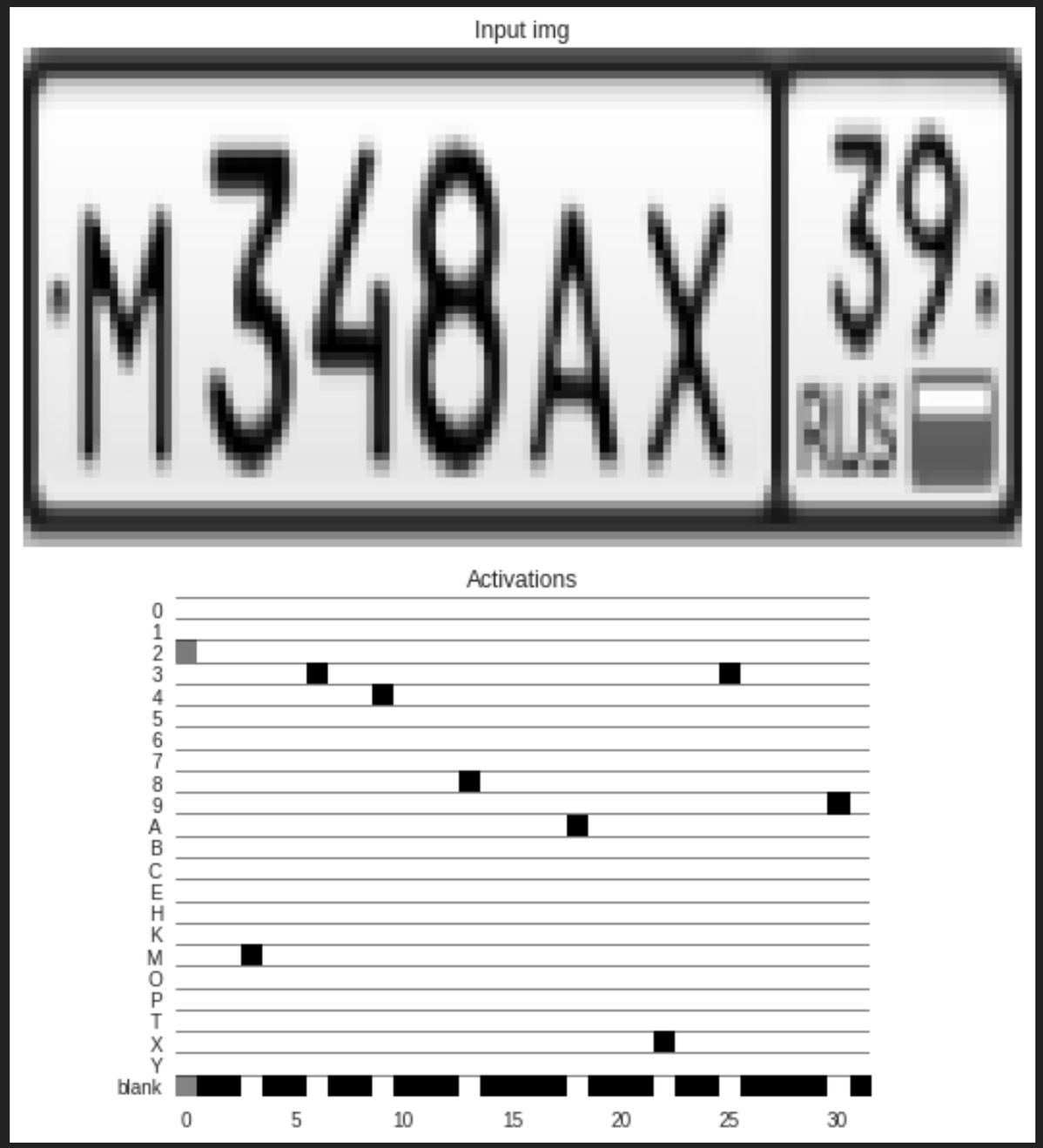
RUNNING THE CODE USING 4 EPOCHS WITHOUT GRU (. . . CONTINUED)

Predicted: X248BX38
True: X248BX38



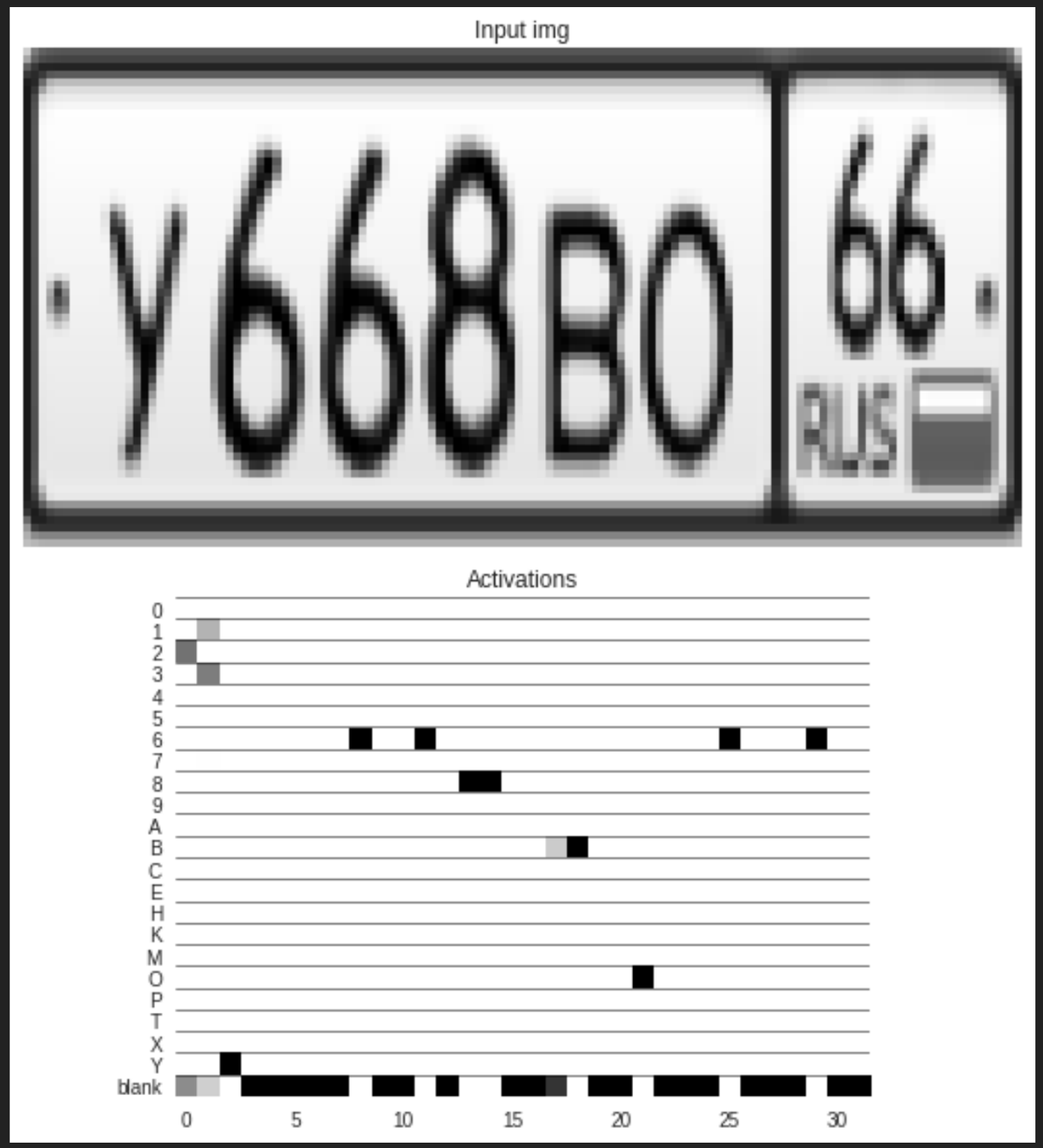
RUNNING THE CODE USING 4 EPOCHS WITHOUT GRU (. . . CONTINUED)

Predicted: M348AX39
True: M348AX39



RUNNING THE CODE USING 4 EPOCHS WITHOUT GRU (. . . CONTINUED)

Predicted: Y668BO66
True: Y668BO66



THE END.