Homework 3
Neural Networks

University of Bucharest, Faculty of Mathematics and Informatics
Sparktech Software

December, 2018

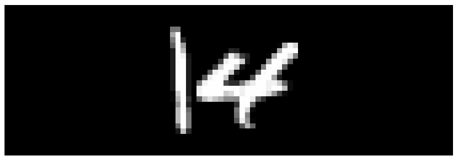
Soft Deadline: Tuesday, January 14, 2019, 11:59 PM. If you do not make this deadline, your submission will receive a 20 points deduction.

Hard Deadline: Wednesday, January 15, 2019, 2:00 PM. If you do not make this deadline, your submission will not be graded.

1 Introduction

In this homework, you are supposed to build a neural model that can read numbers and adds them together. Numbers are formed using images from the MNIST [5] dataset which are concatenated after some random translations / rotations are applied to them. The images you will work with will contain sequences ranging from 1 to 3 digits. The maximum number you can have in an image is 255. Figure 1 provides an example of a data sample.

Training data will be generated using the *training_generator()* function from *data_generator.py* file. You may **not** modify this file. For testing your models, you will use the *test_generator()* function. Each of these functions returns a pair of images of fixed shape (28x84 px), their labels, and the result of the addition.


(a) Label: 14

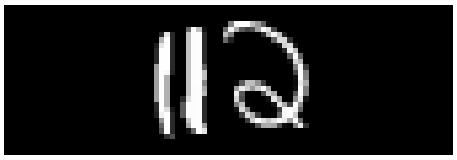

(b) Label: 112

Figure 1: Adding the two numbers: 14 + 112 = 126

2 Task 1: Image Classification (70p)

In this task, you have to build a neural model that recognizes the sequence of digits in a given image. You may choose to work with the images as they are or you can try to split them into parts. You can try different architectures for this task : MLPs, CNNs or any other. You will receive up to maximum points if you use a CNN, otherwise maximum 40p.

For your interest, check out some papers on the subject: [4], [2], [9],

3 Task 2: Addition (30p)

Build a recurrent neural network that adds two numbers. For this task you need to use the labels of the images from the first task as inputs and their sum as the label. Your model must have at least one recurrent layer(LSTM/GRU/SimpleRNN etc.). You will receive up to maximum points if you use an encoder-decoder architecture, otherwise maximum 15p.

If you are interested, check out some papers on the subject: [10], [8], [6], [1]

4 Bonus: End-to-End Network (30p)

Build an end-to-end **trainable** model that receives two images containing numbers and outputs their sum. You have the freedom to experiment with anything you like. There should be no intermediary steps, just a single neural model that does everything.

5 Grading Policy

- The submitted code is sufficient for us to test the models.
- Describe the neural network architecture (layers, types of regularization, activations etc.).
- Include plots to visualize the behavior of the models (loss, accuracy, filter activations etc).
- You should experiment with multiple values for hyperparameters.
- Your decisions must be supported by arguments.

6 Other Considerations

- This homework should be sent via e-mail at the address *ml.lecture@sparktech.ro* in the following format:

```
Subject: [ML - Homework 3] Cobuz_Alexandru_334
Attachements:
    1. Your code or jupyter notebook
    2. Your trained models (h5 file)
         and the tensorflow version you used
    3. PDF if you do did not use a jupyter notebook
```

- You can check out the tensorflow documentation for saving[1] and loading[2] your models.
- Check out this notebook[3] for faster training using Google Colab.
- Check out these papers for building better neural networks: [7], [3]. There are many more, check out each paper's references.
- Make sure all plots are properly labeled (each axis should have a label), and the title is concise, but clear. Use legends when necessary. The meaning of each plot should be clear without the need to inspect the code that produced it.
- Collaboration is allowed and encouraged regarding techniques used, model details (how it works, network architectures, activation functions, etc), but keep in mind that the report and the code submitted has to be **your own work**.
- All code must be written in Python v3.6 (or greater).
- **Do not put your homework on github**

References

[1] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[3] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[1] `https://www.tensorflow.org/api_docs/python/tf/keras/models/save_model`
[2] `https://www.tensorflow.org/api_docs/python/tf/keras/models/load_model`
[3] `https://colab.research.google.com/notebooks/gpu.ipynb`

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[5] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[6] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. Understanding the exploding gradient problem. *CoRR*, abs/1211.5063, 2012.

[7] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[8] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014.

[9] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

[10] Wojciech Zaremba and Ilya Sutskever. Learning to execute. *CoRR*, abs/1410.4615, 2014.