Capstone Proposal

Avraam Th. Tolmidis[1,]

Keywords: Deep Learning, Image recognition, Convolutional Neural Networks

1. Proposal

In this document, a proposal for the Capstone Project of the Udacity Machine Learning Nanodegree will be presented. The aim is, to select a field where the knowledge acquired during the Nanodigree can be used to solve a real-world problem. The interest of the author was primarily Deep Learning, a relevant project proposal is described. Since this is the first real-world problem that will be solved during the Nanodegree, the choice was to solve the problem suggested in the course material. This would make getting support easier.

2. Domain Background

The problem of identifying digits from street images is still considered an open problem. In the recent years there have been advances, mostly due to the use of Deep Learning architectures. An overview of Deep Learning Methods and Applications can be found in [1]. With the availability of hardware that is more efficient in performing large amounts of calculations, the use of Deep Neural Networks for solving difficult problems such as Image Recognition is becoming more common. It additionally leads to an accuracy comparable to that achieved by humans, e.g. 97.84% (per-digit identification in images) [2].

3. Problem Statement

The use of Deep Learning models in order to decode sequences of digits from natural images, will be studied. For this task, Convolutional Neural Networks will be used. The behaviour/performance of the model will be studied, with different network parameters (optimizer, batch size of images used during training, techniques like dropout, etc.). The implementation of the project will be as a Python application, using different Python scripts for independent parts of the application. There will be two versions of the application: one where the bounding boxes of the digits in the images will be provided as input, and a second one, where the model will try to identify the part of the image that contains the digits itself.

Email address: atolmid@gmail.com (Avraam Th. Tolmidis)

4. Datasets and Inputs

The data that will be used will be that from the Google Street View House Number dataset [3]. SVHN is a real-world image dataset for developing machine learning and object recognition algorithms. The images are of small cropped digits (over 600,000) from natural scene images. The dataset is obtained from house numbers in Google Street View images.

The dataset contains 10 classes, 1 for each digit. Digit '1' has label 1, '9' has label 9 and '0' has label 10. The maximum number length is considered to consist of 5 digits. Therefore, to simplify the problem, the number of digits will be considered fixed to 5. To account for the numbers of smaller length, an additional class with label 0 will be added to the dataset. It will represent the digits (actual length of the number == 5).

5. Solution Statement

As already discussed,in order to solve the problem, Deep Neural Networks will be used, and more specifically Convolutional Neural Networks [4] [5]. Several Convolutional layers will be followed by two fully connected layers. There will be 5 features predicted, each of them corresponding to one digit if the number we are trying to identify (0 will be used as a 'no digit' indicator, in numbers that have length smaller than 5 (the maximum considered). The Convolutional layers are common, meaning the same parameters are shared by all features. The final Fully Connected layers are different for each one of the digits. We will study the effect of using dropout, or choosing different parameters, such as different optimisers or number of training steps. Finally, a second version of the model will be created, where besides the 5 features, the network should also be able to identify the bounding boxes of the digits. This will be done using additional nodes in the network and regression. The results from both versions will be compared.

6. Benchmark Model

The Benchmark model for the application will be the number detection accuracy. In the present, the accuracy is defined as described in 7. It will be examined whether the various configurations used can achieve an accuracy comparable to - or not dramatically worse than the one achieved by the configuration the authors of [2] used (96.03% for sequence transcription), or that of humans (98%). The accuracy they achieved is provided, so a comparison is easy to make.

7. Evaluation Metrics

As already mentioned, the evaluation metric will be the detection accuracy. For each of the numbers, the accuracy will be evaluated as shown in Algorithm 1:

Algorithm 1: Estimation of accuracy %

for *all digits i* **do**
  prediction: $p[i]$=$argmax(predictions)$;
  label: $l[i]$=$argmax(labels)$;
end
for *every image* **do**
  successful digits = sum(p[i] == l[i])
end
$accuracy(\%)=100 * count(successfuldigits==5)/numberofimages$

A succesfull detection is only that which includes a correct identification of all five digits (including zero for the lack of one). The percentage of all images, for which all five digits were correctly identified gives us the accuracy.

8. Project Design

In order to solve the problem, that data has to be preprocessed. Images will be scaled down, so that calculations need to be performed only over a more limited amount of pixels. RGB images will be converted to grayscale, as colour information does not affect the identification of the digits themselves. Converting the image to grayscale will allow us to work on arrays with fewer dimensions. Pixel values will be centered around zero, and they will be normalised. Images will be cropped, discarding all other areas except bounding boxes. Finally, the labels will also be processed, in order to conform with the design selected (length of five digits, zeros denoting the absence of a value). Approximately 30% of the original training set will be used as validation test. The models will be created and trained using python [6] and tensorflow [7]

The base model includes five convolution layers with stride one, followed by max pooling, and then two fully connected layers. The optimiser used is Adam. The cost function used is softmax_cross_entropy_with_logits, included in the Tensorflow library. Other configurations that will be tested include the use of dropout [8] after the convolution layer, different optimisers (e.g. Gradient Descent) L1/L2 regularisation (a comparison between the two can be found in [9]), or cost function such as sparse_softmax_cross_entropy_with_logits.

Finally, data augmentation (e.g. random contrast, random flipping, etc) will be explored.

References

[1] D. Y. Li Deng, Deep learning: Methods and applications, Tech. rep. (May 2014).
URL https://www.microsoft.com/en-us/research/publication/deep-learning-methods-and-applications/

[2] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud, V. D. Shet, Multi-digit number recognition from street view imagery using deep convolutional neural networks, CoRR abs/1312.6082.
URL http://arxiv.org/abs/1312.6082

[3] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng, Reading digits in natural images with unsupervised feature learning, in: NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011, 2011.
URL http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf

[4] K. Fukushima, Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position, Biological Cybernetics 36 (4) (1980) 193–202. doi:10.1007/BF00344251.
URL http://dx.doi.org/10.1007/BF00344251

[5] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, in: Proceedings of the IEEE, 1998, pp. 2278–2324.

[6] G. van Rossum, Python Reference Manual, 2nd Edition (Mar. 2006).
URL http://docs.python.org/ref/ref.html

[7] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Man, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Vigas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, Tensorflow: Large-scale machine learning on heterogeneous distributed systems (2015).
URL http://download.tensorflow.org/paper/whitepaper2015.pdf

[8] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
URL http://dl.acm.org/citation.cfm?id=2627435.2670313

[9] A. Y. Ng, Feature selection, l1 vs. l2 regularization, and rotational invariance, in: Proceedings of the Twenty-first International Conference on Machine Learning, ICML '04, ACM, New York, NY, USA, 2004, pp. 78–. doi:10.1145/1015330.1015435.
URL http://doi.acm.org/10.1145/1015330.1015435