Machine Learning with TensorFlow for Mobile and Ubiquitous Interaction

Huy Viet Le
University of Stuttgart
Stuttgart, Germany
huy.le@vis.uni-stuttgart.de

Sven Mayer
University of Stuttgart
Stuttgart, Germany
sven.mayer@vis.uni-stuttgart.de

Niels Henze
University of Stuttgart
Stuttgart, Germany
niels.henze@vis.uni-stuttgart.de

MUM 2017, November 26–29, 2017, Stuttgart, Germany.

ACM ISBN 978-1-4503-5378-6/17/11.
https://doi.org/10.1145/3152832.3156559

Abstract

Due to the increasing amount of sensors integrated into the environment and worn by the user, a sheer amount of context-sensitive data become available. While interpreting them with traditional methods (e.g., formulas and simple heuristics) is challenging, the latest machine learning techniques require only a set of labeled data. *TensorFlow* is an open-source library for machine learning which implements a wide range of neural network models. With *TensorFlow Mobile*, researchers and developers can further deploy the trained models on low-end mobile devices for ubiquitous scenarios. This facilitates the model export and offers techniques to optimize the model for a mobile deployment. In this tutorial, we teach attendees two basic steps to a deployment of neural networks on smartphones: Firstly, we will teach how to develop neural network architectures and train them in *TensorFlow*. Secondly, we show the process to run the trained models on a mobile phone using *TensorFlow Mobile*.

Author Keywords

Machine learning; classification; supervised learning; tensorflow; ubiquitous computing; mobile device.

ACM Classification Keywords

H.5.m [Information interfaces and presentation (e.g., HCI)]: Miscellaneous

Introduction

The recent wave of machine learning demonstrated an impressive performance for a wide range of tasks. This includes playing Go[1], playing Atari games [11], classifying images[2], or to determine the location where an image was taken [13]. In contrast to the previous waves of machine learning, we now have enough processing power for the training process by using graphic cards as the main processing unit. This enables to deploy new network architectures and training algorithms to achieve more efficient and powerful models. Training and using advanced machine learning models recently became much easier due to a variety of open libraries, including Torch, Theano, and TensorFlow. These libraries are not only used and developed by researchers from academia and industry but are also accessible for practitioners.

The human-computer interaction (HCI) community used machine learning for a very wide range of use cases. Amongst others, this includes classification of pro-eating disorder [1], authentication [2], lifelogging [5], workout trainer [12], and chatbots [14]. The organizers of this tutorial used machine learning techniques and deployed the models on mobile devices for the following three domains. This includes (1) extending the input vocabulary on mobile devices by using the raw data of the touchscreen to estimate the pitch and yaw angle of the finger [10], (2) using a novel smartphone prototype [7] for a wide range of use cases such as grip detection and input pattern recognition [6] and (3) reducing touchscreen latency on mobile devices [3, 4]. We further explored the use of wearable sensors to further decrease the latency of touchscreens [8].

With increasing processing power, it became possible to train increasingly complex machine learning models. In the recent wave of machine learning, processing power is mainly needed during training. A unique feature of *TensorFlow* is the possibility to reduce the size of a trained model and compile it for deployment. In particular, it is possible to run models efficiently on mobile and ubiquitous devices. With processor manufacturers recently starting to optimize their processors for machine learning (e.g., Snapdragon 835), the performance of the models can be improved significantly[3]. This benefits especially systems deployed in ubiquitous environments that have low processing power. This makes *TensorFlow* particularly exciting for researchers in the MUM community as it enables to use powerful machine learning models directly on end users devices and ubiquitous systems.

In this tutorial, attendees will learn the basics to develop neural networks and train them using *TensorFlow*. Further, we will show how to port the trained model on to a mobile phone using the model size reduction features of *TensorFlow Mobile*. *TensorFlow* is a graph-based open source library for a wide range of machine learning algorithms. The graph structure enables *TensorFlow* to move the data between CPU's and GPU's to efficiently manipulate them. When using GPUs, *TensorFlow* relies on CUDA[4] and cuDNN[5]. While *TensorFlow* is mostly known for its Neural Network abilities, it is also possible to train other models, such as K-Nearest-Neighbor and Linear Models.

[1] David Silver and Demis Hassabis on AlphaGo: Mastering the ancient game of Go with Machine Learning https://research.googleblog.com/2016/01/alphago-mastering-ancient-game-of-go.html

[2] Blog post Andrej Karpathy about what I learned from competing against a ConvNet on ImageNet http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/

[3] https://www.qualcomm.com/news/onq/2017/08/16/we-are-making-device-ai-ubiquitous

[4] https://developer.nvidia.com/cuda-zone

[5] https://developer.nvidia.com/cudnn

Covered Topics

We will teach how to train a model using *TensorFlow* version 1.3 or later using Python 3.6. We will further focus on all necessary steps to use a trained model within an Android application. This tutorial covers neural networks for classification tasks. Hence, the focus lies primarily on supervised learning which enables the models to be used for novel interaction techniques and data interpretation as shown in previous HCI work. For each topic, we provide multiple exercises that attendees will solve in group work and with the support of the instructors. Additionally, we will discuss open questions with attendees and how machine learning can further advance research on mobile devices.

Intended Audience

This tutorial requires knowledge of programming. As *TensorFlow*'s Python API is at present the most complete and easiest to use, we will use Python throughout the course. Thus, basic knowledge of Python and the fundamentals in machine learning is beneficial. Further, knowledge of state of the art machine learning concepts, such as neural networks is helpful to apply the presented concepts.

Materials

Attendees receive accounts to run Python code on our server during the tutorial. We further provide Jupyter notebooks that comprise plug & play examples for a classification task. Participants are provided with the option to download these notebooks (including their changes during the course) on their own machine. These notebooks are designed so that they can be readily modified to run the classification task with the attendee's own data set on their machine for future projects. We further provide instructions, scripts and a demo Android project to run the trained model on an Android device.

Duration	Topic
10 min	Introduction of the agenda, the organizers and the participants.
40 min	Overview of machine learning and recent advances in the field, covering supervised & unsupervised learning, classification & regression, TensorFlow, a typical tool chain, and neural networks.
45 min	Hands-on introduction to Jupyter. Participants train a neural network using provided Jupyter notebooks and explore the effect of different hyperparameters using the MNIST data set [9] for handwritten digit recognition.
15 min	Discussion of performance metrics for classification as well as cost functions.
10 min	Overview of the process to bring TensorFlow models to Android devices.
30 min	Participants port their trained model to Android devices using provided code samples which will result in a handwritten digit keyboard.
20 min	Discussion and Q&A.
10 min	Wrap-up of the tutorial and pointers to further directions.

Table 1: The structure of the tutorial that mixes discussion of basic concepts and hands-on work. The table shows a rough estimate of the duration in minutes of the different parts based on organizers previous experience.

Procedure

The tutorial is designed as a course that guides participants through the process of designing the architecture of a neural network using state of the art tools, training the model, and deploying it on mobile devices (see Table 1). A half-day workshop will be sufficient to give participants first hands-on insights into using machine learning for mobile devices.

The course mixes introductions to the individual aspects and hands-on parts that enable participants to explore the presented concepts themselves. Thereby, participants can directly apply the introduced concepts on their own computers. To reduce friction caused by installing the required toolchain on participants' computers, we will prepare Jupyter notebooks[6] for participants and prepare accounts on our own server.

Further topics that would be discussed in a half-day course include, preparing participants' own data sets, further network architectures, hands-on exploration of regression, and unsupervised learning.

Instructors

Huy Viet Le is a Ph.D. student at the University of Stuttgart, funded by the MWK Baden-Württemberg within the Juniorprofessuren-Programm and the German research foundation (DFG) within the SimTech Cluster of Excellence. His research focuses on improving touch interfaces on mobile devices. Most recently, he is exploring a wide range of novel touch-based interaction techniques driven by machine learning. He is interested in modeling touch interaction to improve how people interact with mobile devices.

Sven Mayer is a Ph.D. student at the University of Stuttgart. He is part of Cluster of Excellence in Simulation Technology, in brief SimTech, funded by the German research foundation (DFG). He is generally interested in all flavors of HCI. Sven's particular interest is in touch interaction and machine learning. His main research interest is modeling of human behavior patterns for interactive system.

Niels Henze is an assistant professor at the University of Stuttgart. He heads the group for socio-cognitive systems within the Institute for Visualization and Interactive Systems as well as the SimTech Cluster of Excellence in Simulation Technology. Niels is generally interested in mobile human-computer interaction and he is particularly interested in using large-scale studies to develop models of human behavior that can improve the interaction with mobile devices.

Acknowledgements: This work is supported by the DFG within the SimTech Cluster of Excellence (EXC 310/2), through project C04 of SFB/Transregio 161, and the MWK Baden-Württemberg within the Juniorprofessuren-Programm.

REFERENCES

1. Stevie Chancellor, Yannis Kalantidis, Jessica A. Pater, Munmun De Choudhury, and David A. Shamma. 2017. Multimodal Classification of Moderated Online Pro-Eating Disorder Content. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 3213–3226. DOI: http://dx.doi.org/10.1145/3025453.3025985
2. Benjamin Draffin, Jiang Zhu, and Joy Zhang. 2014. KeySens: Passive User Authentication through Micro-behavior Modeling of Soft Keyboard Interaction. In *5th International Conference on Mobile Computing, Applications, and Services (MobiCASE 2014)*. Springer International Publishing, Cham, 184–201. DOI: http://dx.doi.org/10.1007/978-3-319-05452-0_14

[6] http://jupyter.org/

3. Niels Henze, Markus Funk, and Alireza Sahami Shirazi. 2016. Software-reduced Touchscreen Latency. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '16)*. ACM, New York, NY, USA, 434–441. DOI: http://dx.doi.org/10.1145/2935334.2935381

4. Niels Henze, Huy Viet Le, Sven Mayer, and Valentin Schwind. 2017. Improving Software-Reduced Touchscreen Latency. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct (MobileHCI '17)*. ACM, New York, NY, USA. DOI: http://dx.doi.org/10.1145/3098279.3122150

5. Mohammed Korayem, Robert Templeman, Dennis Chen, David Crandall, and Apu Kapadia. 2016. Enhancing Lifelogging Privacy by Detecting Screens. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 4309–4314. DOI: http://dx.doi.org/10.1145/2858036.2858417

6. Huy Viet Le, Sven Mayer, Patrick Bader, Frank Bastian, and Niels Henze. 2017b. Interaction Methods and Use Cases for a Full-Touch Sensing Smartphone. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. ACM, New York, NY, USA, 2730–2737. DOI:http://dx.doi.org/10.1145/3027063.3053196

7. Huy Viet Le, Sven Mayer, Patrick Bader, and Niels Henze. 2017a. A Smartphone Prototype for Touch Interaction on the Whole Device Surface. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '17)*. ACM, New York, NY, USA, Article 100, 8 pages. DOI: http://dx.doi.org/10.1145/3098279.3122143

8. Huy Viet Le, Valentin Schwind, Philipp Göttlich, and Niels Henze. 2017c. PredicTouch: A System to Reduce Touchscreen Latency using Neural Networks and Inertial Measurement Units. In *Proceedings of the 2017 International Conference on Interactive Surfaces and Spaces (ISS'17)*, Vol. 17. ACM, New York, NY, USA. DOI:http://dx.doi.org/10.1145/3132272.3134138

9. Yann LeCun and Corinna Cortes. 2010. MNIST handwritten digit database. (2010). http://yann.lecun.com/exdb/mnist/

10. Sven Mayer, Huy Viet Le, and Niels Henze. 2017. Estimating the Finger Orientation on Capacitive Touchscreens Using Convolutional Neural Networks. In *Proceedings of the 2017 International Conference on Interactive Surfaces and Spaces (ISS'17)*, Vol. 17. ACM, New York, NY, USA. DOI: http://dx.doi.org/10.1145/3132272.3134130

11. Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. 2013. Playing Atari with Deep Reinforcement Learning. *arXiv preprint arXiv:1312.5602* (2013). http://arxiv.org/abs/1312.5602

12. Jaehyun Park. 2016. Classifying Weight Training Workouts with Deep Convolutional Neural Networks: A Precedent Study. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct (MobileHCI '16)*. ACM, New York, NY, USA, 854–858. DOI:http://dx.doi.org/10.1145/2957265.2961861

13. Tobias Weyand, Ilya Kostrikov, and James Philbin. 2016. PlaNet - Photo Geolocation with Convolutional Neural Networks. In *14th European Conference on Computer Vision (ECCV'16)*. Springer International Publishing, Cham, 37–55. DOI: http://dx.doi.org/10.1007/978-3-319-46484-8_3

14. Anbang Xu, Zhe Liu, Yufan Guo, Vibha Sinha, and Rama Akkiraju. 2017. A New Chatbot for Customer Service on Social Media. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 3506–3510. DOI: http://dx.doi.org/10.1145/3025453.3025496