Privacy-Preserving Machine Learning Based Data Analytics on Edge Devices

Jianxin Zhao, Richard Mortier, Jon Crowcroft, Liang Wang
The Computer Laboratory, University of Cambridge
firstname.lastname@cl.cam.ac.uk

ABSTRACT

Emerging Machine Learning (ML) techniques, such as Deep Neural Network, are widely used in today's applications and services. However, with social awareness of privacy and personal data rapidly rising, it becomes a pressing and challenging societal issue to both keep personal data private and benefit from the data analytics power of ML techniques at the same time. In this paper, we argue that to avoid those costs, reduce latency in data processing, and minimise the raw data revealed to service providers, many future AI and ML services could be deployed on users' devices at the Internet edge rather than putting everything on the cloud. Moving ML-based data analytics from cloud to edge devices brings a series of challenges. We make three contributions in this paper. First, besides the widely discussed resource limitation on edge devices, we further identify two other challenges that are not yet recognised in existing literature: lack of suitable models for users, and difficulties in deploying services for users. Second, we present preliminary work of the first systematic solution, i.e. Zoo, to fully support the construction, composing, and deployment of ML models on edge and local devices. Third, in the deployment example, ML service are proved to be easy to compose and deploy with Zoo. Evaluation shows its superior performance compared with state-of-art deep learning platforms and Google ML services.

CCS CONCEPTS

• **Security and privacy** → **Privacy protections**; • **Human-centered computing** → *Ubiquitous and mobile computing*; • **Computing methodologies** → *Machine learning*; • **Software and its engineering** → Software notations and tools;

KEYWORDS

privacy, edge computing, machine learning

ACM Reference Format:
Jianxin Zhao, Richard Mortier, Jon Crowcroft, Liang Wang. 2018. Privacy-Preserving Machine Learning Based Data Analytics on Edge Devices. In *2018 AAAI/ACM Conference on AI, Ethics, and Society (AIES '18), February 2–3, 2018, New Orleans, LA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3278721.3278778

AIES '18, February 2–3, 2018, New Orleans, LA, USA

ACM ISBN 978-1-4503-6012-8/18/02…$15.00
https://doi.org/10.1145/3278721.3278778

1 INTRODUCTION

Machine Learning (ML) techniques have begun to dominate data analytics applications and services. Recommendation systems are the driving force of online service providers such as Amazon, Netflix and Spotify. Finance analytics has quickly adopted ML to harness large volume of data in such areas as fraud detection, risk-management, and compliance. Deep Neural Network (DNN) is the technology behind voice-based personal assistance [28], self-driving cars [3], automatic image processing [11], *etc.* By deploying ML technologies to cloud computing infrastructures, they are benefiting numerous aspects of our daily life.

However, the surge of ML is accompanied by a public concern of personal data privacy. The basic business model of ML-based data analytics is a closed circle: with more data, more accurate model can be trained, which means more useful services and more users, and they finally lead to more data being collected. At the same time, people are increasingly aware of the data privacy issue. Ubiquity of sensing via mobile and IoT devices has caused a surge in personal data generation and use. They contain our photos, browsing history, and voice records, *etc.*, some of which one might not want to share with data analytics service providers. But these data are also perfect targets for them to collect in order to provide personalised services. So even with legal regulatory frameworks such as EU's General Data Protection Regulation [9], the aforementioned business model cycle is still difficult to break.

Existing solutions that aim at solving this data privacy issue mostly focus on improving the cloud-based services, including making databases more secure [7], hiding trained ML models [1], enable users to choose which part of data to remove before uploading to cloud servers [35], *etc.*

However, we argue that patching existing cloud-based data analytics is not enough. Take the speech recognition application as an example. Current approaches require user to upload audio data to server side for inference, which returns the results back to user via the Internet. Aside from the data privacy issue, this workflow also leads to network communication cost and response latency, especially when users are limited by time or bandwidth budgets. In this paper, "users" could be individuals who are looking for a one-off speech recognition tool, business such as banks that seeks to keep record of internal conferences, or researchers who want to preprocessing their large audio dataset before some experiments. In all these cases, the priority is data privacy, integrity or service response time.

One solution is to deploy ML services on edge devices. Moving services from cloud to users' edge devices can keep the data private, and effectively reduce the communication cost and response latency . Some research begin to emerge that aims to solve accompanied challenges. They recognise that mobile devices cannot afford to

support most of today's intelligent systems because of the large amount of computation resource and memory required. As a result, many current end-side services only support simple ML models. These solutions mostly focus on reducing model size [4] [13] [16].

In this paper, we further identify two other main challenges to do ML based data analytics on edge devices. The first one is the lack of suitable models for users. Training a model requires large datasets and rich computing resources, which are often not available to most users. That's one of the reasons that they are bounded to use the models/services provided by large companies. Towards this end we propose the idea "composable services", where the users can construct new services based on existing ones and contribute them to the community. The second new challenge is the deployment of services. Users may lack required knowledge of ML to understand how a model works and how to properly deploy it to any local devices. We present the design of a novel system Zoo to support the construction, compose, and easy deployment of ML models on edge and local devices. In the deployment example, ML services are proved to be easy to compose and deploy with Zoo. Our evaluations show its performance compared with state-of-art deep learning platforms and Google ML services.

2 RELATED WORK

2.1 Social Awareness for Privacy

Ever since Edward Snowden revealed the secret large scale government-level surveillance programmes, social awareness of privacy and personal data is quickly arising. The Internet Trends 2016 report [19] points out that, according to its survey, 45% respondents feel more worried about their online privacy than one year ago, and 74% have limited their online activity because of privacy concerns.

Many online service providers, while collecting large-scale data from users, may be prone to data breaches. Users rely on the promises from big companies to keep their data private and safe [12]. However, these promises are not always infallible. Yahoo, Tumblr, and Ashley Madsion (an online dating service for married people) are only a tiny bit of the current frequent enterprise data breach cases [5]. As a result, billions of users' private information is endangered. All too often, these leakages lead to a series of social and ethical aftermaths. As part of the effort to restrain this trend, Regulations such as EU's General Data Protection Regulation [9] are implemented to "give citizens back control over of their personal data, and to simplify the regulatory environment for business".

2.2 Privacy-preserving Analytics on Cloud

Many popular data analytics are conducted on cloud, therefore its data privacy issue has attracted a lot of research interests. Making databases private and secure is one of the solutions. [2] introduces a cloud-based framework that enables secure management and analysis of large, and potentially sensitive, datasets. It claims to ensure secure data storage that can only be accessed by authorised users. [15] has developed an ontology so that big data analytics consumers can write data privacy policies using formal policy languages and build automated systems for compliance validation. [7] presents a short review of existing research efforts along this line.

Hiding data alone is not enough. The model itself can also reveal private information. [10] has shown that, with access to the face recognition ML model, the authors can use that to recover recognisable images. [1] develops a method that can prevent these kinds of model inversion attacks against a strong adversary who has full knowledge of the training mechanism and access to the model parameters. Similarly, [23] proposes a "teacher-student" approach that is based on the knowledge aggregation and transfer technique, so as to hide the models trained on sensitive data in a black box.

Instead of hiding data or models, some research suggest user to choose which part of data to upload. [35] proposes a mechanism to allow users to clean their data before uploading them to process. It allows for prediction of the desired information, while hiding confidential information that client want to keep private. RAPPOR [8] enables collecting statistics data from end-user in privacy-preserving crowdsourcing.

However, all of the aforementioned work focus on the traditional cloud side solution. Users' data are still collected to central server for processing, which are prone to issues such as increased service response latency, communication cost, and single point failure.

2.3 Data Analytics on Edge Devices

Computing on edge and mobile devices has gained rapid growth. Recently HUAWEI has identified speed and responsiveness of native AI processing on mobile devices as the key to a new era in smartphone innovation [14]. Applications in this field are of great interest to academia and industry. [20] provides a collection of software components that enable individual data subjects to manage, log and audit access to their data by other parties. [22] presents an environment monitoring smartphone app that allow users to takes photos of the outdoor to recognise air quality. Intrusion Detection System (IDS) is important in securing computer networks. Using ML for IDS is especially suitable in stopping attacks that do not have know signatures. [27] designs specific hardware for ML based IDS tasks. It achieves high detection accuracy and low energy consumption.

More and more advanced smart services are being pushed to edge devices. Intel's Movidius Neural Compute Stick [21] is a tiny deep learning device that one can use to accelerate AI programming and DNN inference application deployment at the edge. Kvasir [34] is a semantic recommendation system built on top of latent semantic analysis and other state-of-the-art data analytics technologies. Specifically, it seamlessly integrates an automated and proactive content provision service into web browser on users' end.

Many challenges arise when moving ML analytics from cloud to edge devices. One is that compared with resource-rich computing clusters, edge and mobile devices only have quite limited computation power and working memory. To accommodate heavy ML computation on edge devices, one solution is to train suitable small models to do inference on mobile devices [6]. This method leads to unsatisfactory accuracy and user experience.

Some techniques are recently proposed to enhance this method. To reduce the memory and disk usage of speech recognition application, [18] uses a compressed n-gram language model to do on-the-fly model rescoring. [4] presents HashNets, a network architecture that can reduce the redundancy of neural network models to

decrease model sizes, while keeping little impact on prediction accuracy. [13] from Google reduces model size by a different technique: factorisation of convolution operation.

One of our previous research work [25] explores the method of training personalised model on local devices from an initial shared model. Instead of moving data from user to cloud, our method provides for model training and inference in a system where computation is moved to the data. Specifically, we take an initial model learnt from a small set of users and retrain it locally using data from a single user. It is proved to both be robust against adversarial attacks and can improve accuracy.

Besides deploying data analytics purely on cloud or edge, putting computation dynamically on both ends or somewhere in between is also a popular trend. In [16], the authors propose partitioning a DNN into two parts, half on edge devices and the other half on cloud, and then develop the *Neurosurgeon* system, aiming at reducing total latency and energy consumption. A key observation is that, in a DNN, output size of each node decreases from front-end to back-end, while the change of computation latency is the opposite. [32] recognises that countless network services reside in ISP networks instead of on the cloud to provide low-latency access. It further investigates computation congestion control strategies to effectively distribute service load within a neighbourhood.

3 SYSTEM DESIGN AND IMPLEMENTATION

The basic idea of *Composable Services* is users should not have to construct new ML services every time new application requirements arise. In fact, many services can be composed from basic ML services: image recognition, speech-to-text, recommendation, etc. One example is that a service to recognise multiple objects from one image can be composed from two services: image segmentation (which partitions a digital image into multiple segments/regions) and image recognition on each segment. The Zoo system aims at providing user-centric, ML-based services that enables service pulling, composing, sharing, and compatibility checking. The system architecture of Zoo is shown in Figure 1.

We define that a ML service consists of two parts: *development* and *deployment*. *Development* is the design of interaction workflow and the computational functions of different ML services. We provide primitives to construct complex services from simple ones. For example, one of the most important primitives is sequential connection, where the output of one service is used as input of another service.

In Figure 1, a user first designs the target service on computer C (Step ①) in a configuration file, and executes the compose script with Zoo. The square, hexagon, *etc.* here represent different existing ML services. They are connected with the sequential connection primitive in the new structure design. The local repository is first checked to see if models of these required services are cached locally. If not, they will be first pulled from remote repositories. In our current implementation, all published models are stored and accessed from Github Gist (represented by Server A), but this approach could be extended to support pulling services from other peer devices such as machine B in the figure (Step ②).

Deployment deals with service interface and location definition. It is separated from the logic of service construction. Users can move services from being local to remote and vice versa, without changing the structure of constructed service. Deployment is not limited to edge devices (machine E), but can also be on cloud servers (server D), or a hybrid of both cases, to minimise the data revealed to the cloud and the associated communication costs (Step ③). Thus by this design a data analytics service can easily distributed to multiple devices. Finally, user can contribute newly created services to the model repositories so that it can be accessed by others (Step ④).

The Zoo system is implemented on Owl [31], an open-source numerical computing system in OCaml language. The reason we choose Owl to support the implementation of Zoo is some of its nice features. Owl provides a full stack support for numerical methods, scientific computing, and advanced data analytics on OCaml. Built on the core data structure of matrix and n-dimensional array, Owl supports a comprehensive set of classic analytics such as math functions, statistics, linear algebra, as well as advanced analytics techniques, namely optimisation, algorithmic differentiation, and regression. On top of them, Owl provides Neural Network and Natural Language Processing modules. Zoo relies on these modules to construct basic models. It has static type checking, and Owl's ML modules have shown great expressiveness and code flexibility. Moreover, Owl can use the parallel and distributed engine at lower level to support distributed numerical computing and data analytics. It supports different protocols and multiple synchronisation techniques [33], which is a crucial building block for collaboration between network nodes, and thus important for scaling up data analytics in heterogeneous edge networks.

4 DEPLOYMENT EXAMPLE

Next we show an example to use Zoo to deploy an image classification services based on InceptionV3 neural network architecture. The service is deployed on a representative resource-constrained personal device: a Raspberry Pi 3 Model B [24].

InceptionV3 [30] is one of Google's latest effort to do image recognition. It is trained for the ImageNet Large Visual Recognition Challenge [26]. This is a standard task in computer vision, where models try to classify an image into 1000 classes, like "Zebra" or "Dishwasher". Compared with previous DNN models, InceptionV3 has one of the most complex network architectures in computer vision.

The service is composed of two services: an InceptionV3 network that output a vector representing the recognised image class, and an decoding service for ImageNet that translates the previous vector to human-readable format. These two services are sequentially connected. By using Zoo, we can deploy this new services to local Raspberry Pi devices with only one line of command. With minor change of configuration, the service can also be deployed to our server on Digital Ocean, a cloud infrastructure provider. Currently we have used this localised deployment instance for internal automatic image tagging tasks. For the latter instance, we use it to construct an online demo of this service[1].

[1] http://138.68.155.178/

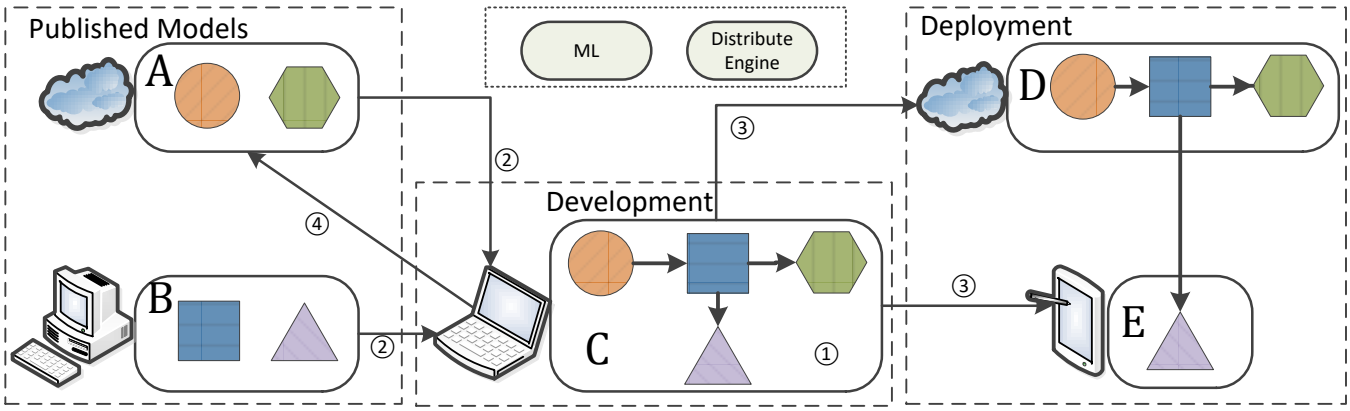

Figure 1: Zoo System Architecture

5 PERFORMANCE EVALUATION

In this section we present preliminary experiment results with Zoo system. We want to answer two questions here. First, we choose to use Owl to base the Zoo system on. To what degree does its expressiveness has shown? Does it bring any performance trade-off? Second, we deploy services on local devices, so what are their performance compared with popular cloud-based analytic solutions such as Google ML API?

Let's start with the first question. As stated before, Owl provides extraordinary expressiveness and flexibility. One example to show this is that using Owl to constructs the InceptionV3 network only takes 150 lines of code (LoC), compared with the 400 or more LoC used by Tensorflow. Another one is that, we insert instrumentation code into Owl to enable collecting forward computation latency of each node (or *layer* according to some literature, *et pass.*) in a neural network when doing inference. Adding this feature only takes 50 LoC in the source code of Owl.

Since the Zoo relies on inference using Owl's Neural Network module, we want to compare the inference time on Owl and the other state-of-art deep learning platforms. We choose three representative DNN models that vary greatly in both architecture complexity and parameter sizes: 1) one small neural network (LeNet-5 [17]) that only consists of 8 nodes and contains about 240KB parameters (each parameter in a model is represented by a 32-bit float number, *et pass.*) for the MNIST handwriting recognition task; 2) a VGG16 [29] model that has a simple architecture with 38 nodes but a large amount of parameters (500MB) for real-world image recognition task; 3) an InceptionV3 model also for image recognition, with less parameters (100MB), but a far more complex architecture (313 nodes). We compare the time it takes for each model to finish its inference task on different platforms: Owl, TensorFlow, and Caffe2. Each measurement is repeated 20 times.

The results are shown in Figure 2. Regardless of great diversities in these models' architectures and sizes, Owl takes less time to do inference than Tensorflow and Caffe2. It means that Owl can achieve both expressiveness and good preformance. The superior performance of Owl on large models is attributed to its efficient math operations.

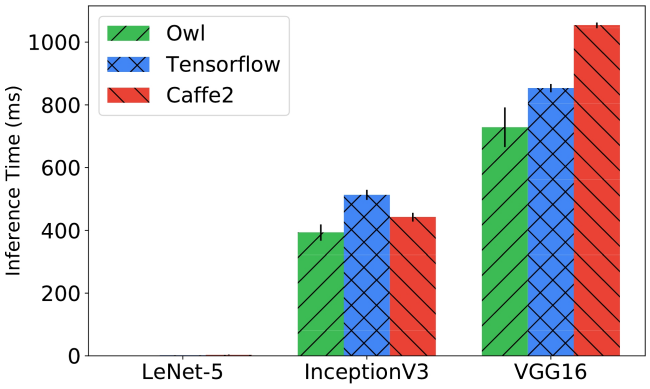

Figure 2: Inference time comparison on different platforms.

Next, we investigate the performance of Zoo system compared with Google ML API. The Google Cloud Vision API [11] encapsulates machine learning models in a REST API. It can classify images into thousands of categories as well as detects individual objects and faces within images, and finds and reads printed words contained within images. Its workflow is simple: a user creates service token on the Google Cloud Platform (GCP), and passes the token and an image to Google's server for processing, and then the processed results will be returned to the user as response in the form of JSON. In this evaluation we compare the image classification development.

We deploy Vision API service on GCP. The network connection bandwidth is 34 Mbps measured on `speedtest.net`. In the previous section we have shown a deployment example of our image classification service on local devices with Zoo. For this evaluation, we deploy this service on a Thinkpad T460s laptop with a Intel Core i5-6200U CPU. First, we have collected 100 animal images as a dataset[2]. These images are of different sizes, ranging from 7KB to 1243KB, to better simulate users' requests in real world. Specifically, we compare the time required for both methods to process different number of images.

The results are shown in Figure 3. It has shown that our Zoo service achieves lower response latency. When the number of input

[2]Available at https://goo.gl/BJqiBD

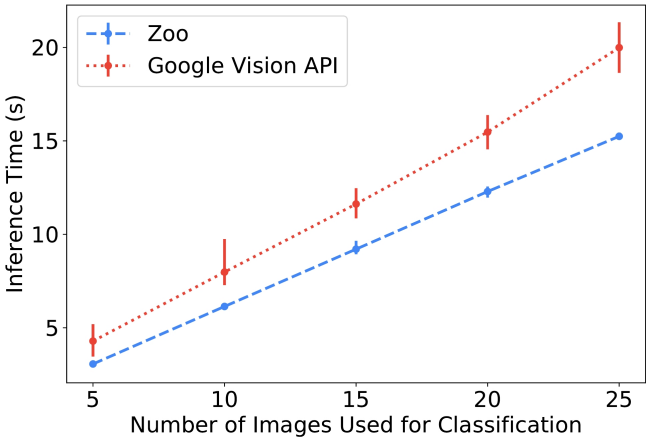

Figure 3: Performance comparison of local image classification services deployed Zoo with Google's Machine Learning Vision service.

images from a user gradually increases from 5 images to 25 images, the response time of the Zoo service increases linearly, which means the process time of each image basically keeps constant despite the size difference of input images, while the increase speed Google service's response time seems to grow with more input images. The measurement at each point is repeated for 10 times. Note that the Zoo service response time keep stable (about 0.6s per image) with very small deviation, and thus predictable. The same cannot be said of Google Vision service. It shows relatively large deviation, and with the change of network connection, the response time could easily fluctuate and become unpredictable.

6 DISCUSSION

In this section we briefly discuss the threat model that is applied in this paper. We assume the companies that provide ML based data analytics services have enough incentives to collect users' data as much as possible. These data are all stored in the companies' servers, and thus face various security and privacy threats. First, the companies can make mistakes and may published or even delete users' data unintentionally. For example, in February this year, Gitlab lost six hours of database data by accident. Soon after, AWS, one of the most popular cloud computing services provider, also reported an accidental data loss cause by misoperation of an authorized team member. Second, as part of the companies' assets, these data may of enormous economic value and thus are often targets of hackers. Not all of the attacks can be successfully evaded. This year a plastic surgery was blackmailed by a gang of hackers, who threatened to release patient list and photographs. Large scale data breaches of companies like Yahoo can have detrimental effect on users' privacy. Last but not the least, some individuals may not be comfortable with the idea that one entity hold their personal information such as sexuality or years of browsing history, even these data are not yet used for any analytics. Moving ML services and data analytics from cloud to the edge devices on users' side can effectively mitigate these threats.

However, we do not claim this approach can solve all security and privacy threats on user data. One factor we do not consider in this paper is that sometimes personal devices are also vulnerable to hackers. For example, in the WannaCry ransomware attack this year, the hackers encrypted data on individual's PC and demanded ransom payment. Addressing this threat often requires users' effort to use strong password, upgrade system frequently, *etc.* Another one is, multiple users may collude to infer users data from the shared model. It is a good topic for further investigation.

7 CONCLUSION

Machine Learning based data analytics have been widely adopted in today's online applications and services. However, with social awareness of privacy and personal data rapidly rising, they also bring about a pressing societal issue: data privacy. Most existing solutions to this issue focus on enhancing the currently popular cloud-based analytics approach, where users' data need to be collected to central server for processing. Besides data privacy, this approach is also prone to issues such as increased service response latency, communication cost, and single point failure.

In this paper, we argue that to avoid those costs, reduce latency in data processing, and minimise the raw data revealed to service providers, many future AI and ML services could be deployed on users' devices at the Internet edge rather than putting everything on the cloud. This brings many challenges. Besides the widely discussed factor of resource limitation on edge devices, we identify two other challenges that are not yet recognised in existing literature: lack of suitable models for users, and difficulties in deploying services for users. We present the design of our Zoo system to support the construction, compose, and easy deployment of ML models on edge and local devices. Then we show how services can be composed and deployed with Zoo, and its superior performance compared with existing deep learning platform and cloud-based services. We believe this area of research is only just beginning to gain momentum.

Acknowledgements This work is funded in part by the EPSRC Databox project (EP/N028260/1), NaaS (EP/K031724/2) and Contrive (EP/N028422/1).

REFERENCES

[1] Martín Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 308–318.

[2] Yadu N Babuji, Kyle Chard, Aaron Gerow, and Eamon Duede. 2016. Cloud Kotta: Enabling secure and scalable data analytics in the cloud. In *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 302–310.

[3] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. 2016. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316* (2016).

[4] Wenlin Chen, James Wilson, Stephen Tyree, Kilian Weinberger, and Yixin Chen. 2015. Compressing neural networks with the hashing trick. In *International Conference on Machine Learning*. 2285–2294.

[5] Long Cheng, Fang Liu, and Danfeng Daphne Yao. 2017. Enterprise data breach: causes, challenges, prevention, and future directions. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 7, 5 (2017).

[6] Byung-Gon Chun and Petros Maniatis. 2009. Augmented smartphone applications through clone cloud execution.. In *HotOS*, Vol. 9. 8–11.

[7] Alfredo Cuzzocrea, Carlo Mastroianni, and Giorgio Mario Grasso. 2016. Private databases on the cloud: Models, issues and research perspectives. In *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 3656–3661.

[8] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the*

2014 ACM SIGSAC conference on computer and communications security. ACM, 1054–1067.

[9] European-Commission. 2016. Protection of Personal Data. http://ec.europa.eu/justice/data-protection/reform/index_en.htm. Accessed: 2017-10-07.

[10] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1322–1333.

[11] Google. 2017. Google Cloud Vision API. https://cloud.google.com/vision. Accessed Nov. 11, 2017.

[12] Google. 2017. Google Privacy. https://privacy.google.com. Accessed Nov. 11, 2017.

[13] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *CoRR* abs/1704.04861 (2017). arXiv:1704.04861

[14] Huawei-News. 2017. HUAWEI Reveals the Future of Mobile AI at IFA 2017. http://consumer.huawei.com/en/press/news/2017/ifa2017-kirin970/. [Online; accessed 10-Nov-2017].

[15] Karuna P Joshi, Aditi Gupta, Sudip Mittal, Claudia Pearce, Anupam Joshi, and Tim Finin. 2016. Semantic approach to automating management of big data privacy policies. In *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 482–491.

[16] Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor Mudge, Jason Mars, and Lingjia Tang. 2017. Neurosurgeon: Collaborative Intelligence Between the Cloud and Mobile Edge. In *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems*. ACM, 615–629.

[17] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[18] Xin Lei, Andrew W Senior, Alexander Gruenstein, and Jeffrey Sorensen. 2013. Accurate and compact large vocabulary speech recognition on mobile devices.. In *Interspeech*, Vol. 1.

[19] Mary Meeker. 2016. Internet Trends 2016 Report. http://www.kpcb.com/internet-trends. Accessed: 2017-10-01.

[20] Richard Mortier, Jianxin Zhao, Jon Crowcroft, Liang Wang, Qi Li, Hamed Haddadi, Yousef Amar, Andy Crabtree, James Colley, Tom Lodge, et al. 2016. Personal Data Management with the Databox: What's Inside the Box?. In *Proceedings of the 2016 ACM Workshop on Cloud-Assisted Networking*. ACM, 49–54.

[21] Movidius. 2017. Movidius Neural Compute Stick. https://developer.movidius.com/. Accessed Nov. 11, 2017.

[22] Zhengxiang Pan, Han Yu, Chunyan Miao, and Cyril Leung. 2017. Crowdsensing Air Quality with Camera-Enabled Mobile Devices.. In *AAAI*. 4728–4733.

[23] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian Goodfellow, and Kunal Talwar. 2016. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755* (2016).

[24] RaspberryPi. 2016. Raspberry Pi. https://www.raspberrypi.org/products/raspberry-pi-3-model-b/. Accessed February 15, 2017.

[25] Sandra Servia Rodríguez, Liang Wang, Jianxin R. Zhao, Richard Mortier, and Hamed Haddadi. 2018. Privacy-preserving Personal Model Training. *Internet-of-Things Design and Implementation (IoTDI), The 3rd ACM/IEEE International Conference on* (2018).

[26] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)* 115, 3 (2015), 211–252. https://doi.org/10.1007/s11263-015-0816-y

[27] Rajesh Sankaran and Ricado A Calix. 2016. On the feasibility of an embedded machine learning processor for intrusion detection. In *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 1082–1089.

[28] Mike Schuster. 2010. Speech recognition for mobile devices at Google. In *Pacific Rim International Conference on Artificial Intelligence*. Springer, 8–10.

[29] Karen Simonyan and Andrew Zisserman. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR* abs/1409.1556 (2014).

[30] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the Inception Architecture for Computer Vision. *CoRR* abs/1512.00567 (2015).

[31] Liang Wang. 2017. Owl: A General-Purpose Numerical Library in OCaml. *CoRR* abs/1707.09616 (2017). http://arxiv.org/abs/1707.09616

[32] Liang Wang, Mário Almeida, Jeremy Blackburn, and Jon Crowcroft. 2016. C3PO: Computation Congestion Control (PrOactive).. In *ICN*. 231–236.

[33] L. Wang, B. Catterall, and R. Mortier. 2017. Probabilistic Synchronous Parallel. *ArXiv e-prints* (2017).

[34] Liang Wang, Sotiris Tasoulis, Teemu Roos, and Jussi Kangasharju. 2016. Kvasir: Scalable provision of semantically relevant web content on big data framework. *IEEE Transactions on Big Data* 2, 3 (2016), 219–233.

[35] Ke Xu, Tongyi Cao, Swair Shah, Crystal Maung, and Haim Schweitzer. 2017. Cleaning the Null Space: A Privacy Mechanism for Predictors.. In *AAAI*. 2789–2795.