www.ijecs.in

International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 6 Issue 4 April 2017, Page No. 21081-21085
Index Copernicus value (2015): 58.10 DOI: 10.18535/ijecs/v6i4.47

Privacy Preservation in Dynamic Bigdata for Verification and Correctness

Guthula Anitha Rani, Prof P Suresh Varma
Adikavi Nannya University,Computer Science and Engineering ,
Rajamahendravaram,Andra Pradesh,India
anitharani574@gmail.com
Department of Computer Science and Engineering, Adikavi Nannya University
Rajamahendravaram,Andra Pradesh,India
vermaps@yahoo.com

Abstract: ***Mobile devices makes the mobile crowd sourcing is possible, since the mobile devices are everywhere in a tourism(theme)/network if a requester(travel agency) crowd, sources the data from worker(tourist). However data collection aggregation and data analysis have become challenging problems for requester when the volume of data is huge which is categorized as Bigdata. The data analysis includes set operations like intersection, union, and complementation for filtering redundant data and pre-processing raw data. theme is a necessity of data exchange between the worker, requester for better analysis of the interested worker. But workers may not be willing to participate if the privacy of their sensing data and identity are not well preserved in the untrusted cloud. Hence, the proposed work, establishes the usage of cloud to compute the intersect operation between the requester and the workers data. Also preserves the workers identity and accessible data. This paper says that use of cloud to compute set operation for the requester, at the same time workers data privacy and identity privacy are well preserved. And also, the requester can verify the correctness of set operation results on the dataset sourced by workers and send to the cloud. With this batch verification and data update are comparatively increased and reduce computational costs of the system.***

Key Words: Bigdata, Mobile Crowdsourcing, batch verification, Privacy

1. Introduction

Crowdsourcing is defined as the practice of obtaining needed services or content by soliciting contributions from a large group of workers. Recently, with the rapid development of mobile Internet and mobile social networking techniques, the scope of crowd problem-solving system using mobile devices has been broadened and the traditional Internet Crowdsourcing is evolving into a new paradigm, i.e., Mobile crowd sourcing (MCS), Which facilitates the increasing number of mobile device users to participate crowdsourcing tasks.

Mobile crowdsourcing enables a task owner to obtain data from a large number of smartphone users, and further perform data analysis on the aggregated data. The task owner is also known as the requester, while the participating smartphone users are mobile workers who will collect and/or sense the data for the requester. With the development of the low cost sensing devices, many sensors have been embedded on mobile devices, such as GPS, accelerator, gyroscope, digital compass, temperature sensors, etc. More sensors measuring humidity, air quality, chemical, barometer, and biomedical information can be Equipped into smart phones or connected via wireless technologies. These affordable sensor-rich smart phones make them capable of sensing the environment around people and people’s physiological data as well. In mobile crowd sourcing, requester can make use of the data crowd sourced from mobile worker’s to achieve certain tasks. For example, transportation management bureau can utilize the speed data reported from the commuters to analysis the traffic condition. Obviously, mobile crowd sourcing as many advantages : first, the ubiquitous smart phone users cover a large geographic area, which makes data and information divers and rich; second, the requester does ‘not need to deploy specific sensor networks or employees to collect the targeted data; third, workers can receive rewards such as reputation and revenue from the crowd sourcing participation.

Set operations or often used in data processing. For example, travel agencies want to know the most popular places that the tourists have visited during holidays. Here, the data from worker (tourist) will be a set, and thus the requester (travel agency) needs to find the intersection of all set. Set union may be used to merge different data bases collected from different database owners. Set difference is use full one requester wants to find the unique feature of one database compare to another. When the number of workers is very large, the requester requires huge amount of storage space for storing the crowd sourcing Bigdata even if each worker’s data is relatively small.as a result, a storage limited requester is not able to handle the above. Taking step for that, even if the requester can store all collected” Bigdata” , the data processing and analysis maybe another stumbling block when he /she lacks computations capability.

An untrusted cloud may return a wrong set operation result to the requester. When computing set operations, the cloud may discard some data sets to reduce expense. Facing these challenges, we propose a verifiable set operation in Bigdata for could assist mobile crowd sourcing. Our solution leverages the cloud to release computation burden of the requester while preventing all the above security and privacy issues. With our scheme worker’s data and identity privacy are well preserved. Meanwhile, the requester can verify the correctness of the result retrieved from the cloud. We also extend our scheme to support data preprocessing, batch verification, and efficient data update.

1. Related work

2.1. Private Set Intersection:

Many workers have been done to achieve private set intersection(PSI). PSI enables two parties to compute the intersection with private input and only the intersection is known to each party. The first protocol for PSI is proposed in Kissner et al. use polynomial representations to solve set operations between two parties, and utilize paillier Crypto system to protect the privacy of polynomial when trusted third party is not available. Private set intersection with linear complexity is proposed. Dong et al. make use of new variant of bloom filter to achieve efficient PSI. bloom filter and homomorphic encryption are used to achieved outsourced private set intersection. All of these works can achieve private set intersection, however, none of them offers verifiability of the result. Thus,none of them can be applied in our work directly.

2.2 Verifiable Computation:

Verifiable computation was introduced and formalized by Gennaro et al. which enables a resource-limited client to outsource the computation of a function to one or more workers. The workers return the result of function evaluation. The client should be able to efficiently verify the correctness of the results. After that, many workers have been done to achieve verifiable computation. In they propose the first practical verifiable computation scheme for high degree polynomial function. Fiore et al. propose a solution for publicly verifiable computation of large polynomials and matrix computations, where anyone can verify the correctness of the results. Popamanthou et al. study the problem of cryptographically checking the correctness of outsourced set operations performed by an untrusted server, and the sets are dynamic. However, all of them are designed for verifiable computation over plaintexts where data privacy is not considered. Verifiable computation for encrypted data is provided. This protocol allows multiple clients to upload their datasets and obtain the intersection from the cloud. Guo et al. propose a verifiable computation over encrypted data for mHealth systems, where a paitent can ask the cloud to evaluate a polynomial over his encrypted personal health record, and verify the correctness of the evaluation result. Although can achieve verifiable computation over encrypted data, they are all two party architecture, which is not suitable for our scenario. scenario.

3. Frame work

System Architecture:

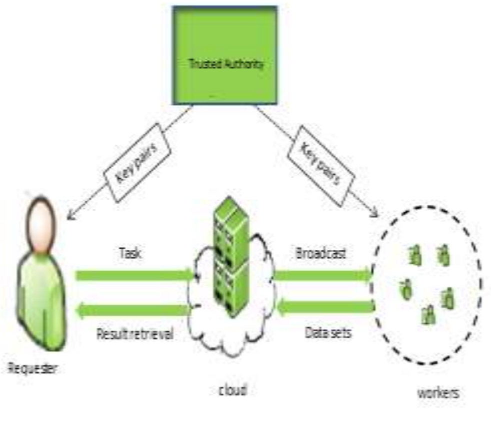

Fig.1 System Architecture

3.3 System initialization

The system consists of Trust Authority (TA), Cloud, Requester, Workers. TA is responsible for initializing the whole system which includes registering workers, requesters and the cloud, generating public parameters, and distributing keys, and maintaining the system. TA may be offline unless a dispute arises. The requester wants to obtain the intersection set of the workers' data sets. However, due to his/her limitation on the storage and computation capability, the requester will delegate storage and most of the computation tasks to the cloud. The cloud receives the delegation requests from the requester and the encrypted data sets from mobile workers, then it computes the intersection set for the requester. The cloud also needs to provide some proof information to prove the correctness of the result.

3.2 Crowd sourcing

Under Crowdsourcing, users cover a large geographic area, which makes the data and information diverse and rich; second, the requester does not need to deploy specific sensor networks or employees to collect the targeted data. User uploaded data are stored in the cloud after encryption.

3.3 Data Encryption

The data privacy is preserved through encryption. The requester will get the computation result from the cloud together with a proof information. Every worker W generates his data set D, and encrypts it with kp. The data will be signed with ring signature before sending to the cloud. After receiving encrypted data sets from all workers, the cloud verifies the authenticity of each of them, and computes the intersection set based on the encrypted data sets. Then the cloud sends the result together with its corresponding proof information to the requester. Finally, the requester decrypts the result and checks its correctness.

3.4 Set operation and verification

Set operation is performed when the workers are uploading data to cloud. Supposing the range limit set defined by the requester is SR, which means all valid data should be within SR. Worker Wx has data set Dx. There are four possible relationships between SR and Si. When the requester delegates the set intersection computation to the cloud, the cloud needs to excludes set Dx if the relationship between Si and SR, which means Dx contains at least an element that's not in SR

3.5 Data Dynamic and verification

To reduce the cost on processing the operation on collected data, we need to carefully exam the reported data. Normally, the requester has a specific range requirements on the data set. The requester may determine that only sets of a specific range of tourist sites are eligible for the computation of intersection. This is especially useful for improving efficiency and accuracy in big data analysis, because it will greatly reduce the unnecessary raw data for data processing. The requester needs

DOI: 10.18535/ijecs/v6i4.47

to compute a hashing set, based on relationships the cloud finds out all sets Si which satisfies requester conditions.

4. Implementation

4.1 Design

As shown in Fig. 2, our Design mainly consists of four entities, the mobile workers (W), the requester (R), the cloud (C), and the trusted authority (TA).

- ➢ *Trust Authority* (TA): TA is in authority for make ready the complete system which contains registering workers, requesters and the cloud producing public parameters, and allocating keys, and preserving the system. TA may be disconnected without a dispute arises.
- ➢ *Cloud*: The cloud receives the assignment requests from the requester and the encrypted data sets from mobile workers, then it calculates the intersection set for the requester. The cloud also wants to provide some proof information to verify the rightness of the result.
- ➢ *Requester*: The requester needs to get the intersection set of the workers' data sets. However, due to his/her control on the storage and computation capability, the requester will representative storage and most of the computation tasks to the cloud
- ➢ *Mobile Workers*: Mobile workers refer to those who have smartphones and are willing to contribute data to the requester's tasks. Each worker generates her own data set, and encrypts it before sending it to the cloud.

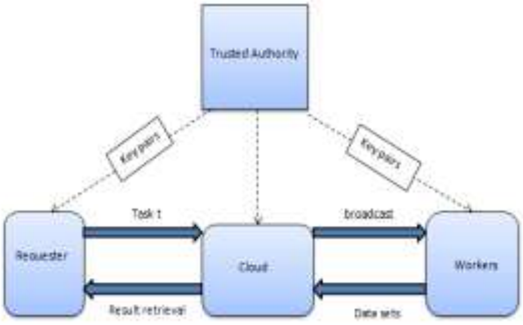

Fig.2 : Design

- ➢ Security Model

In our security model, the TA is fully trusted and will not be breached by any adversary. The security requirements for other entities are given below.

- *Mobile Workers*: In our scheme, a worker's data set should be kept confidential from other workers and the cloud.
- *Requester*: The security requirement for the requester is that he should be able to verify the correctness of the computation result received from the cloud.
- *Cloud*: In our model, the cloud is curious but honest. It should not be able to know workers' data sets or the intersection set.

We have three main objectives for our privacy-preserving verifiable computation of set intersection for mobile crowdsourcing. First, the cloud can compute the intersection set of the workers' data sets without knowing the content and source of the data sets, thus workers' data privacy and identity privacy are well preserved. Second, the requester can verify the correctness of the intersection set retrieved from the cloud. Third, to better adapt the privacy requirements for the collected big data, the proposed scheme should be scalable and efficient for processing huge volume of reported data

Notation used in our implementation

Notation	Description
M	Total number of mobile workers
W_x	Worker
p	Prime order of group G,G_T ,and z_p
g	Generator of group G
$(kp_x, kpriv_x)$	Public/private key pair of worker
$(kp, kpriv)$	Public/private key pair of requester
shk	Keyed hash function
D_x	Data set of worker W_x
C_x	Ciphertext set of D_x
H_x	Hashing set of D_x
u	Upper limit of $\|D_x\|$
N_D	Intersection set of all data sets
N_H	Intersection set of all hashing sets
N_C	Set of ciphertexts whose plaintexts are elements in N_D

Table 1

4.2 System Initialization

TA first generates necessary parameters and keys for the system. Then, TA registers all workers, requester and cloud into the system. We present the two steps as follows. Main notations are listed in Table.I.

- General Setup: Given the security parameter k, TA generates the bilinear parameter (p, G,G_T, e, g). Also, a hash function $H_0(): [1,0]^* \rightarrow z_p$ is defined. TA chooses a random value d $\in$ Zp, and computes $g^d, g^{d_2}, \dots\dots, g^{d^u}$ Then TA publishes
{ p, G, , e,$g^d, g^{d_2}, \dots\dots, g^{d^u}$,$H_0()$} .
- Entities Registration: Assume there are m mobile workers in the system$\{w_1, w_2, \dots\dots, w_m\}$. For each worker$W_i$, TA assigns him a public/private key pair ($kp_x, kpriv_x$), where $kpriv_x$ =$a_x \in R$ R Zp and kp_x = g^{a_x} . TA registers the cloud and the requester by sending the private/public key pairs ($kpriv_c, kp_c$) = (a, g^{a_c}) and (kpriv, kp) = (A,g^a) to the cloud and the requester respectively, where a_c and x are random number from Zp. Besides, both requester and workers obtain the encryption key for a private hash function H(H_k) : G → Zp..

In our scheme, the plaintext space is group G, while the data space could be of any type. Therefore, the requester needs to build a mapping table between the data space and the plaintext space for every task T. The mapping table can be built as follows. The requester first defines a data space for the collected data. Then for every element in data space, a new random element in plaintext space G is chosen. The mapping table for the task is public to all. Then he sends T and kp as a task to the cloud. After receiving T and kp from the requester, the cloud broadcast the task to all the workers. When a worker receives the task from the cloud, she generates a data set based on task tag T , and maps every element in data set to element in plaintext space according to the mapping table provided by the requester to get her plaintext set , where ni is the cardinality of , and , y = 1, 2, ... In the following, we assume every worker will map her collected data set to plaintext set automatically, and use data set and plaintext set interchangeably. Then, a worker needs to perform the following steps.

- ***Data Encryption*** :

Given Data set $D_x = \{s_{x,1}, s_{x,2}, \ldots\ldots, s_{x,n_i}\}$

And Requester public key is $kp = g^a$

Worker W_i chooses n_x random values $r_{x,y} \in R\ Zp$,

Where y=1,2,3…..n_x

Computes ciphertext set C_x

$C_{x,y} = \{c_{x,1}, c_{x,2}, \ldots\ldots c_{x,n_i}\}$

Where $C_{x,y}$ is computes as follows:

$C_{x,y} = \left(g^{r_{x,y}}, m_{x,y}, kp^{r_{x,y}}\right)$

- ***Data Hashing*** :

Given data set $D_x = \{s_{x,1}, s_{x,2}, \ldots\ldots, s_{x,n_i}\}$

And the shared secret hash key shk_h

Worker W_i computes the hashing set $H_x = \{h_{x,1}, h_{x,2}, \ldots\ldots, h_{x,n_x}\}$

Where $h_{x,y}$ is computed as follows,

$h_{x,y} = H\left(shk_h, s_{x,y}\right)$

4.3 Batch Verification

- ***Signature Generation***:

When finishing the above three steps, worker Wi will compute her signature on $acc(H_x)$. The original ring signature scheme is described . Given all worker'spublickeys($kp_1, kp_2 \ldots\ldots, kp_m$), $acc(H_x)$, and her private key kpriv, worker W randomly chooses $b_{x,y} \in R\ z_p$ for all the other workers W_y , where y = 1, 2, ..., m, y $\neq$ x, and computes

$sig_{x,y} = g^{b_{x,y}}$

$$sig_{x,x} = \left(\frac{M_x}{\prod_{y \neq x} kp_{,}^{\,b_{x,y}}}\right)^{\frac{1}{kpriv_x}}$$

The ring signature for $acc(H_x)$ is $sig_{w_m} = \{sig_{x,1}, sig_{x,2}, \ldots\ldots, sig_{x,m}\}$. However, in real life, when the number of workers t is large and workers are distributed over a wide area, it's very time-consuming or impossible for a worker to communicate with all the other workers to get their public keys. Therefore, we cannot directly apply the above ring signature. Instead, we assume every worker belongs to a ring signature group, and all workers in the same group are in proximity with each other. We use K_i to denote the index set of workers who are in the same signature group as W_i, and $M_{min}|K_i| \leq M_{max}$ where $M_{min}$and M_{max} are the minimum and maximum number of workers in any signature group. Then, W_x's ring signature is $sig_{w_m} = \{sig_{x,y}\}$, $y \in K_x$, where

$$sig_{x,x} = \left(\frac{M_x}{\prod_{y \neq x} kp_{,}^{\,b_{x,,y}}}\right)^{\frac{1}{kpriv_x}}$$

4.4 Intersection Computation:

After successful verification of the ring signatures, the cloud computes the intersection set for the requester. Define N_D as the intersection set of the original data sets $\{D_1, D_2, \ldots\ldots, D_m\}$, i.e.,

$N_D = \{D_1 \cap D_2 \cap \ldots\ldots \cap D_m\}$

. Because all data sets $\{D_1, D_2, \ldots\ldots, D_m\}$ are encrypted by workers before being sent to the cloud, the cloud is unable to find N_D for the requester based on the ciphertexts. Instead, the cloud needs to find all the ciphertexts whose plaintexts correspond to the intersection set N_D. Assuming $s_{x,y} \in N_D$, for some x's and y's, then we define N_c as the set of ciphertexts $c_{x,y}$ of all elements $s_{x,y} \in N_D$

$N_C = \{C_{x,y}\} s_{x,y} \in N_D$

The cloud derives N_D based on hashing sets $\{H_1, H_2, \ldots\ldots, H_m\}$, because $s_{x,y}$ and $h_{x,y}$ are one-to-one mapping, N_D is equivalent to NH, where

$NH = H_1 \cap H_2 \cap \ldots\ldots.. \cap H_m$

II. Take H_1 and H_2 as an example,

where $H_1 = \{h_{1,1}, h_{1,2}, \ldots\ldots, h_{1,n_1}\}$and $H_2\{h_{2,1}, h_{2,2}, \ldots\ldots, h_{2,n_1}\}$If $h_{1,p} = h_{2,q}, 1 \leq p \leq n_1, 1 \leq v \leq n_2$,then $s_{1,p} = s_{2,q}$, $s_{1,p} \in D_1$ and $s_{2,q} \in D_2$. This means that $s_{1,p}$ (or $s_{2,q}$)$\in D_1 \cap D_2$. If $s_{1,p} \in D_1 \cap D_x$ for all x = 1, 2, ..., m, then the cloud knows that $c_{,p} \in N_C$. After comparing every pair of elements $h_{x,p} \in H_x$ and $H_{y,q} \in H_y$, the cloud finally obtain $N_H$andN_C.

5. Result and Discussion

Parameters setup

Parameters	values
Group order	128 bits
Number of workers M	10000 to 50000
Size of data set D	1000 to 5000
Size of intersection set	50 to 250

Table 2

Batch Verification:

To make verification more efficient, we propose batch verification. When there are 104 workers and each worker has 1000 elements in the data set, the data volume is at least 108 Bytes. The computational cost is high even for the cloud. We show the cost reduction at the cloud when batch verification is used in Fig. 5(a). The cost reduction at cloud is 34.1s when there are 104 workers, and 155.8s when there are 5 times 104 workers. The cost reduction at the requester is also very obvious, as shown in Fig. 5(b). When there are 5 time 104workers, the cost reduction for verification can be 840s, which is a great improvement compared with original cost of 912s.

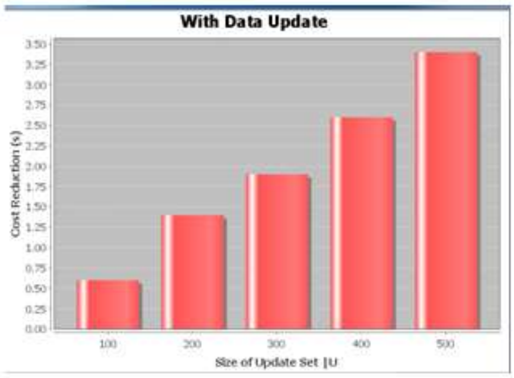

Fig. 5(a) Batch verification

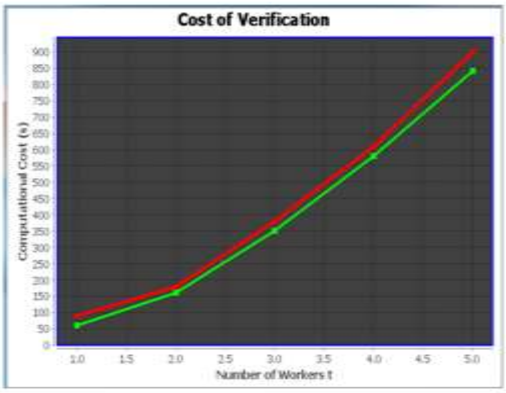

Fig.5(b) cost of verification

6. Conclusion:

In this paper we proposed a scheme to enable the requester to delegate set operations over crowd sourced Bigdata to the cloud.so that it can achieve worker's data and identity privacy are preserved,data pre-processing, batch verification, and data updates.

REFERENCES**:**

[1] S. S. Kanhere, "Participatory sensing: Crowdsourcing data from mobile smartphones in urban spaces," in Mobile Data Management (MDM), 2011 12th IEEE International Conference on, vol. 2. IEEE, 2011, pp. 3–6.

[2] Q. Li and G. Cao, "Privacy-preserving participatory sensing."

[3] "Efficient and privacy-preserving data aggregation in mobile sensing," in Network Protocols (ICNP), 2012 20th IEEE International Conference on, Oct 2012, pp. 1–10.

[4] Q. Li, G. Cao, and T. La Porta, "Efficient and privacy-aware data aggregation in mobile sensing," Dependable and Secure Computing, IEEE Transactions on, vol. 11, no. 2, pp. 115–129, March 2014.

[5] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos, "Anonysense: privacy-aware people-centric sensing," in Proceedings of the 6th international conference on Mobile systems, applications, and services. ACM, 2008, pp. 211–224.

[6] R. Zhang, J. Shi, Y. Zhang, and C. Zhang, "Verifiable privacy-preserving aggregation in people-centric urban sensing systems," Selected Areas in Communications, IEEE Journal on, vol. 31, no. 9, pp. 268–278, September 2013.

[7] S. S. Kanhere, "Participatory sensing: Crowdsourcing data from mobile smartphones in urban spaces," in Distributed computing and internet technology. Springer, 2013, pp. 19–26.

[8] H. Yue, L. Guo, R. Li, H. Asaeda, and Y. Fang, "Dataclouds: Enabling community-based data-centric services over the internet of things," IEEE Internet of Things Journal, vol. 1, no. 5, pp. 472–482, Oct 2014.

[9] K. Hara, S. Azenkot, M. Campbell, C. L. Bennett, V. Le, S. Pannella, R. Moore, K. Minckler, R. H. Ng, and J. E. Froehlich, "Improving public transit accessibility for blind riders by crowdsourcing bus stop landmark locations with google street view: An extended analysis," ACM Transactions on Accessible Computing (TACCESS), vol. 6, no. 2, p. 5, 2015.

[10] B. Liu, Y. Jiang, F. Sha, and R. Govindan, "Cloud-enabled privacypreserving collaborative learning for mobile sensing," in Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems .ACM, 2012, pp. 57–70.

[11] G. Zhuo, Q. Jia, L. Guo, M. Li, and Y. Fang, "Privacy-preserving verifiable proximity test for location-based services," in 2015 IEEE Global Communications Conference (GLOBECOM). IEEE, 2015, pp. 1–6.

[12] G. Zhuo, Q. Jia, L. Guo, M. Li, and P. Li, "Privacy-preserving verifiable data aggregation and analysis for cloud-assisted mobile crowdsourcing," in INFOCOM, 2016 Proceedings IEEE. IEEE, 2016.

[13] H. Li, D. Liu, Y. Dai, and T. H. Luan, "Engineering searchable encryption of mobile cloud networks: when qoe meets qop," Wireless Communications, IEEE, vol. 22, no. 4, pp. 74–80, 2015.

[14] H. Li, Y. Yang, T. Luan, X. Liang, L. Zhou, and X. Shen, "Enabling finegrained multi-keyword search supporting classified sub-dictionaries over encrypted cloud data," IEEE Transactions on Dependable and Secure Computing, vol. PP, no. 99, pp. 1–1, 2015.

[15] X. Chen, X. Wu, X.-Y. Li, Y. He, and Y. Liu, "Privacy-preserving highquality map generation with participatory sensing," in INFOCOM, 2014 Proceedings IEEE. IEEE, 2014, pp. 2310–2318.