Fully Homomorphic Encryption and k-Nearest Neighbour Classification

Kiowa Scott-Hurley[1], Christopher Watkins[1]

[1]CSIRO, Scientific Computing, Clayton, Australia, kiowa.scott-hurley@csiro.au

INTRODUCTION

The advent of cloud computing has enabled users to outsource costly data stor-age and computations to the cloud. However, much of the data we work with, such as medical and financial data, is sensitive and must be secured to avoid legal and business ramifications. This requires encryption prior to outsourcing, but it is costly to encrypt data, outsource it, then localise and decrypt it to per-form useful computations. Developments in homomorphic cryptography make it possible for computations to be performed on encrypted data, allowing the cloud architecture to securely store and process sensitive data. Moreover, current interest in machine learning techniques have provided methods of processing and interpreting large amounts of data, making it suitable for use with in the cloud architecture. As a result, interest in encrypted machine learning and encrypted cloud computing has arisen [5, 12, 14, 13, 4, 8]. This paper will implement the k-Nearest-Neighbour (kNN) classification technique, due to its relative simplicity. Previous implementations of encrypted kNN have used partially or somewhat homomorphic encryption, or other privacy preserving techniques, whereas our implementation aims for a fully homomorphically encrypted scheme [5, 12, 14, 13, 4]. Whilst previous techniques have been successful, this implementation serves to demonstrate that fully homomorphic methods have become fast enough for handling more complex computations, and that further research should consider what other algorithms which cannotbe implemented yet may be possible in this space.

BACKGROUND

2.1 Fully Homomorphic Encryption

Homomorphic encryption (HE) schemes allow operations to be performed on encrypted data. Since HE was first imagined by Rivest, Adleman and Dertouzos in 1978 [10], the field has made several advancements, including Gentrys 2009 discovery that fully homomorphic encryption (FHE) was possible [6]. This meant it was possible, albeit slowly, to perform additions and multiplications on encrypted data an arbitrary amount of times. Following Gentrys discovery, many improvements have been made to FHE schemes, finally allowing them to be implemented in libraries like HeLib [7] and SEAL [9]. These libraries have taken FHE from a purely theoretical space into a practical one, and our implementation takes advantage of the Microsoft SEAL library and underlying CKKS scheme in C++ [3]. This library was chosen due to its user-friendly nature and ease of parameter selection.

2.2 K-Nearest Neighbours Classification

kNN classification is both non-parametric, requiring no assumptions on the data distribution, and lazy, requiring no generalisation of training data points, and hence no training phase. kNN classification consists of reading in training data and corresponding classifications, reading in a test point, and calculating the Euclidean distance from each point to the test point. The k-nearest training points are deduced, and the most common classification among them provides a classification prediction for the test point. The computations in this technique are minimal, making it ideal for FHE without employing optimisation techniques. For these reasons, kNN classification was chosen to implement in encrypted space.

RESULTS

We were able to demonstrate that the scaling behavior of the algorithm in encrypted space was similar to the original algorithm albeit at a fixed increased cost. Figure 1 demonstrates how the model runtimes sales with the number of samples, number of features in each data point and the number of nearest neighbours used in the algorithm. Furthermore Figure 1 illustrates the tolerable run times of our simple implementation across a range of industry relevant hyperparameter combinations.

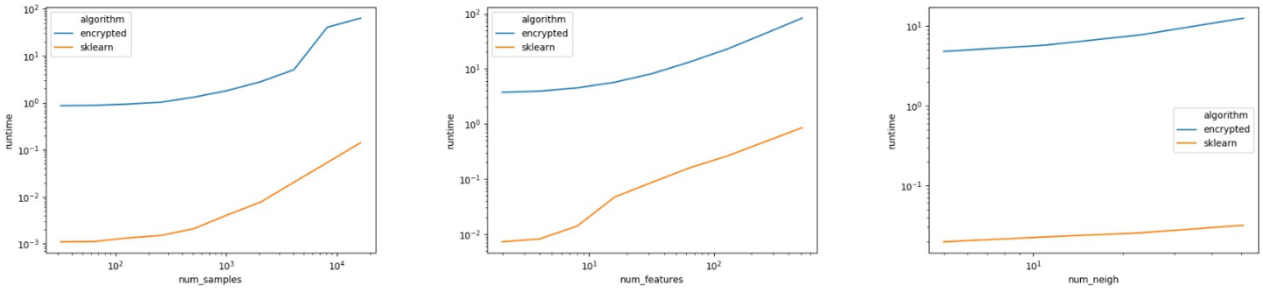

Figure 1 – These curves compare the runt time for our encrypted kNN implementation as compared to the scikit-learn python implementation. The first figures shows run times as the number of data points is increased, the middle figure illustrates run times for increasing feature size and the final figure displays run times as the number of nearest neighbours are adjusted.

FUTURE RESEARCH

The current implementation could be improved upon in a few key ways. Firstly, despite its ability to manage data relatively quickly, no FHE optimisation techniques like modulus switching [1] or re-linearisation [2] have been employed. This could provide further speed ups to the process. Moreover, the current scheme does not ensure the security of data access patterns, which could possibly provide information to the cloud about the type of data it is handling. Our scheme is easily adapted for kNN regression, where in the final step the user finds the average of the classifications instead of the most occurring to make their predictions. In general, looking into other machine learning techniques in FHE space looks promising. Finally, the next step in this scheme is to modify it to allow multiparty computation. In this scenario multiple users would upload data to the cloud without sharing it with each other, thus forming predictions from a larger database with-out compromising valuable or private information. At current our model would rely on no user-cloud collusion, as each party would need both the public and private keys, so if the cloud provided access to encrypted training and

classification data it would no longer be a secure way to share data. Techniques in secure multiparty computation such as secret sharing [11] may guide the development of this scheme.

CONCLUSION

Our implementation of kNN in fully homomorphic encryption has demonstrated it is possible to process large encrypted datasets in a reasonable time. Whilst kNN does not require fully homomorphic encryption, this opens pathways to using other machine learning techniques requiring more operations to be performed whilst data is encrypted. This would allow researchers in medical and financial areas to take advantage of private data to make advances in areas such as stock market prediction and medical diagnoses.

REFERENCES

1. Coron, J-S., Naccache, D., Tibouchi, M.: Public Key Compression and Modulus Switching for Homomorphic Encryption over the Integers. Cryptology reprint Archive. https://eprint.iacr.org/2011/440(2011).

2. Chen, H.: Optimizing relinearization in circuits for homomorphic encryption. CoRR.(2017).

3. Chen, H., Chillotti, I., Song, Y.: Improved Bootstrapping for Approximate Homo-morphic Encryption. Microsoft Research. (2018).

4. Elmehdwi, Y., Samanthula, B. K., Jiang, W.: secure k-nearest neighbor query over encrypted data in outsourced environments. IEEE ICDE. pp. 664-675, (2014).

5. Kesarwani, M., Kaul, A., Naldurg, P., Patranabis, S., Singh, G., Mehta, S.,Mukhopadhyay, D.: Efficient Secure k-Nearest Neighbours over Encrypted Data. EDBT. (2018).

6. Gentry, C.: A Fully Homomorphic Encryption Scheme. Stanford University (2009).

7. Halevi S., Shoup V.: Algorithms in HElib. Advances in Cryptology p. 554-571, Berlin (2014).

8. Hu, H., Xu, J., Ren, C., Choi, B.: Processing private queries over untrusted datacloud through privacy homomorphism. IEEE ICDE. pp. 601-612, (2011).

9. Microsoft: Simple Encrypted Arithmetic Library (release 3.1.0). Microsoft Research.https://github.com/Microsoft/SEAL (Dec 2018).

10. Rivest, R. L., Adleman, L., Dertouzos, M. L.: On data banks and privacy homo-morphisms. Foundations of secure computation. 4, 11, pp. 169-180 (1978).

11. Shamir, A.: How to share a secret. Communications of the ACM, 22, 11, pp.612-613(1979).

12. Wong, W. K.,Cheung, D. W.-l., Kao, B., Mamoulis, N.: Secure knn computation on encrypted databases. SIGMOD. pp. 139-152 (2009).

13. Yao, B., Li, F., Xiao, X.: Secure nearest neighbor revisited. IEEE ICDE. (2013).

14. Zhu, Y., Xu, R., Takagi, T.: Secure k-nn computation on encrypted cloud data without sharing key with query users. Cloud Computing. pp. 55-60 (2013).