The Implementation of BERT over FHE

羅采蓁[1], 郭璋羽[2]

National Taiwan University[1, 2]

b09705059@ntu.edu.tw[1]

b09705050@ntu.edu.tw[2]

ABSTRACT

本次實作結合了現今自然語言處理技術中十分熱門的預訓練技術，BERT，和密碼學上的同態加密方式，HE，針對選定的資料集分析特定模型在加密前後的結果來進行評估，進而探討 BERT 進行同態加密前後的比較與未來密碼學可以應用的範圍。

KEYWORDS

BERT, HE

1. INTRODUCTION

本學期資訊安全的課程當中介紹了許多密碼學相關的知識，讓我們了解：有密碼學的基礎才能有所謂的資訊安全。在課堂之餘，我們探討了關於同態加密的加密形式，對此種加密方式有更深入的認識，也發現課堂內介紹的許多密碼學加密方式其實可以歸類成同態加密。

而另一方面，我們在系上專題正在利用 BERT 進行文本分析，而經過一學期資訊安全課程學習到的知識，我們特別想要去了解是否存在 BERT 與密碼學融合的應用。同態加密一直是密碼學領域的一個重要課題，我們也希望利用本次實作，結合所學，來學習到更多的知識也增強實作經驗。

2. PROBLEM

使用 BERT 時，embedding 可能會引起隱私問題：某些時候可以透過 embedding 回推原始的文本內容（英文：Embedding Inversion）。

在本篇報告中我們討論使用同態加密的概念將 embeddings 加密，若這種方法奏效，則可以規避 Embedding Inversion Attack，且其理論上也幾乎不會削弱 BERT 下游的文件分類任務的能力。

3. RELATED KNOWLEDGE

本段落介紹基於變換器的雙向編碼器表示技術（英文：Bidirectional Encoder Representations from Transformers，BERT）與同態加密（英文：Homomorphic Encryption，HE）的基本知識，協助讀者對於接下來實作有更全面的認識。

3.1 BERT

BERT 是由 Google 提出，用於自然語言處理（英文：Natural Language Processing，NLP）的預訓練技術。模型的結構主要爲 Transformer 這個經典 NLP 模型中的編碼器（英文：Encoder），透過雙向（英文：Bidirectional）設計讓模型能夠考慮文本字詞的前後關係，以增強模型對文本的理解，最後將理解的結果透過文字表徵（英文：Representation）的方式輸出。

BERT 會將模型輸入放入 12 或 24 層（對應到 BERT-Base 版及 BERT-Large 版）串連的 Encoder 中，下一個 Encoder 的輸入爲上一個 Encoder 的輸出，而最後一層的 Encoder 輸出即爲 BERT 的最終輸出。輸出的每個 token 表徵爲模型對於該 token 的理解，可以用在後續的 NLP 任務當中。因此，將 BERT 應用在 NLP 的整個流程大致爲 [4]：

1. 將新任務的文本資料輸入 BERT
2. 將 BERT 輸出的 token 表徵放入一個分類器當中
3. 透過分類器判斷 token 表徵的類別

在實際任務類型上，我們可以先認識以下三種類型：

1. 單一句子分類任務：將最後輸出的句首 CLS token 表徵，視爲 BERT 對整個句子的理解結果，並以此分類句子的類別。例如判斷語句的情緒爲正面或負面；或是語句是否符合文法。
2. 單一句子標注任務：將最後輸出的每個 token 表徵，視爲 BERT 對每個 token 在句中的意義，並

分出每個 token 的類別。例如判斷每個單詞的詞性或是標籤分類，如人物、動物、建築。

3. 一次考慮兩個句子間的關係，並將最後輸出的句首 CLS token 表徵視爲 BERT 對句子間關係的理解結果，並對句子間的關係進行分類。例如用來區分兩句話是否相似或矛盾，也可應用於判斷問答句之間是否匹配。

3.2 Homomorphic Encryption

在大數據時代，越來越多人利用雲端執行作業，將原端當成儲存的空間。而目前，雖然雲端計算帶來了低成本、高性能與便捷性等優勢，但從安全的層面來說，許多用户還是對於將個人敏感資訊上傳到雲端存有疑慮。因此，如何確保上傳資訊的機密性，在不暴露資料內容的情況下對資料進行操作則成爲當務之急。

而同態加密提供一個媒介，讓用户把資料加密後傳給雲端電腦，根據用户寫好的算式，電腦執行對應密文的運算，最終把結果傳回給用户。同態加密扮演的角色是提供一個方法，將資訊傳到一個公開平台前先加密，之後在平台做密文運算。對於平台方，他不會知道正在處理的數據明文爲何，但是他卻能正確得到用户期許的結果。當用户取得結果，只要將其解密，便可以解決用户的問題。

同態加密的安全性基於代數，定義 Δ 爲一個運算元，加密函數爲 E 其中若是函數滿足以下方程式，則意味此函數滿足同態性：

$$E(X \ \Delta \ Y) = E(X) \ \Delta \ E(Y)$$

同態性來自代數領域，一般包括四種類型：加法同態、乘法同態、減法同態和除法同態，而基於所滿足的性質，同態加密又分爲三種：半同態加密（英文：Partially Homomorphic Encryption，PHE）、特定同態（英文：Somewhat Homomorphic Encryption，SHE）和全同態加密（英文：Fully Homomorphic Encryption，FHE）。

假設 *Eval* 爲一輸入和輸出值都是密文，並對密文進行運算的函數 [1]。

1. Partially Homomorphic Encryption：若是 *Eval* 只滿足加法同態或是乘法同態的運算，則稱之半同態加密。課堂上所提過的 RSA 或是 ElGamal 都是所謂的 PHE，其都滿足乘法同態，以下舉例 RSA 來證明其乘法同態的性質：假設 m_1 和 m_2 爲任意的明文，e 爲 RSA 產生金鑰時使用的常數，$E(m_1) * E(m_2) = ({m_1}^e \mod n) * ({m_2}^e(\mod n)) = (m_1 * m_2)^e \mod n = E(m_1 * m_2)$。RSA 的乘法同態性說明當計算 $E(m_1) * E(m_2)$ 時可以直接由 $E(m_1)$ 和 $E(m_2)$ 得出結果，而不用對其進行解密。

2. Somewhat Homomorphic Encryption: 若是 *Eval* 只同時滿足加法同態或是乘法同態的運算，但只能實現部分特定操作的同態性，或是計算次數有限，則稱之特定同態加密。BGN 加密方法（英文：Boneh-Goh-Nissim Encryption）爲一個例子，其爲基於子群決策難題（英文：Subgroup Decision Problem）提供運算上的保證。

3. Fully Homomorphic Encryption: 若是 *Eval* 支持任意次數同態運算和隨機函數的同態加密，則稱之全同態加密。主要是 Gentry 加密方法，使用理想格（Ideal Lattice）和密碼學中的難題，例如 Learning with Errors（縮寫：LWE）問題和 Ring Learning with Errors（縮寫：RLWE）問題。這些難題提供了運算上的保證，確保在加密的密文上進行運算不會泄露明文的資訊。

4. SOLUTION

4.1 Architecture

在 Privacy-Preserving Text Classification on BERT Embeddings with Homomorphic Encryption [2] 中，提出了以下的加密結構：

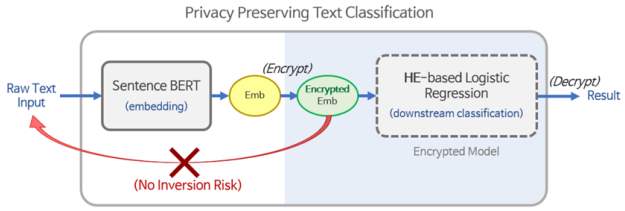

圖 1 The Architecture of Text Classification.

這個結構是在初始文本做完 embedding 轉換後，進行 Logistic Regression 前先對 embedding 使用同態加密，再將加密後的 embedding 放進 Logistic Regression 去跑。

這裡選用 Logistic Regression 是因爲 Homomorphic Encryption 的方法相較於直接對 plaintext 處理，需要更多的運算成本，因此如果使用神經網路等更複雜的模型，這裡提出的方法可能無法支援。

4.2 Code

我們實作 BERT over FHE 的程式中有以下幾個較

爲重要的部分：

1. BERT embedding：使用 BERT 算出訓練與測試資料的 embedding，並將 embedding 作爲後續模型訓練及預測的輸入資料。使用'bert-base-cased'版本。

2. Logistic Regression：使用 LR 的方式訓練模型，並使用 Concrete-ML 提供的 fhe="execute" 設定進行 FHE 的運算。

   (a) model.compile(input)：將模型編譯以取得 FHE 的 inference engine。這裡的 input 我們使用訓練資料的 embedding。

   (b) model.predict(input, fhe="execute")：在模型預測時開啓 fhe="execute" 的設定，使模型在 FHE 的狀態下進行預測。開啓 FHE 的狀態意味著此時模型會先將 input 加密，然後使用支援 FHE 的 inference engine 進行預測。這裡的 input 是測試資料的 embedding。

完整程式碼可見此 Google Colab 連結：BERT with FHE - 20Newsgroups

5. RESULT

本次實驗選擇的任務爲文本分類，即我們目的是建立一個能將文本進行分類的模型。實驗使用的程式語言爲 python3，操作的平台爲 Google Colab。

5.1 Experiment Dataset

我們使用了兩組資料集進行實驗，皆爲新聞分類的資料集，一組是 20 Newsgroups，一組是 news articles。

- 20 Newsgroups：有 11314 篇訓練資料與 7532 篇測試資料，共 18846 篇新聞文本，算是一個大資料集，文本語言爲英文。

- news articles：共 244 篇新聞文本，需自行拆分訓練與測試樣本，是篇小的資料集，文本語言爲中文。

新聞分類的資料集中，每一筆資料包含文本內容（一篇新聞）以及其所屬類別。而我們的任務就是建立一個分類模型，當傳入一篇新聞後，模型能正確地給出該新聞所屬類別。

5.2 Experimental Data

實驗數據如表 1 所示，20NewsG 指的是 20 Newsgroups 這個資料集，news 指的是 news articles 這個資料集，LR 爲 Logistic Regression，XGBC 爲 XGB Classifier。

Clear 是指在不加密的情況下，模型直接運算所得到的分數，FHE 則是模型在 FHE 的情況下運算所獲得的分數。Loss 則是表示模型進使用 FHE 後的能力下降，即 Clear 的分數減 FHE 的分數。

而判斷模型的好壞有以下四種計分方式：accuracy、precision、recall、f1-score，後三種在計算方式上又分爲 Micro 以及 Macro 兩種類型，分別以 mi- 與 ma- 表示。

表 1 Model Score

Dataset	Model	Score	Clear	FHE	Loss
20NewsG	LR	accuracy	0.5882	0.5882	0
		mi-precision	0.5882	0.5882	0
		mi-recall	0.5882	0.5882	0
		mi-f1-score	0.5882	0.5882	0
		ma-precision	0.5772	0.5772	0
		ma-recall	0.5787	0.5787	0
		ma-f1-score	0.5762	0.5762	0
news	LR	accuracy	0.8	0.8	0
		mi-precision	0.8	0.8	0
		mi-recall	0.8	0.8	0
		mi-f1-score	0.8	0.8	0
		ma-precision	0.5942	0.5942	0
		ma-recall	0.4444	0.4444	0
		ma-f1-score	0.4593	0.4593	0
	XGBC	accuracy	0.76	0.76	0
		mi-precision	0.76	0.76	0
		mi-recall	0.76	0.76	0
		mi-f1-score	0.76	0.76	0
		ma-precision	0.5833	0.5833	0
		ma-recall	0.3889	0.3889	0
		ma-f1-score	0.3910	0.3910	0

在表 1 中可以發現沒有 20 Newsgroups 在 XGB Classifier 下的結果，是因爲 XGB Classifier 較 Logistic Regression 複雜許多，而 20 Newsgroups 是偏大的資料集，因此在 FHE 的狀態執行時間會過長。使用 FHE 所需的時間可見表 2。

表 2 Time for FHE

Dataset-Model	Compilation time	FHE inference
20NewsG-LR	3.8561	880.2507
news-LR	0.2223	3.1181
new-XGBC	0.6109	69.7516

另外在表 1 中可以看到 Loss 全都是 0，意味著使用 FHE 進行運算並不會削弱模型的分類能力。不過我們認爲使用 FHE 是可能導致分類器的能力有所削減，但實驗中使用的模型都是偏簡單的模型，且 20 Newsgroups 是一個優質的資料集，因此我們認爲是基於上述原因才導致在本次實驗中使用 FHE 仍然能擁有與原先相同的分類能力。

6. LEARNINGS

在嘗試結合 BERT 和 Homomorphic Encryption 時，我們首先嘗試了可以進行 FHE 運算的 python libary，如 Pyfhel 等，但由於其支援 FHE 的運算方式爲加密訊息成一個由他們所設計出的特別的資料型態，因此加密後的資料與常態的 Machine Learning 的運算方式並不相容，而 Pyfhel 本身也沒有提供支援 Machine Learning 的相關 API，因此這種方法暫無法實作 BERT 結合 FHE。（除 Pyfhel 外，還有許多提供 Homomorphic Encryption 運算的 python libary，但他們在運用到我們的實驗上都有和 Pyfhel 相同的問題）

因此後來我們參考 Sentiment Analysis on Encrypted Data with Homomorphic Encryption [3] 的做法，使用 Concrete-ML 提供的 "fhe='execute'" 設定直接進行 FHE 的計算，但這個方法同樣有侷限性，我們只能使用 Concrete-ML 有支援的訓練模型，比如 Logistic Regression, XGB (eXtreme Gradient Boosting) Classifier 等，而無法自己建立訓練模型的架構，導致**我們可以對於模型表現進行的優化十分有限** 。

7. CONCLUSION AND FUTURE WORKS

現今的 Concrete-ML 函式庫對於 FHE 在機器學習上的應用只支持對於未加密的資料進行訓練，產生出的模型再進行同態加密，來對加密後的資料做預測。就實際層面來說，最根本保護資料的方法應當是將輸入的資料直接加密，但我們認爲利用 BERT 的技術無法在密文的領域做預測，原因是 BERT 沒有看過加密過後的文字，因此我們選擇利用經過 BERT 後的 embedding 層來完成實作。

在未來，我們預計嘗試將加密後的文章進行訓練，若是加密後的文章也可以和未加密的資料有相同的分類結果，我們會大吃一驚，畢竟照理來說每次加密後的結果應當不會相同，所以一個字元在不同次加密後的結果應當也會不一樣，應當是不存在可以正確訓練出一個可以分類的模型的。

若是能找到存在計算任兩個字元間的相對位置的計算方法，且利用 FHE 的運算結果也相同，則我們認爲可能可以利用加密後的資料來進行分類的操作。

在機器學習的領域中， Embedding Inversion Attack 是一個可能造成嚴重後果的資安問題，透過 HE 有機會填補這個漏洞，但目前的技術還無法支撐 HE 應用在大資料集或者複雜的運算模型，因爲 HE 實現安全運算的代價就是運算效率會大大的降低，這部分也呼應了這門課一直有提到的，資訊安全在實務上就是會與效率、成本等等互相牴觸，追求安全是需要付出代價的。

References

[1] A. Acar et al., "A survey on homomorphic encryption schemes: Theory and implementation", ACM Computing Surveys (CSUR), vol. 51, no. 4, pp. 1-35, 2018.

[2] Privacy-Preserving Text Classification on BERT Embeddings with Homomorphic Encryption (Lee et al., NAACL 2022)

[3] Sentiment Analysis on Encrypted Data with Homomorphic Encryption https://huggingface.co/blog/sentiment-analysis-fhe

[4] BERT 也懂人話？NLP 模型的可解釋性簡易指南 https://edge.aif.tw/aicafe-bert-20220728/