Capitolo 1

Conclusioni e sviluppi futuri

In questo lavoro di tesi, sono stati inizialmente introdotti gli smart contracts e le tecnologie correlate, come la blockchain, delineando il loro funzionamento e la loro importanza nel panorama tecnologico attuale. In seguito, stata effettuata un'analisi delle vulnerabilit pi comuni che affliggono gli smart contracts, soffermandosi sopratutto sulle vulnerabilit prese in considerazione durante la fase sperimentale di questo lavoro. Successivamente, stata esaminanta anche la letteratura presente e le metodologie di classificazione attuali.

Successivamente, il focus si spostato sulla presentazione dei modelli utilizzati in questo studio, in particolare sui modelli della famiglia BERT basati sull'architettura dei Transformers. Tra le principali contribuzioni di questo lavoro vi la dimostrazione della maggior efficacia dei modelli CodeBERT, preaddestrati su codice sorgente, rispetto ai modelli BERT pre-addestrati su testi generici. Inoltre, sono state testate diverse metodologie che hanno permesso di superare il limite di lunghezza di sequenze di token in input ai modelli BERT, consentendo di analizzare interi (o quasi) smart contracts. Nello specifico, il codice dei contratti stato suddiviso in blocchi da 512 token, e sono stati utilizzati approcci in cui gli embedding prodotti dal modello per ciascun blocco sono stati concatenati o aggregati. I risultati ottenuti hanno dimostrato che l'approccio di concatenazione degli embedding garantisce i migliori risultati in termini di accuratezza e F1 score, mentre in termini di recall l'approccio con l'aggregazione degli embedding utilizzando la funzione max risultato il migliore per entrambe le modalit (bytecode e source code).

Infine, stato presentato un approccio di stacking, che ha permesso di combinare il miglior modello preaddestrato sul codice sorgente ed il miglior modello preaddestrato sul bytecode degli smart contracts. Nello stacking un meta-classificatore ha il compito di predire la classificazione finale a partire dalle predizioni dei classificatori base. Sono stati testati vari algoritmi di machine learning utilizzati come meta-classificatori con l'obiettivo di migliorare le performance dei modelli base. I risultati ottenuti hanno dimostrato come l'approccio di stacking permetta di ottenere risultati migliori rispetto all'utilizzo di un

singolo modello, ma sia facilmente prono all'overfitting dei modelli. Nello specifico,I risultati ottenuti dimostrano come modelli pi complessi come SVM e Random Forest siano pi soggetti all'overfitting rispetto a modelli pi semplici come il Gaussian Naive Bayes. Il miglior modello, per, risultato essere la regressione logistica, che ha permesso di ottenere i risultati migliori in termini di accuratezza, F1 score e recall, senza mostrare segni di overfitting. Questo modello ha raggiunto un'accuratezza dell'81,30% ed un F1-Score del quasi 90%. Inoltre, le metriche di recall tutte superiori all'85% a meno della classe access-control (la classe minoritaria del dataset) che si attesta comunque all'82% mostrano come il modello sia in grado di classificare correttamente la maggior parte delle vulnerabilit presenti nei contratti.

Per valutare la necessit di utilizzare modelli ad hoc per task complessi come questo, rispetto ai modelli LLM (Large Language Models) come GPT di OpenAI o Gemini di Google entrati ormai nelle nostre vite sotto forma di chatbot utilizzatissimi per tantissimi task, sono stati effettuati test comparativi. Utilizzando la versione gratuita delle API di Gemini, stato raccolto un campione di 1169 classificazioni dal dataset di test. I risultati ottenuti sono stati molto scarsi, con un'accuratezza del 27% e delle recall pari quasi allo zero in tutte le classi a meno della classe Other, che raggiunge livelli di recall poco piu bassa del 50%. Questi risultati dimostrano come per task complessi come questo sia ancora necessario utilizzare modelli ad hoc.

Possibili direzioni future di ricerca di questo lavoro implicano sicuramente un miglior tuning dei modelli, sopratutto per quanto riguarda la Batch Size. A seguito, infatti, della limitata potenza di calcolo non stato possibile tunare correttamente la batch size, che in molti esperimenti stata obbligata ad essere impostata a valori molto bassi. Sarebbe inoltre, molto interessate utilizzare modelli di classificazione che possano gestire sequenze di token di lunghezza maggiore come Big Bird [**?**] e LongFormer [**?**]. Allo stesso tempo, avendo a disposizione pi risorse computazionali, sarebbe interessante provare a fare finetuning su dei modelli LLM divenuti lo stato attuale al giorno d'oggi come GPT, LLaMa o Mistral. Durante lo svolgimento di questo lavoro, stato gi tentato un approccio di fine-tuning di questi modelli, ma la limitata potenza di calcolo non ha permesso di avere tempi di training ragionevoli con una lunghezza delle sequenze di token accettabile.