Towards Access Control for Machine Learning Embeddings

Roman Matzutt
roman.matzutt@fit.fraunhofer.de
Fraunhofer FIT
Sankt Augustin, Germany

ABSTRACT

In this work, we explore the potential to make embeddings, which are becoming an integral part of machine-learning pipelines, shareable with the general public while providing self-contained access control. To this end, we apply attribute-based encryption and discuss a potential application for supply chain management.

CCS CONCEPTS

• **Information systems → Specialized information retrieval**.

KEYWORDS

Attribute-based encryption, embeddings, access control

ACM Reference Format:
Roman Matzutt. 2024. Towards Access Control for Machine Learning Embeddings. In *European Interdisciplinary Cybersecurity Conference (EICC 2024), June 05–06, 2024, Xanthi, Greece.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3655693.3661296

1 INTRODUCTION

Embeddings, for example graph embeddings, are nowadays commonly used to transform more complex structures into a suitable input for machine-learning algorithms [4]. However, obtaining embeddings may take hours for large inputs [4] and they, thus, are not suitable for on-the-fly generation. In this work, we outline our considerations for making embeddings shareable, such that they can be periodically generated and released to the public. To provide self-contained access control over an embedding's data, we seek to apply attribute-based encryption. In the context of our funded research project CRYPTO4GRAPH-AI, we explore the potential applicability of this scheme to distribute insights about supply chains.

2 BACKGROUND

In the following, we give a brief overview of *embeddings* in the context of machine learning (ML) and *attribute-based encryption (ABE)*.

Embeddings. In ML, an embedding $\mathcal{E}$ generally maps an input from a considered object space $\mathbb{S}$ to a d-dimensional vector $x \in \mathbb{R}^d$, which is then subsequently used as input for further ML algorithms [4]. For instance, this approach allows applying established methods for ML-based *similarity assessment* between objects if $\mathcal{E}$ sufficiently captures this similarity, i.e., the vectors corresponding to "similar" objects have a small distance from each other [3].

EICC 2024, June 05–06, 2024, Xanthi, Greece

ACM ISBN 979-8-4007-1651-5/24/06
https://doi.org/10.1145/3655693.3661296

In this work, we focus on *graph embeddings*, which take a graph $G = (V, E)$ and target dimensionality d as input and can return (depending on the application) a set of d-dimensional vectors, e.g., one vector per node $v \in V$ or one vector per edge $e \in E$ [4].

Attributed-based Encryption. With ciphertext-policy attribute-based encryption (CP-ABE) [2], a user can encrypt a with inherent access control over a set of preassigned *attributes*. To this end, a central manager distributes secret keys to the users based on their attributes. When receiving an ABE-encrypted message, the user can only decrypt it if their attributes match the specified policy.

3 PROTECTION PIPELINE

We now present our ABE-based protection pipeline for shareable ML embeddings. Figure 1 gives an overview of this pipeline.

We assume that a *knowledge owner* has control over a large knowledge base containing sensitive information. Nevertheless, the knowledge owner wants to (or obliged to) share insights with a larger audience in the form of ML embeddings for further processing by the recipients. However, not all recipients are entitled to a full view of these insights, which we assume to be computationally taxing for the knowledge owner to derive. For example, obtaining higher-dimension graph embeddings for huge input graphs easily takes hours even after optimization [4]. Hence, the knowledge owner wants to compute the embedding once and *protect* its components against unauthorized access before publishing it. In the following, we detail the steps of our pipeline, i.e., *obtaining* and *protecting* an embedding before *publicly releasing* it.

① **Obtain Embedding.** We assume that the confidential model of the knowledge owner consists of a large graph $G = (V, E)$ with possible node and edge labels. For example, the labeled graph can constitute a *knowledge graph* [6]. The knowledge owner occasionally runs a graph embedding algorithm such as node2vec [3] on relevant subgraphs of G, or G may evolve over time. In the example of node2vec, the embedding consists of a set of vectors $x_i \in \mathbb{R}^d$ for each node $v_i \in V$. Additionally, the knowledge owner may augment this vector with an additional *data* component D_i to selectively annotate individual embedding points. For a two-dimensional embedding (as shown in Figure 1), the embedding yields a mapping $v_i \mapsto (x_i, y_i, D_i)$ for each node v_i.

② **Protect Embedding.** The knowledge owner protects the generated embedding such that they can subsequently release it to the public, and each recipient may only process the subset of the embedding they are entitled to. To this end, the knowledge owner uses CP-ABE (cf. Section 2) as follows. First, the knowledge owner has to define the relevant attributes and assign them to the users. Afterward, they can define the access policies in a fine-granular manner. Namely, each component as well as the preimage of an embedding vector may be subject to a specific policy. By defining attributes on a per-embedding basis, the knowledge owner gains

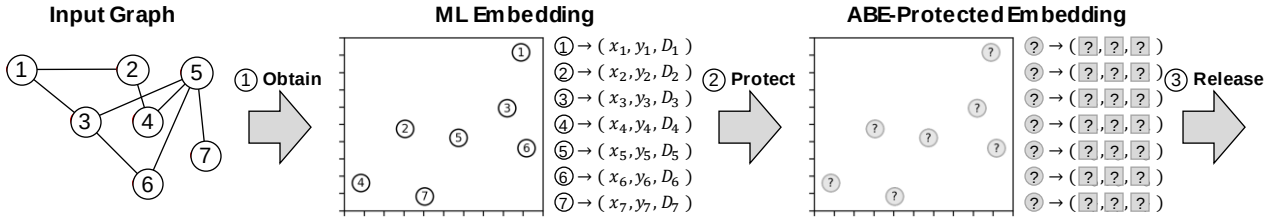
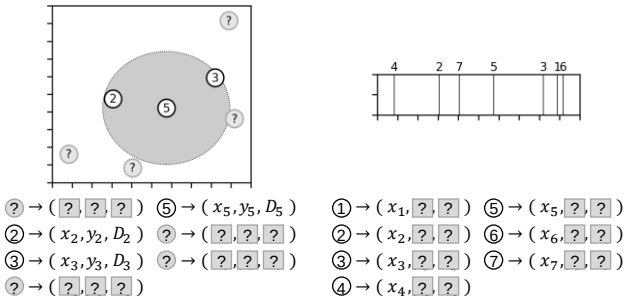


Figure 1: Our pipeline consists of ① obtaining the ML embedding, ② protecting it using CP-ABE, and then ③ releasing it.



(a) Restricted Distance **(b) Restricted Dimensions**

Figure 2: Embedding data obtainable for different policies.

the flexibility to reuse partial insights from the embedding to define the policies. Figure 2 gives two examples of the access restrictions the knowledge owner can make. In the first example (Figure 2a), access is based on the Euclidean distance between vectors of the embedding. Here, the recipient is allowed to explore the proximity of Node 5 in the embedding, yielding full access to the embedding data corresponding to Nodes 2, 3, and 5. In the second example (Figure 2b), the recipient may only inspect a lower-dimension projection of the embedding (only the x-coordinates) and is not allowed to access any data components. However, now the recipient may see all preimages of this low-dimensional representation.

③ **Release Embedding.** After protecting the embedding, the knowledge owner can release it to the public. However, if they generated attributes or attribute assignments on the fly, the knowledge owner additionally has to distribute the corresponding keying material. For simplicity reasons, each potential recipient can initially register with the knowledge owner and announce a public key during this process. Then, the knowledge owner can encrypt each recipient's key with their respective public key and attach those values to the protected embedding. Hence, the embedding is self-contained and, once distributed, can be applied by any eligible recipient even if the knowledge owner goes offline.

4 CASE STUDY: SUPPLY CHAIN KNOWLEDGE

In the context of our funded research project CRYPTO4GRAPH-AI, we are eager to refine and apply the approach described in Section 3 to a use case from the domain of supply chain management.

Typically, supply chains are organized with a certain degree of decentralization, i.e., individual links are not necessarily aware of the whole supply chain [1]. Furthermore, organizations are expected to increasingly exchange information across the boundaries of individual supply chains in the future [5]. This development paves the way for dedicated *knowledge providers* to assist this exchange. However, an over-reliance on a central entity in an otherwise decentralized ecosystem should be avoided. Hence, we are interested in exploring our protection pipeline for ML embeddings in this context.

Scenario. In addition to the central knowledge provider, we assume that the participants are predominantly the organizations making up the covered supply chains. The knowledge provider maintains an expressive knowledge graph describing this federation of supply chains, but (for now) we assume that the nodes of the obtained ML embeddings correspond to the covered organizations.

Knowledge Benefits. Having access to ML embeddings enables the organizations to identify similarities in private. Hence, they can either compare themselves to other clusters of organizations of interest or identify other organizations similar to the ones they are already monitoring. Such comparisons can help improve, e.g., performance, sustainability, or public visibility. Contrarily, organizations may identify new potential suppliers to increase their flexibility, or identify risk-ridden organizations to avoid them, e.g., if they turn out to be likely to engage in unethical work practices.

Policy Considerations. Yet, organizations should be protected, as the embedding may otherwise leak sensitive information. To respect this need, the knowledge provider may distribute attributes based on relationships between the organizations, e.g., whether they are direct competitors or have a direct supplier relationship.

5 CONCLUSION

In this work, we have outlined the potential for creating shareable machine-learning embeddings by protecting them via attribute-based encryption prior to release. We are eager to further explore this concept in our funded project CRYPTO4GRAPH-AI.

ACKNOWLEDGMENTS

This work has been funded by the German Federal Ministry of Education and Research (BMBF) under funding reference number 01IS21100A. The responsibility for the content of this publication lies with the authors.

REFERENCES

[1] Lennart Bader, Jan Pennekamp, Roman Matzutt, David Hedderich, Markus Kowalski, Volker Lücken, and Klaus Wehrle. 2021. Blockchain-Based Privacy Preservation for Supply Chains Supporting Lightweight Multi-Hop Information Accountability. *Information Processing & Management* 58, 3 (05 2021).
[2] John Bethencourt, Amit Sahai, and Brent Waters. 2007. Ciphertext-Policy Attribute-Based Encryption. In *IEEE S&P'07*.
[3] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *ACM KKD'16*.
[4] Jason Mohoney, Roger Waleffe, Henry Xu, Theodoros Rekatsinas, and Shivaram Venkataraman. 2021. Marius: Learning Massive Graph Embeddings on a Single Machine. In *USENIX OSDI'21*.
[5] Jan Pennekamp. 2024. Evolving the Industrial Internet of Things: The Advent of Secure Collaborations. In *IEEE/IFIP NOMS'24*.
[6] Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Merialdo. 2021. Knowledge Graph Embedding for Link Prediction: A Comparative Analysis. *ACM Trans. Knowl. Discov. Data* 15, 2 (2021).