Privacy-Preserving Large Language Models (PPLLMs)

Draft Document (for Review)

***Abstract*—Recently large language models (LLMs) have gained significant attention as they have shown surprising signs of artificial general intelligence (AGI). Artificial intelligence and large language models can be used for various good purposes, such as digital assistants for knowledge creation. However, such powerful models can have potential risks as well. Among other concerns and risks are security and privacy risks that AI models can pose to data as well as users. In this article, we discuss how mathematical structures, such as polynomial and vector spaces, and privacy-preserving delegation of polynomial and matrix-vector functions can be used for transforming a computational model (including LLMs) to a privacy-preserving computational model.**

Furthermore, we highlight some well-known cryptographic constructions along with some solutions by which LLMs can be improved, in the sense that they can preserve the privacy and security of data and thus users. Overall, privacy-preserving and zero-knowledge LLMs, that we introduce in this article, could be potential solutions for preserving the privacy of data and users to some good and reasonable extent. More importantly, perhaps AI models should be trained on publicly available trustworthy data; and the trained models should be compressed and used by users locally.

***Index Terms*—Privacy-Preserving Computation, Private Polynomial Computation, Privacy-Preserving Large Language Models, Secure Computation, Fully Homomorprhic Encryption, Peivacy-Preserving Machine Learning, Zero-Knowledge Language Models, Trustworthy AI, Security and Privacy Risks of AI Models, FHE, PPLLM, ZKLLM, PPML, ZKML**

I. Introduction

In the last couple of years artificial intelligence (AI) and large language models (LLMs) have gained significant attention; and have already been used by many users around the world. This has been due to advances in generative AI models that can show human-level capabilities in text and image generation; and can even possibly pass the Turing test. Since OpenAI released its state-of-the-art pre-trained large language model (known as ChatGPT) the AI research and development have, perhaps, entered a new era. On one hand, companies and the so-called tech giants have accelerated the development of their AI tools (e.g., OpenAI's ChatGPT and DALL-E, Google's PaLM-2, PaLM-E, Bard, and Gemini, Meta's LLaMA, Microsoft's Bing Chat, Baidu's Ernie Bot, etc.) [1]. On the other hand, there have been more concerns about how these models are and will be used. The pioneers of AI, by large, agree that further research and development on AI should be done very cautiously. Some AI pioneers, even, believe that there should be a short pause on AI research and development; so that the society can be prepared for adopting such advanced AI tools.

Large language models (LLMs) and generally AI tools can be used for various good purposes, e.g., in education and research, as digital assistants, as knowledge or search engines, medical education [2], etc. But, there are other important aspects of AI technologies that should be taken care of more responsibly. Some challenging aspects of AI models that have raised concerns include, but are not limited to, algorithmic bias, alignment, inaccuracy of the models due to hallucination, data privacy and security [3]. To make a long story short, most of the story revolves around *data*, computational *models*, and how they are developed and used. Perhaps, it would be appropriate to call these two terms (i.e., *data* and computational *models*) the digital gold or maybe digital oil of the current technological era; in the sense that computational models and data function like the engine and fuel for some of the important and revolutionary businesses that might affect the lives of billions of people in the near future.

To have a better understanding of the situation, we are going to focus on these two key terms, i.e., *data* and computational *models*. Non-digital data, e.g., paper documents, has been around for a long time. Yet, as the world has become more digital data-driven, more challenges and opportunities have emerged. The result of decades of scientific works and technological development can be seen on several popular branches of computer science/engineering, i.e., big data, cryptography (with sub-fields such as zero-knowledge proof, secure computation, fully homomorphic encryption, Blockchain and cryptocurrencies) and artificial intelligence (with sub-fields such as artificial neural networks, machine learning, deep learning, reinforcement learning, NLP, large language models, computer vision, etc.).

When it comes to preserving data and also users' privacy, there is a well-known technique called cryptography. Cryptography, as a tool or technique for data protection, has been around for a long time. According to historical evidence, people have used basic cryptographical techniques since thousands of years ago. Since mid and in the second half of the twentieth century and with the growth of digital data and devices, modern cryptography gained much more attention and saw significant progresses. Among others, there are several areas in cryptography that are more relevant to this

work, e.g., secure computation [4], [5], fully homomorphic encryption (FHE) [6]–[8], private polynomial computation (PPC) [9], zero-knowledge proofs (ZKP) [10], secret sharing (SS) [11], [12], verifiable secret sharing (VSS) [13] (see also [14]), garbled circuits (GC) [4], [5], oblivious transfer (OT) [15], universal circuits (UC) [16] (see also [17]), differential privacy [18], secure function evaluation (SFE) [19], etc. The main theme behind these constructions and notions is how to perform computation securely, or how to do computation on data without revealing the data.

The cryptographic constructions mentioned above enable one to run the computational *models* in such a way that the privacy and security of *data* can be preserved to some good extent. These techniques provide a variety of solutions and mathematical tools to configure different levels of privacy and security measures both on data and computational models. While hard security solutions such as fully homomorphic encryption (FHE) [6]–[8] can provide very strong data security guarantees, soft security measures such as trust and reputation [20], [21] can be helpful for designing trustworthy and robust models. On the other hand differential privacy-based solutions [18], [22] enable us to preserve data privacy by adding some randomness to data; and zero-knowledge proof (ZKP) systems [10] enable a party to prove a statement without revealing any information about the statement.

In this article we take a systematic approach for tackling the data privacy problem. Specifically, we discuss how polynomial spaces [23] and vector spaces [24] can be used for encoding and encrypting data; and how the delegation of polynomial and matrix-vector functions can be used for transforming a computational model to a privacy-preserving computational model. Furthermore, we highlight some well-known cryptographic techniques by which a typical computational model (including machine learning, artificial neural network, artificial intelligence, and large language models), can be turned into a privacy-preserving and zero-knowledge computational model. To this end, we first define the concepts of zero-knowledge and privacy-preserving computational models.

A. Article Organization

This article is organized as follows. In Section II we provide some definitions and propositions which highlight the goals that we want to achieve in this article. In Section III we discuss the cryptographic building blocks that are used in this article. Particularly, in sub-section III-A we review the delegation of polynomial and matrix-vector functions. In sub-section III-B we discuss some privacy-preserving transformations of polynomial and matrix-vector functions. In sub-section III-C we present a novel transformation for transforming (or turning) AI and large language models (LLMs) into privacy-preserving models. Section IV provides the discussion and future research directions. The article is concluded in section V.

II. Definitions and Cryptographic Constructions

In this section we provide some definitions to highlight the key aspects of current data and computational paradigm which is common in various data- and model-driven applications including in AGI, AI, ML, and LLM models.

Definition 1 (Computational Model)**.** *A computational model, or simply a model, is a computer program (or in general a Turing machine) that can perform computations or arithmetic operations on data.*

In the context of this article, a computational model can be as simple as a computer program that performs basic arithmetic gates/operations on numbers (e.g., addition and multiplication), or basic but challenging operations (e.g., secure comparison a.k.a., Yao's Millionaires' problem [4], [5]). A computational model can also be as complicated as a deep and very large artificial neural network (such as large language models with billions of parameters) that performs extensive computations on vectorized data and tensors. Definition 1 is the base definition which we will use for the next definitions.

Definition 2 (Zero-Knowledge Model)**.** *A zero-knowledge computational model, or a zero-knowledge model, is a computational model that gains no knowledge about the data, prompts, or queries that the model receives and processes.*

A zero-knowledge computational model (zk-model) is essentially a computational model that cannot gain any knowledge about the private data which is given to it for processing. One might wonder how it could be possible to perform computations on data without gaining knowledge about the data. The answer would be that this paradigm of computation has been the motivation behind many attempts on secure or privacy-preserving computation (see, e.g., seminal works [4], [25] and [6], [7] among other references listed throughout this article).

It should be noted that the notion of zero-knowledge in this article is a little different than zero-knowledge in other contexts in which there is a prover and a verifier. In such contexts, typically there is a prover and a verifier, where the prover wants to prove possessing some statement without revealing any information about the statement. However, here in the context of this article zero-knowledge refers to the notion of performing some computation on data without revealing any knowledge about the data (thus zero-knowledge). This terminology is to emphasize the fact that in this context there is a client (or user) who has some query or prompt (possibly with private data). The user wants to send the query or prompt to a computational model so that the model can process it. But, the user is not interested in revealing any information about the query. This setting is similar to the scenario of private information retrieval (PIR) or private set intersection (PSI) which are well-studied problems in secure computation on databases [26] and [27].

Definition 3 (Privacy-Preserving Model)**.** *A privacy-preserving computational model, or a privacy-preserving model, is a zero-knowledge model that does not reveal any information about the private data, prompts, or queries that the models receives and processes.*

Definition 3 defines and highlights the requirements that our desired model should meet. A privacy-preserving computational model is a model which has two important properties: 1) it is zero-knowledge; and 2) it does not reveal any information about the private data (or private prompts or queries) it receives and processes. It should be emphasized that these two requirements are independent. Importantly, as we will discuss later in Section III, a privacy-preserving model can be a model which is trained on publicly available data, then is compressed and would be utilized locally by a client without sending any data to any (trusted or untrusted) third parties. While the zero-knowledge property of the model stresses more on the fact that the local model itself should not gain any knowledge about the queries or prompts that it receives and processes.

Next, we discuss how a computational model can be transformed to a privacy-preserving computational model. The following proposition provides the theoretical guarantees that any computational model can be transformed to (or converted into) a model that it does not gain any knowledge about the *data* that it processes and does not reveal any information about that data. Furthermore, the proposition pinpoints the general-purpose cryptographic constructions that can be used for the transformation or conversion of the computational model.

Proposition 1 (Privacy-Preserving Model Transformation)**.** *Any computational model can be transformed to a privacy-preserving model using appropriate cryptographic constructions. This can be achieved thanks to constructions such as secure computation and secure function evaluation (SFE) techniques.*

Proposition 1 essentially states that for any computational model we can design a privacy-preserving version of that, in the sense that the model can do its job without gaining any knowledge about the private data that it receives and without revealing the data to any third parties (regardless of the third party being trusted or untrusted). This important can be achieved thanks to cryptographic constructions, e.g., private polynomial computation (PPC) [9], fully homomorphic encryption (FHE) [6]–[8], garbled circuits [4], [5] or universal circuits [16] (see also [17]), secure function evaluation (SFE) [19], and other encoding and encryption techniques that we will discuss in the next section. It should be noted that Proposition 1 is in spirit of significant works and research on secure computation and secure function evaluation constructions (including the well-known GMW protocl [25] and the OT protocols [15]).

III. Privacy-Preserving LLMs (PPLLMs) using Secure Polynomial and Matrix-Vector Delegations

In this section we present some ideas that are useful for transforming or turning a computational *model* into a privacy-preserving computational model. Particularly, we illustrate how to instantiate a privacy-preserving version of a typical large language model (PP-LLM).

The main idea for constructing a privacy-preserving model is as follows. The two main components, i.e., *data* and the computational *model* need to be re-engineered. Regarding the computational *model*, it needs to be re-designed while considering and applying secure function evaluation (SFE) techniques at the functionality level, e.g., using PPC constructions [9], FHE techniques [6]–[8] or garbled circuits [4], [5]. In addition, the model needs to be compressed and the user should run a local copy of the trained model. This allows the users to have more/better control over their data and the computational model.

For the other component, i.e. *data*, well-designed data and feature engineering techniques should be considered at the level of data preparation; some prompt engineering techniques should be considered for query submission as well as for decoding the model's responses to the users' query. Furthermore, we should differentiate between (and separate) the publicly available data (on which the model is trained) and the user's data that is sent to a trained model (which consist of user's queries or prompts, possibly with private data). This step can consist of encoding the private data in some embedding spaces, e.g., vector spaces [24] or polynomial spaces [23]. As a result of this step, the prompt or query is encoded as a numerical vector or a polynomial in an embedding space.

Encoding and embedding data in some embedding spaces provides the opportunities to apply appealing and powerful transformations or techniques on the data. It also facilitates applying other security/privacy countermeasures on the encoded data. For example, one can apply some differential privacy technique (e.g., the sparse vector technique, a.k.a., the SVT [18]) on data to add some randomness to the private data. It is also possible to take a more protective approach and completely encrypt the private data using (fully) homomorphic encryption techniques for performing computation on encrypted data (e.g., by applying some of the schemes suggested in the draft of FHE standard [8]).

We would like to emphasize that encoding data in polynomial spaces [23], enables applying a wide variety of cryptographic techniques on the encoded data (e.g., private polynomial computation [9], polynomial delegation [28], [29], data provenance using provenance polynomials or semirings [30], [31], polynomial commitments [28], secret sharing [12], Reed-Solomon and MDS codes [32] for PIR [33], [34], quadratic arithmetic programs (QAP) [35] and R1CS [36] for applying ZKP protocols, polytopes and polynomial zonotopes for verification and increasing robustness of AI models [37], [38]), etc.

A. Polynomial and Matrix-Vector Delegations as Novel Prompt Engineering Solutions for PPLLMs

Polynomial spaces [23] (including rings of polynomials and polynomials as a mathematical structure) have shown to be of much applicability in different scientific areas, including in secure and verifiable computation. For example, most of the promising fully homomorphic encryption schemes are based on the cyclotomic ring of polynomials and the Ring-LWE

problem [39], which is a well-known and extensively-used problem in applied cryptography. Polynomials have also been extensively utilized in zero-knowledge proof systems as a powerful tool for arithmetization of computation (see e.g., QAP [35] and R1CS [36]). Other applications of polynomials include private polynomial computation (PPC) [9], provenance polynomials and semirings [30], [31], polynomial classifiers [40], polynomial neural networks [41]–[43], and a more recent application of RS codes called FRI protocol (fast Reed-Solomon Interactive Oracle Proof) [44], which is useful for zero-knowledge proof (ZKP) systems.

Verifiable polynomial delegation (VPD) [28], [29] is yet another polynomial-based construction which can potentially be a suitable technique for data privacy in outsourced computations to untrusted parties [45] as well as for zero-knowledge proof systems [29]. Polynomial delegation is a technique that allows a party to delegate the computation (evaluation) of a polynomial to a third party (also to verify the computation result). This technique has been very useful for computation outsourcing and verifiable computation (see e.g., [45]–[47]).

A recent and appealing research trend is to add privacy-preserving capabilities to polynomial delegation techniques [45], [48]. There are different approaches for adding privacy-preserving capabilities to polynomial delegation techniques [45], [48]. Some examples in the literature include using fully homomorphic encryption techniques (e.g., [49], [50]), multiparty computation (e.g., [51]), secret sharing (e.g., [52]), homomorphic hashing technique (e.g., [53]), linearly homomorphic structure-preserving signatures (e.g., [54]), homomorphic authenticated encryption (e.g., [55]), privacy-preserving homomorphic MACs [56], [57] etc. In this article we consider techniques similar to those proposed in [45], [48].

B. Privacy-Preserving Transformations of Polynomial and Matrix-Vector Functions

Here we briefly describe the interesting privacy-preservation approach of [45], as it supports the delegation of both polynomial and matrix-vector functions. We then present our solution for privacy-preserving large language models (PP-LLMs), which can be instantiated using the transformations described in [45] or other similar constructions, e.g., those of [46], [48] or [58].

Researchers in [45] have proposed two transformations for privacy-preserving delegation of polynomial and matrix-vector functions on inputs in finite fields (e.g., Z_q for a prime number q). The transformation for privacy-preserving delegation of polynomial functions uses a noisy encoding algorithm and relies on hard computational assumptions, i.e., noisy curve reconstruction assumption [59] (which in turn is based on noisy polynomial reconstruction [60]). The transformation consists of several steps, including key generation, problem generation, compute and verify algorithms. The main steps of the polynomial transformation are summarized below [45] (detailed steps and description of the transformation for polynomial functions can be seen in Section 3.3 of [45]):

- KeyGeneration: generates a pair of public and private key.
- Problem Generation: generates a public noisy variation of the input vector and a private value for its reconstruction.
- Compute: computes an encoded variation of the function output.
- Verify: verifies the computation result.

The transformation for matrix-vector functions (i.e., matrix-vector multiplication) relies on other cryptographic constructions, i.e., pseudo-random functions (PRFs), somewhat or partial homomorphic encryption (SHE), and homomorphic hash functions [45]. The main steps of this transformation are similar to the steps of transformation for the polynomial functions (described above), where each step consists of several other steps pertinent to the cryptographic construction being used [45].

C. A Privacy-Preserving Transformation for AI and LLM Models

In this section we provide a transformation for turning a typical AI and LLM model into a privacy-preserving one.

In data-driven and model-driven computation scenarios, there are typically two key components and two or more parties involved. For example, for the case of AI and large language models (LLMs), there are two key parts, namely the user's data (e.g., query or prompt) and the trained AI or LLM model. The parties in this scenario include the user (client), the computational model as an AI agent, and sometimes the model owner (typically the developers and maintainers of the computational model). Our idea for turning an AI or LLM model to a privacy-preserving one is to encode and embed the user's data in polynomial spaces [23] and vector spaces [24]; and to represent the model as polynomial and matrix-vector functions.

Our proposed transformation takes place in two steps. The user's data is encoded in polynomial and/or vector spaces [23] and [24]. This can be achieved after applying feature extraction and vectorization techniques on the user's query or prompt. For this purpose common techniques such as Word2Vec [61], GloVe [62], or other similar techniques can be utilized. The trained AI or LLM model needs to be represented as polynomial and/or matrix functions. To do this, the model might need to be compressed first. Model compression [63] is a technique for reducing the number of parameters of a large computational model using different techniques such as pruning or knowledge distillation (see e.g., [64], [65]). It should be noted that some compression approaches can significantly decrease the size of computational model. For example, researchers in [66] have discussed that hashing techniques [67] along with embedding tables and parameter sharing setups have the potential to compress a computational model $10000\times$.

Once the data is encoded in polynomial and vector spaces and the model is compressed and represented as polynomial and matrix functions, the prediction task of the model is reduced to evaluation of polynomial and/or matrix-vector functions. This can be done using privacy-preserving delegation of polynomial and matrix functions [45], [48]. With these

pre-processing on data and computation model, our proposed transformation for privacy-preserving large language model is as follows:

- Feature Extraction: extracts the features of the user's query or prompt.
- Vectorize & Encode the Data: embeds the extracted features in a vector and/or polynomial embedding space.
- KeyGeneration: generates the required cryptographic parameters (including public and private keys).
- Encrypt: applies approperiate encryption techniques on the encoded data.
- Compress and Arithmetize the Model: compresses the computational model and provides a polynomial and/or matrix-vector representation of the compressed computational model.
- Delegate the Computation: passes the encoded/encrypted data to the arithemtized computational model.
- Evaluate & Compute: computes the arithemtized model on the encoded (encrypted) data, e.g., on encoded queries or prompts.
- Decode & Decrypt: decodes (and decrypts) the response of the computational model to the user's query or prompt.

The above steps provide a general-purpose solution for transforming a computational model to a privacy-preserving computational model. Depending on the computational model different customized cryptographic constructions can be used for instantiating a privacy-preserving version of the model.

It should be noted that most of the computations in artificial neural networks, which are commonly used in artificial intelligence and large language models, are readily in the form of matrix-vector functions. This can provide a significant gain thanks to fast computational frameworks such as PyTorch, TensorFlow, or Google's XLA (accelerated linear algebra framework). Furthermore, other commonly-used building blocks in neural networks (including activation functions such as the ReLu function) can be approximated using polynomial function techniques such as Taylor series or Chebyshev polynomials. For approximating the (ReLu) activation function in homomorphic evaluation of neural networks, interested readers might refer to [68], [69] or [70]–[72], that have provided solutions based on Taylor series or Chebyshev polynomials.

IV. Discussion and Future Research Directions

The rise of artificial general intelligence (AGI) models, including the large language models (LLMs) such as OpenAI's ChatGPT, DALL-E, Google's PaLM-E and Bard, have increased the concerns about the risks that AI models can potentially have to the society and the future of humanity. Among other risks and concerns are risks to the security and privacy of data (as well as users' privacy).

When it comes to data security and privacy (and users' privacy), cryptographic solutions are among the most useful and effective solutions. There are different cryptographic constructions that can be applicable for data privacy in ML, AI and particularly large language models (LLMs). Such constructions generally fall under the category of secure computation (a.k.a., privacy-preserving or secure multiparty computation). Some seminal and important landmarks in secure computation include the GMW protocl [25], the interesting idea behind fully homomorphic encryption [6], secure multiparty computation (MPC) [73], [74], oblivious transfer protocols [15], garbled circuits [4], [5] or universal circuits [16], secret sharing [11], [12], zero-knowledge proof (ZKP) [10], differential privacy [18], [22], Blockchain and decentralized computation [75].

The aforementioned references have laid the foundations, perhaps, for the next generation of computing systems, i.e., secure and verifiable computation that allows parties to perform computation on their data without revealing their data. With these fundamental works in mind, there are different avenues for future research and works.

One direction is to evaluate how well and efficient the existing secure computation techniques perform for privacy-preserving artificial intelligence models (including large language models). Particularly, the interaction of a user with a large language model can be modeled as a two-party computation scenario (2PC), wherein the model owner (typically the company that develops and hosts the trained model) is one party; and the second party is the user or client who sends prompts and queries to the model. This scenario is also closely related to private information retrieval (PIR) [76], in which there is a database server and a client who sends queries to the database. PIR is a fairly well-researched application of privacy-preserving computation; and this allows one to apply PIR techniques on LLM models. The similarity between PIR and PPLLMs is that in both cases there are some user(s) who want to submit queries or prompts to the database or the language model. But the user is not willing to reveal their private data to the database server or to the language model. Similarly, the owner of the computational model or the database server might not be interested in revealing the algorithms behind their models.

Another line of research is with regard to the notion of robustness and trustworthy AI and LLM models. Large language models turn out to suffer from some intrinsic shortcomings, such as hallucination, algorithmic bias and biasedness (being biased), alignment, uncertainty, and sometimes inaccuracy. Soft security measures, e.g., trust and reputation, [20], [21] along with secure computation constructions, e.g., secure trust evaluation (STE) [77], can be used for addressing such fundamental issues of AI or LLM models. For instance, trust and reputation systems can be used for categorizing different sources of knowledge on which a language model is trained. Then by training AI and language models on more trustworthy data (originated from trustworthy sources), the predictions of the model can potentially be more reliable. In addition, uncertainty quantification techniques [78], e.g., the well-known concept of entropy in information theory, can be used for assessing the uncertainty of AI, ML, and LLM models.

Some interesting and less explored lines of research might potentially be to study the integration of polynomial networks [43] and polynomial classifiers [40] with secure computation

constructions based on polynomials (e.g., secret sharing [11], [12], FHE schemes based on the cyclotomic ring of polynomials [8], and private polynomial computation [9]). Polynomials have very appealing properties and have been utilized for various applications quite extensively (see for example, polynomial neural networks (PNN) [41], orthogonal polynomial neural networks ([79], [80]), deep polynomial neural networks [42], and polynomial classifiers [40]). As encoding techniques, polynomials enable encoding/encrypting the data (see e.g., Reed-Solomon Error-Correcting codes [32]); and as universal approximators [40], they enable approximating various functions. Yet, their capability for arithemtization of computation makes them appropriate tools for zero-knowledge proof applications (see e.g., QSP and QAP [35], QRP [81] and R1CS [36]).

Robustness analysis of neural networks and their verification are other interesting and important areas of research that seek more attentions, particularly because of extensive usage of neural networks in different domains of AI, e.g., in self-driving and autonomous vehicles. There are several works that have proposed some solutions for the verification and robustness of neural networks [82]–[84]. Among others, polytopes and polynomial zonotopes are some polynomial-based approaches that have been studied [38], [83]. Particularly, since polynomials can behave as a bridge between secure computation and other applications, polytopes and zonotopes can be a potential approach for designing secure verification methods for neural networks that can increase the robustness of AI systems. By unifying and mixing different secure computation and polynomial-based constructions with machine learning models, it is possible to build more secure and trustworthy machine learning and artificial intelligence models as well as efficient and privacy-preserving big data frameworks [85].

V. Conclusion

In this article we defined the concepts of zero-knowledge and privacy-preserving computational models. We discussed how polynomial and vector spaces (as encoding and embedding techniques) can be used for encoding and encrypting data; and how privacy-preserving delegation of polynomial and matrix-vector functions can be used for transforming a computational model into a privacy-preserving computational model. Furthermore, we highlighted some well-known cryptographic solutions that can be used for preserving data and users' privacy in artificial intelligence and large language models (LLMs).

There are various cryptographic solutions, e.g., private polynomial computation (PPC), fully homomorphic encryption (FHE), secure computation, secure function evaluation (SFE), garbled circuits (GC), verifiable secret sharing (VSS), that can be used for encrypting or encoding users' query or prompts before the prompts or queries are sent to a computational model, e.g., a large language model (LLM). On the other hand, soft security measures, e.g., trust and reputation, are helpful social mechanisms that can be used for modeling trustworthy data and constructing robust and trustworthy ML and AI models. Besides cryptographic solutions, compressing language models and cloning privacy-preserving models to clients computational devices can add another layer of privacy-protection. Yet, decentralized computational models trained using federated learning techniques on publicly available trustworthy data, along with reinforcement learning from human feedback (RLHF), can be potential trustworthy solutions that consider and incorporate human values.

References

[1] J. Rudolph, S. Tan, and S. Tan, "War of the chatbots: Bard, bing chat, chatgpt, ernie and beyond. the new ai gold rush and its impact on higher education," *Journal of Applied Learning and Teaching*, vol. 6, no. 1, 2023.

[2] T. H. Kung, M. Cheatham, A. Medenilla, C. Sillos, L. De Leon, C. Elepaño, M. Madriaga, R. Aggabao, G. Diaz-Candido, J. Maningo, *et al.*, "Performance of chatgpt on usmle: Potential for ai-assisted medical education using large language models," *PLoS digital health*, vol. 2, no. 2, p. e0000198, 2023.

[3] OpenAI, "Gpt-4 technical report," 2023.

[4] A. C. Yao, "Protocols for secure computations," in *23rd annual symposium on foundations of computer science (sfcs 1982)*, pp. 160–164, IEEE, 1982.

[5] A. C.-C. Yao, "How to generate and exchange secrets," in *27th annual symposium on foundations of computer science (Sfcs 1986)*, pp. 162–167, IEEE, 1986.

[6] R. L. Rivest, L. Adleman, M. L. Dertouzos, *et al.*, "On data banks and privacy homomorphisms," *Foundations of secure computation*, vol. 4, no. 11, pp. 169–180, 1978.

[7] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pp. 169–178, 2009.

[8] M. Albrecht, M. Chase, H. Chen, J. Ding, S. Goldwasser, S. Gorbunov, S. Halevi, J. Hoffstein, K. Laine, K. Lauter, *et al.*, "Homomorphic encryption standard," *Protecting privacy through homomorphic encryption*, pp. 31–62, 2021.

[9] N. Raviv and D. A. Karpuk, "Private polynomial computation from lagrange encoding," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 553–563, 2019.

[10] S. GOLDWASSER, S. MICALI, and C. RACKOFF, "The knowledge complexity of interactive proof systems," *SIAM journal on computing*, vol. 18, no. 1, pp. 186–208, 1989.

[11] G. R. Blakley, "Safeguarding cryptographic keys," in *Managing Requirements Knowledge, International Workshop on*, pp. 313–313, IEEE Computer Society, 1979.

[12] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[13] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable secret sharing and achieving simultaneity in the presence of faults," in *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, pp. 383–395, IEEE, 1985.

[14] A. Chandramouli, A. Choudhury, and A. Patra, "A survey on perfectly secure verifiable secret-sharing," *ACM Computing Surveys (CSUR)*, vol. 54, no. 11s, pp. 1–36, 2022.

[15] M. O. Rabin, "How to exchange secrets with oblivious transfer," *Cryptology ePrint Archive*, 2005.

[16] L. G. Valiant, "Universal circuits (preliminary report)," in *Proceedings of the eighth annual ACM symposium on Theory of computing*, pp. 196–203, 1976.

[17] H. Lipmaa, P. Mohassel, and S. Sadeghian, "Valiant's universal circuit: Improvements, implementation, and applications," *Cryptology ePrint Archive*, 2016.

[18] C. Dwork, M. Naor, O. Reingold, G. N. Rothblum, and S. Vadhan, "On the complexity of differentially private data release: efficient algorithms and hardness results," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pp. 381–390, 2009.

[19] T. Schneider, "Practical secure function evaluation.," in *Informatiktage*, pp. 37–40, 2008.

[20] L. Rasmusson and S. Jansson, "Simulated social control for secure internet commerce," in *Proceedings of the 1996 workshop on New security paradigms*, pp. 18–25, 1996.

[21] B. Yu and M. P. Singh, "A social mechanism of reputation management in electronic communities," in *Cooperative Information Agents IV-The Future of Information Agents in Cyberspace: 4th International Workshop, CIA 2000, Boston, MA, USA, July 7-9, 2000. Proceedings 4*, pp. 154–165, Springer, 2000.

[22] C. Dwork, "Differential privacy," in *Automata, Languages and Programming: 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II 33*, pp. 1–12, Springer, 2006.

[23] C. D. Godsil, "Polynomial spaces," *Discrete Mathematics*, vol. 73, no. 1-2, pp. 71–88, 1988.

[24] J.-L. Dorier, "A general outline of the genesis of vector space theory," *Historia mathematica*, vol. 22, no. 3, pp. 227–261, 1995.

[25] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game, or a completeness theorem for protocols with honest majority," in *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*, pp. 307–328, 2019.

[26] W. Gasarch, "A survey on private information retrieval," *Bulletin of the EATCS*, vol. 82, no. 72-107, p. 113, 2004.

[27] D. Morales, I. Agudo, and J. Lopez, "Private set intersection: A systematic literature review," *Computer Science Review*, vol. 49, p. 100567, 2023.

[28] A. Kate, G. M. Zaverucha, and I. Goldberg, "Constant-size commitments to polynomials and their applications," in *Advances in Cryptology-ASIACRYPT 2010: 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings 16*, pp. 177–194, Springer, 2010.

[29] J. Zhang, T. Xie, Y. Zhang, and D. Song, "Transparent polynomial delegation and its applications to zero knowledge proof," in *2020 IEEE Symposium on Security and Privacy (SP)*, pp. 859–876, IEEE, 2020.

[30] T. J. Green, G. Karvounarakis, and V. Tannen, "Provenance semirings," in *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 31–40, 2007.

[31] J. Závodnỳ, "On factorisation of provenance polynomials," in *3rd USENIX Workshop on the Theory and Practice of Provenance (TaPP 11)*, 2011.

[32] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the society for industrial and applied mathematics*, vol. 8, no. 2, pp. 300–304, 1960.

[33] R. Freij-Hollanti, O. W. Gnilke, C. Hollanti, and D. A. Karpuk, "Private information retrieval from coded databases with colluding servers," *SIAM Journal on Applied Algebra and Geometry*, vol. 1, no. 1, pp. 647–664, 2017.

[34] R. Tajeddine, O. W. Gnilke, and S. El Rouayheb, "Private information retrieval from mds coded data in distributed storage systems," *IEEE Transactions on Information Theory*, vol. 64, no. 11, pp. 7081–7093, 2018.

[35] R. Gennaro, C. Gentry, B. Parno, and M. Raykova, "Quadratic span programs and succinct nizks without pcps," in *Advances in Cryptology–EUROCRYPT 2013: 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings 32*, pp. 626–645, Springer, 2013.

[36] E. Ben-Sasson, A. Chiesa, M. Riabzev, N. Spooner, M. Virza, and N. P. Ward, "Aurora: Transparent succinct arguments for r1cs," in *Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part I 38*, pp. 103–128, Springer, 2019.

[37] N. Kochdumper, C. Schilling, M. Althoff, and S. Bak, "Open-and closed-loop neural network verification using polynomial zonotopes," in *NASA Formal Methods Symposium*, pp. 16–36, Springer, 2023.

[38] C. Schilling, M. Forets, and S. Guadalupe, "Verification of neural-network control systems by integrating taylor models and zonotopes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, pp. 8169–8177, 2022.

[39] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," *Journal of the ACM (JACM)*, vol. 60, no. 6, pp. 1–35, 2013.

[40] W. M. Campbell, K. T. Assaleh, and C. C. Broun, "Speaker recognition with polynomial classifiers," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 4, pp. 205–212, 2002.

[41] S.-K. Oh, W. Pedrycz, and B.-J. Park, "Polynomial neural networks architecture: analysis and design," *Computers & Electrical Engineering*, vol. 29, no. 6, pp. 703–725, 2003.

[42] G. G. Chrysos, S. Moschoglou, G. Bouritsas, J. Deng, Y. Panagakis, and S. Zafeiriou, "Deep polynomial neural networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 8, pp. 4021–4034, 2021.

[43] D. F. Specht, "Generation of polynomial discriminant functions for pattern recognition," *IEEE Transactions on Electronic Computers*, no. 3, pp. 308–319, 1967.

[44] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Fast reed-solomon interactive oracle proofs of proximity," in *45th international colloquium on automata, languages, and programming (icalp 2018)*, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[45] L. F. Zhang and R. Safavi-Naini, "Protecting data privacy in publicly verifiable delegation of matrix and polynomial functions," *Designs, Codes and Cryptography*, vol. 88, no. 4, pp. 677–709, 2020.

[46] D. Fiore and R. Gennaro, "Publicly verifiable delegation of large polynomials and matrix computations, with applications," in *Proceedings of the 2012 ACM conference on Computer and communications security*, pp. 501–512, 2012.

[47] S. Benabbas, R. Gennaro, and Y. Vahlis, "Verifiable delegation of computation over large datasets," in *Advances in Cryptology–CRYPTO 2011: 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings 31*, pp. 111–131, Springer, 2011.

[48] B. Song, D. Zhou, J. Wu, X. Yuan, Y. Zhu, and C. Wang, "Protecting function privacy and input privacy in the publicly verifiable outsourcing computation of polynomial functions," *Future Internet*, vol. 15, no. 4, p. 152, 2023.

[49] M. Barbosa and P. Farshim, "Delegatable homomorphic encryption with applications to secure outsourcing of computation.," in *CT-RSA*, vol. 7178, pp. 296–312, Springer, 2012.

[50] K.-M. Chung, Y. Kalai, and S. Vadhan, "Improved delegation of computation using fully homomorphic encryption," in *Advances in Cryptology–CRYPTO 2010: 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings 30*, pp. 483–501, Springer, 2010.

[51] P. Ananth, N. Chandran, V. Goyal, B. Kanukurthi, and R. Ostrovsky, "Achieving privacy in verifiable computation with multiple servers–without fhe and without pre-processing," in *Public-Key Cryptography–PKC 2014: 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings 17*, pp. 149–166, Springer, 2014.

[52] L. F. Zhang, "Multi-server verifiable delegation of computations: Unconditional security and practical efficiency," *Information and Computation*, vol. 281, p. 104740, 2021.

[53] D. Fiore, R. Gennaro, and V. Pastro, "Efficiently verifiable computation on encrypted data," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pp. 844–855, 2014.

[54] B. Libert, T. Peters, M. Joye, and M. Yung, "Linearly homomorphic structure-preserving signatures and their applications," *Designs, Codes and Cryptography*, vol. 77, pp. 441–477, 2015.

[55] C. Joo and A. Yun, "Homomorphic authenticated encryption secure against chosen-ciphertext attack," in *Advances in Cryptology–ASIACRYPT 2014: 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, ROC, December 7-11, 2014, Proceedings, Part II 20*, pp. 173–192, Springer, 2014.

[56] S. Li, X. Wang, and R. Xue, "Toward both privacy and efficiency of homomorphic macs for polynomial functions and its applications," *The Computer Journal*, vol. 65, no. 4, pp. 1020–1028, 2022.

[57] S. Li, X. Wang, and R. Zhang, "Privacy-preserving homomorphic macs with efficient verification," in *Web Services–ICWS 2018: 25th International Conference, Held as Part of the Services Conference Federation, SCF 2018, Seattle, WA, USA, June 25-30, 2018, Proceedings 16*, pp. 100–115, Springer, 2018.

[58] Q. Yu and A. S. Avestimehr, "Entangled polynomial codes for secure, private, and batch distributed matrix multiplication: Breaking the" cubic" barrier," in *2020 IEEE International Symposium on Information Theory (ISIT)*, pp. 245–250, IEEE, 2020.

[59] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, "Cryptography from anonymity," in *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pp. 239–248, IEEE, 2006.

[60] M. Naor and B. Pinkas, "Oblivious polynomial evaluation," *SIAM Journal on Computing*, vol. 35, no. 5, pp. 1254–1281, 2006.

[61] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems*, vol. 26, 2013.

[62] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[63] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541, 2006.

[64] F. Mireshghallah, A. Backurs, H. A. Inan, L. Wutschitz, and J. Kulkarni, "Differentially private model compression," in *Advances in Neural Information Processing Systems*.

[65] M. Gupta and P. Agrawal, "Compression of deep learning models for text: A survey," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 16, no. 4, pp. 1–55, 2022.

[66] A. Desai and A. Shrivastava, "The trade-offs of model size in large recommendation models: A 10000 × compressed criteo-tb dlrm model (100 gb parameters to mere 10mb)," *arXiv preprint arXiv:2207.10731*, 2022.

[67] A. Shrivastava, *Probabilistic hashing techniques for big data*. Cornell University, 2015.

[68] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*, pp. 409–437, Springer, 2017.

[69] J. S. Yoo, J. H. Hwang, B. K. Song, and J. W. Yoon, "A bitwise logistic regression using binary approximation and real number division in homomorphic encryption scheme," in *Information Security Practice and Experience: 15th International Conference, ISPEC 2019, Kuala Lumpur, Malaysia, November 26–28, 2019, Proceedings 15*, pp. 20–40, Springer, 2019.

[70] S. Obla, X. Gong, A. Aloufi, P. Hu, and D. Takabi, "Effective activation functions for homomorphic evaluation of deep neural networks," *IEEE Access*, vol. 8, pp. 153098–153112, 2020.

[71] R. Podschwadt and D. Takabi, "Classification of encrypted word embeddings using recurrent neural networks.," in *PrivateNLP@ WSDM*, pp. 27–31, 2020.

[72] E. Hesamifard, H. Takabi, and M. Ghasemi, "Cryptodl: towards deep learning over encrypted data," in *Annual Computer Security Applications Conference (ACSAC 2016), Los Angeles, California, USA*, vol. 11, 2016.

[73] M. Hastings, B. Hemenway, D. Noble, and S. Zdancewic, "Sok: General purpose compilers for secure multi-party computation," in *2019 IEEE symposium on security and privacy (SP)*, pp. 1220–1237, IEEE, 2019.

[74] R. Cramer, I. B. Damgård, *et al.*, *Secure multiparty computation*. Cambridge University Press, 2015.

[75] G. Zyskind, O. Nathan, and A. Pentland, "Enigma: Decentralized computation platform with guaranteed privacy," *arXiv preprint arXiv:1506.03471*, 2015.

[76] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *Journal of the ACM (JACM)*, vol. 45, no. 6, pp. 965–981, 1998.

[77] M. G. Raeini and M. Nojoumian, "Secure trust evaluation using multipath and referral chain methods," in *Security and Trust Management: 15th International Workshop, STM 2019, Luxembourg City, Luxembourg, September 26–27, 2019, Proceedings 15*, pp. 124–139, Springer, 2019.

[78] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, *et al.*, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Information Fusion*, vol. 76, pp. 243–297, 2021.

[79] C. K. Chak, G. Feng, and C. M. Cheng, "Orthogonal polynomials neural network for function approximation and system modeling," in *Proceedings of ICNN'95-International Conference on Neural Networks*, vol. 1, pp. 594–599, IEEE, 1995.

[80] A. Gu, T. Dao, S. Ermon, A. Rudra, and C. Ré, "Hippo: Recurrent memory with optimal polynomial projections," *Advances in neural information processing systems*, vol. 33, pp. 1474–1487, 2020.

[81] C. Ganesh, A. Nitulescu, and E. Soria-Vazquez, "Rinocchio: Snarks for ring arithmetic," *Cryptology ePrint Archive*, 2021.

[82] T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev, "Ai2: Safety and robustness certification of neural networks with abstract interpretation," in *2018 IEEE symposium on security and privacy (SP)*, pp. 3–18, IEEE, 2018.

[83] Y. Zhang and X. Xu, "Safety verification of neural feedback systems based on constrained zonotopes," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 2737–2744, IEEE, 2022.

[84] G. Anderson, S. Pailoor, I. Dillig, and S. Chaudhuri, "Optimization and abstraction: a synergistic approach for analyzing neural network robustness," in *Proceedings of the 40th ACM SIGPLAN conference on programming language design and implementation*, pp. 731–744, 2019.

[85] M. G. Raeini and M. Nojoumian, "Privacy-preserving big data analytics: from theory to practice," in *Security, Privacy, and Anonymity in Computation, Communication, and Storage: SpaCCS 2019 International Workshops, Atlanta, GA, USA, July 14–17, 2019, Proceedings 12*, pp. 45–59, Springer, 2019.