PARADIGM SHIFT FROM VAGUE LEGAL CONTRACTS TO BLOCKCHAIN-BASED SMART CONTRACTS

Kritagya Raj Upadhyay

Dissertation Prepared for the Degree of

DOCTOR OF PHILOSOPHY

UNIVERSITY OF NORTH TEXAS

July 2023

APPROVED:

Ram Dantu, Major Professor
Yanyan He, Co-Major Professor
Eduardo Blanco, Committee Member
Cihan Tunc, Committee Member
Gergely Záruba, Chair of the Department of Computer Science and Engineering
Paul S. Krueger, Dean of the College of Engineering
Victor Prybutok, Dean of the Toulouse Graduate School

Upadhyay, Kritagya Raj. *Paradigm Shift from Vague Legal Contracts to Blockchain-Based Smart Contracts*. Doctor of Philosophy (Computer Science and Engineering), July 2023, 154 pp., 13 tables, 48 figures, 192 numbered references.

In this dissertation, we address the problem of vagueness in traditional legal contracts by presenting novel methodologies that aid in the paradigm shift from traditional legal contracts to smart contracts. We discuss key enabling technologies that assist in converting the traditional natural language legal contract, which is full of vague words, phrases, and sentences to the blockchain-based precise smart contract, including metrics evaluation during our conversion experiment. To address the challenge of this contract-transformation process, we propose four novel proof-of-concept approaches that take vagueness and different possible interpretations into significant consideration, where we experiment with popular vendors' existing vague legal contracts. We show through experiments that our proposed methodologies are able to study the degree of vagueness in every interpretation and demonstrate which vendor's translated-smart contract can be more accurate, optimized, and have a lesser degree of vagueness. We also incorporated the method of fuzzy logic inside the blockchain-based smart contract, to successfully model the semantics of linguistic expressions. Our experiments and results show that the smart contract with the higher degrees of truth can be very complex technically but more accurate at the same time. By using fuzzy logic inside a smart contract, it becomes easier to solve the problem of contractual ambiguities as well as expedite the process of claiming compensation when implemented in a blockchain-based smart contract.

ACKNOWLEDGMENTS

First and foremost, I express my wholehearted gratitude to my esteemed mentor, advisor, and dissertation committee chair, Dr. Ram Dantu, for providing me the unwavering guidance since the beginning of my journey here at the University of North Texas. He has provided me with valuable knowledge and academic expertise and played a crucial role in shaping my career growth. I have always felt very fortunate to have such a visionary, encouraging, and compassionate advisor who always goes above and beyond his role as a professor.

Similarly, I would also like to express my deep appreciation to my respected co-advisor, Dr. Yanyan He, for her knowledgeable mentorship and invaluable support. In the same way, I would like to sincerely thank my respected dissertation committee members: Dr. Eduardo Blanco and Dr. Cihan Tunc, for their expert guidance and insightful advice. I am also incredibly grateful to Dr. Kirill Morozov for the numerous resourceful tips, opportunities, and recommendations he has generously provided me.

I would like to convey my warm gratitude to the professors and staffs of the Department of Computer Science and Engineering, and for a pleasant ambience to finish my journey here as a doctoral student. I would also like to thank my friends from the Network Security Lab for their collaborations, genuine suggestions, and feedback on my research works.

Lastly, but importantly, I am extremely grateful to my family, without whose love and support, this significant accomplishment in my life would not have been attainable. I am forever indebted to my loving parents, Kushal Raj Upadhyay and Arati Upadhyay, for inspiring me to embark on this endeavor and for all the countless sacrifices they selflessly made to facilitate the achievement of this long-cherished dream. I want to thank my dearest grandmothers, Sita Devi Upadhyay and Sharada Shrestha, my beloved sister, Akriti Sharma, and all my supportive aunties and uncles. I am deeply grateful to all my friends and everyone I have crossed paths with for their constant belief and support in me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

2D Two-dimensional

3D Three-dimensional

AI Artificial intelligence

AR Augmented reality

BERT Bidirectional Encoder Representations from Transformers

CF Certainty factor

CFG Control flow graph

CLI Command-line interface

COG Center of gravity

COS Center of sum

DAO Decentralized autonomous organization

DeFi Decentralized finance

EOA Externally owned account

ETH Ether

EVM Ethereum Virtual Machine

GPT-3 Generative Pre-Trained Transformer 3

HD Hierarchical deterministic

IDE Integrated development environment

ISP Internet service provider

IoT Internet of things

MAX Maximum operator

MF Membership function

MIN Minimum operator

MR Mixed reality

NFT Non-fungible token

NLP Natural language processing

PoS Proof-of-stake

PoW Proof-of-work

RQ Research question

SC Smart contract

SLA Service-level agreement

TXN Transaction

USD United States dollar

UTC Coordinated Universal Time

VAT Value-added tax

VR Virtual reality

XR Extended reality

CHAPTER 1

INTRODUCTION AND MOTIVATION[1]

1.1. Digital Economy Revolution

First Industrial Revolution started when we witnessed the paradigm shift from an agricultural and handicraft economy, where people used to produce goods by hand, to one where people started to produce goods by machines. This was when the first automation started on a mass scale in the late 1700s, and hence we used the phrase "industrial revolution" [51]. Similarly, in the late 1800s, we climbed one more step of development and built the electricity, telegraph, and railroad networks for a faster increase in productivity and economic growth, which is known as the Second Industrial Revolution [121]. Likewise, after 100 years, in the late 1900s, we saw an immense advancement in technology, especially in computer and communication technologies and production processes, where the shift took place from mechanical, analog, and electrical technology to digital electronics technology. We call this period the Third Industrial Revolution or Digital Revolution [87]. Finally, we are in what is known as the Fourth Industrial Revolution. Also known as Industry 4.0 [181], we believe that the boundaries between the physical, digital, and biological worlds will be hazy during this revolution as the world is already trending rapidly towards smart automation and data exchange, and due to this, the technologies are expected to cause disruption in every aspect of the economy. As humans are already connected to each other by billions of mobile devices, with the highest processing and storage capacity ever, and unlimited access to information, these advancements in technology are even augmented with emerging technologies such as blockchain, artificial intelligence, extended realities, the internet of things, and so on. Although it is in a germination phase now, digital economy [28] has a huge potential to be one of the core parts of our fourth industrial revolution as it allows users to

[1]Portions of this chapter are reproduced from K. Upadhyay, R. Dantu, Y. He, A. Salau and S. Badruddoja, "Paradigm Shift from Paper Contracts to Smart Contracts," 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), Atlanta, GA, USA, 2021, pp. 261-268, doi: 10.1109/TPSISA52974.2021.00029.

have unprecedented real-time interactions and decentralized experiences through the use of current and future integrated digital platforms such as blockchain, smart contracts, artificial intelligence, internet of things, and so forth.

1.1.1. Customer-Centric Economy

Just like technology, the economy is also ever-changing. Due to changing dynamics of the economy, the organizations providing services and doing business have already shifted their priorities to the customers. Instead of retaining complete power control over themselves and having a long line of bureaucracy [45], business organizations and institutions have been focusing on decentralization. This has been allowing the customers to gain more benefits as, due to the decentralization approach, they can avoid the legal quagmire of mediators, hence saving a lot of time and money. In addition to decentralization, due to the usage of state-of-the-art technologies in recent times, the whole economy is reaping the benefits of automation, accessibility, and security. For instance, when one person is selling the land to another person, then, instead of going to the office, standing in a long queue, and waiting for a long time due to the intermediaries and bureaucracy taking over the process, it is going to happen almost instantly when compared. Therefore, the main objective of today's and the future economy is to be more customer-centric and always put the customer first [145].

1.1.2. Digital Assets and Their Digital Owners

Digital assets are something that has value, are uniquely identifiable, and are stored in the crypto wallet and recorded on the blockchain ledger [30]. The person or organization with the legal right and authority over digital assets is known as a digital owner. Digital assets include graphics, videos, documents, manuscripts, project files, etc. In this new era of the digital economy, there are many examples of digital assets, such as cryptocurrencies, Non-fungible tokens (NFTs), arts and collectibles, virtual real estate, in-game items, etc. The digital owners can use their digital assets for social networking, gaming, trading, remote working, and digital events [30].

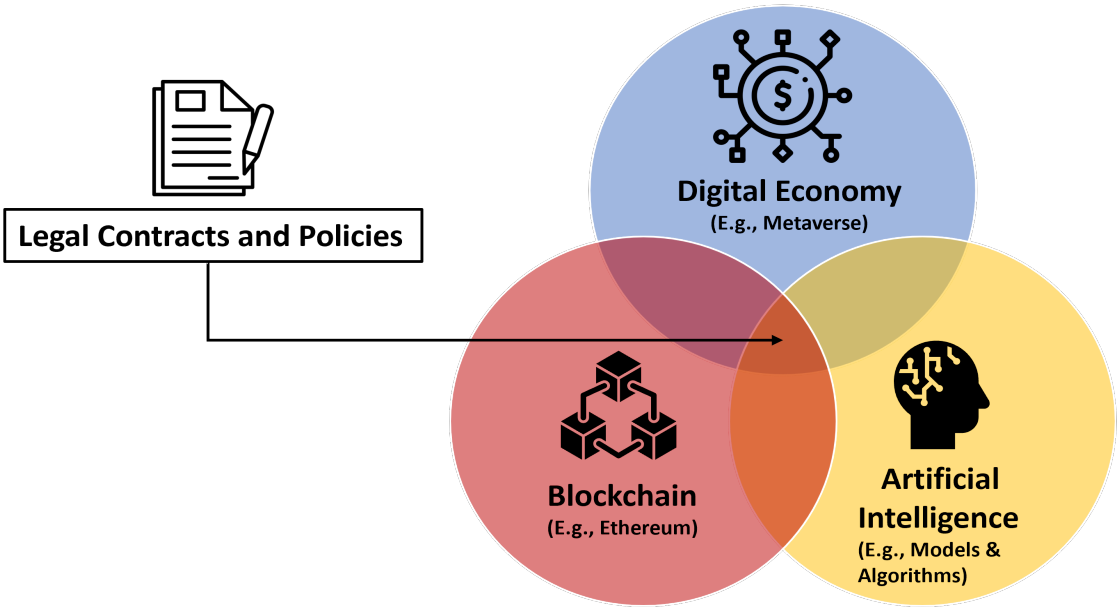

FIGURE 1.1. Legal Contracts sits at the core of three major technologies that are Digital/Virtual Economy, Blockchain, and Artificial intelligence [15].

1.1.3. Legal Contracts and Policies for Digital Economy

Legal contracts and policies always have been a core component of the economy regardless of the time frame of the industrial revolution [103]. There are different kinds of contracts that have been used for various purposes since the beginning of the first industrial revolution for binding two or more parties in a lawful agreement. With the economy, contracts also have evolved throughout time. In the digital economy, when the digital owners are trading with their digital assets, the process still needs to be legally bound with agreements between the involved parties that delineate the rules, rights, and obligations of each party involved. The recent advancement in contracts, known as the smart contract, is starting to get popular with the activities in the digital economy. A person selling their real estate title to the other person via a blockchain-based smart contract without the need for any intermediary where all the events, activities, and transactions of every kind are recorded immutably in a distributed, decentralized, and cryptographically secure manner in the blockchain is just one example out of many in digital economy [93].

1.2. Real-World Knowledge and its Impact on Legal Contracts

Humans are excellent at understanding a concept quickly through abstraction rather than precision. It is obvious that all human beings are from different backgrounds and occupations. Hence, it has always been more natural for humans to communicate in abstraction and approximation. Due to this reason, the interpretation of real-world knowledge in human beings can always differ. The incompleteness, inaccuracy, and inconsistency of real-world knowledge in human beings due to their different background and knowledge possession result in a phenomenon in semantics, metaphysics, and philosophical logic known as vagueness.

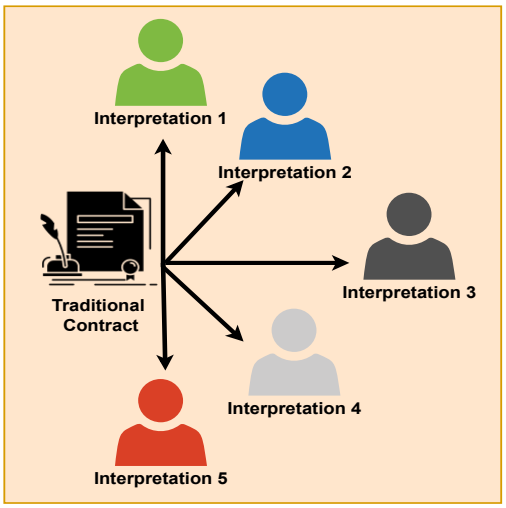

FIGURE 1.2. One traditional contract can create multiple interpretations in multiple parties due to its inherent vague nature

Vagueness arises when there are multiple meanings and interpretations from a single source and also arises when the given information is incomplete, inaccurate, and inconsistent [178]. Vagueness is inherent to traditional legal contracts as they are intentionally designed and drafted by lawyers in such a way that it includes vague words and phrases for their flexibility and open nature due to the uncertainty of the future [59]. Hence, the legal contract can either be vague and fuzzy when it is unclear what the other party means or ambiguous when one party is clear about what the other party means, but it could have several meanings. Nevertheless, these phenomena in legal contracts invite plenty of serious problems when the

involved parties have different interpretations, confusion, and misunderstandings.

Although real-world knowledge can have various ways of classification, the following concepts [4] are the ones that are more significant to understand and are also inherent to traditional legal contracts.

- Incompleteness: Incompleteness is a situation when either one or both involved parties in a legal contract do not possess all the parts and details that are needed for that particular event or action and therefore are indeterminate [13]. As the properties of real-world knowledge are continuously changing around us and will always vary, hence knowledge is always and inevitably be incomplete.
- Inaccuracy: Inaccuracy is a situation when something is not precisely correct. In the real world, although maximum precision in science is still the main objective to quest after, at the same time, philosophically, it is immeasurable, as real-world knowledge has unlimited degrees of truth. In some cases, real-world knowledge can be accurate but again be outdated by new pieces of evidence or developments [142].
- Inconsistency: Inconsistency is when there is a lack of coherence and agreement as various people from various backgrounds in the legal contract possess different levels of real-world knowledge [81]. As knowledge obtained by people varies according to country, culture, education, occupation, religion, gender, etc., it is not possible to always have a unanimous decision and get the same result. It is only possible to eliminate the inconsistency factor from a limited system.

1.3. Common Types of Language Modifiers in Legal Contracts

Language modifiers, also known as linguistic modifiers or language qualifiers, are words or phrases that modify the meaning in a sentence [138]. They alter the degree, certainty, specificity, and context of the information in a given sentence. Although language modifiers can have some more examples, they have been classified as the following major types that a legal contract can have, which invites the problem of the generation of multiple interpretations.

- Intensifiers: Intensifiers are words or phrases such as "very", "definitely", "absolutely", "extremely", etc., that strengthen and intensify another word's meaning.
- Diminishers: Diminishers are words or phrases such as "rather", "slightly", "a bit", "kind of", etc., that weaken another word's intensity and meaning.
- Hedging words: Hedging words are words or phrases such as "maybe", "might", "probably", "possibly", "roughly", "fairly", "likely", "appears", "think", etc., that indicates caution and tentativeness, and hence, uncertainty.
- Generalizers: Generalizers are the words or phrases such as "generally", "usually", "typically", "oftentimes", "broadly speaking", etc., that makes the meaning of sentence more general and inclusive.
- Specificity modifiers: Specificity modifiers are the words or phrases such as "particularly", "specifically", "precisely", etc., that narrow down the words to enhance the precision and specificity of a sentence.
- Vague words: Vague words are the words such as "good", "bad", "best", "reasonable", "interesting", etc., that only help in providing a general description of a sentence as it lacks clarity and precision.
- Ambiguity-inducing words: Ambiguity-inducing words are the words such as "bank", "book", "cool", "bat", etc., which have more than one meaning and create multiple interpretations. There are many types of ambiguity-inducing words as well. They are lexical ambiguity, syntactic ambiguity, antecedent ambiguity, temporal (time-based) ambiguity, and contract-reference ambiguity. The central focus for this chapter and the rest of the dissertation would be contract-reference ambiguity from ambiguity-inducing words.
  - Contract-reference ambiguity: The words such as "hereunder", "herein", "foregoing", "reasonable", "best efforts", "good faith", "may", and "might" are a few popular examples of contract-reference ambiguity with various degrees of truth. These words do not have any specific meanings. For example, suppose the contract drafter uses the word "hereunder". In that case, it either means it

applies for everything else that's below until the end of the contract or just for everything else until the end of that particular clause or section. Hence, these words generate many ambiguities in legal cases [3].

1.4. Research Motivation

1.4.1. What is a Contract?

An agreement that is in written or spoken form is known as a contract. A contract settles an agreement or a dispute between one or more parties, generally, an offeror and an offeree, since it is intended to be enforceable by law [43]. As a contract is legally enforceable, if one party fails to do what they have promised to do, the other party has the right to ask the court to enforce the agreement or award damages for injury sustained because the contract has been breached. All the responsibilities, do's, and don'ts are outlined in a contract. People have been using verbal agreements too, but the risk of disagreements and confusion can be reduced by only using written and tangible legal contracts.

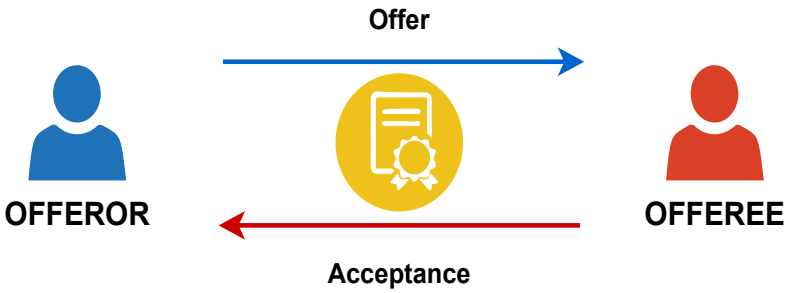

FIGURE 1.3. Entities and processes in a contract

For a contract to be legally enforced, it needs to meet four requirements which are as follows [103]:

- Agreement: The involved parties in a contract must reach a mutual agreement. An offeror will make an offer, and an acceptance will be replied to by an offeree.
- Consideration: Each agreement must be made in return for the performance of a legally sufficient act. An agreement lacks sufficient consideration if one party is not required to exchange something of legal value.

- Contractual capacity: All the involved parties in the legal contract must possess the entire legal capacity to fulfill contractual duties.
- Lawful object: The purpose of the contract must be legal.

1.4.2. Contractual Confusion due to Fuzzy Contracts

The most infamous characteristic of a legal contract is that it is unclear, vague, and full of jargon and hedge words. Hence, this almost always results in multiple interpretations from multiple parties. A vague contract means that a specific term, word, phrase, or definition is vague and has multiple meanings depending on a person's knowledge, experience, or perception [3].

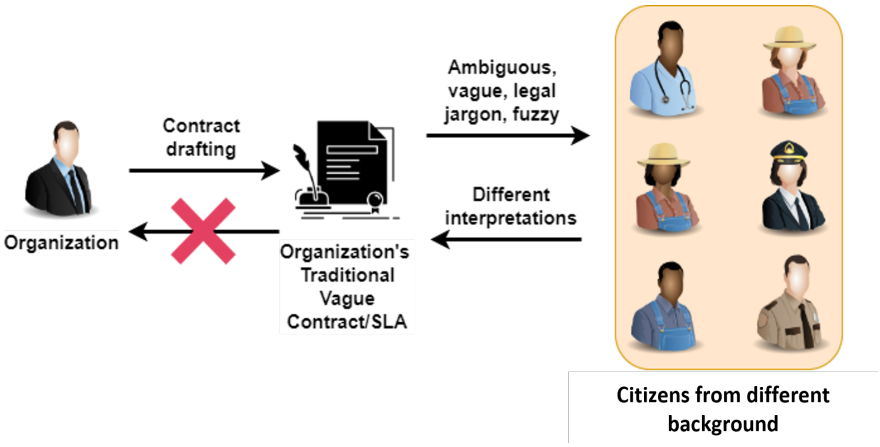

FIGURE 1.4. In an organization, the drafters of the legal contract intentionally and strategically put vague and fuzzy words as they want the legal contract to be as flexible as possible since the future is uncertain. Nonetheless, on the other hand, there are customers from different backgrounds with various levels of real-world knowledge who perceive the contents of the legal contract differently. The main cause for different interpretations is the presence of vague and fuzzy words and phrases that are put in legal contracts.

We have all been in that situation when we are constantly making complaints to the organization, especially internet service providers (ISPs), whenever their service gets constantly interrupted [31]. For the subscription and the payment the customers have made to the ISPs, they are entitled to provide the customers with very good service in all aspects,

at least what they have mentioned in their service-level agreement (SLA).

A service-level agreement (SLA) is also a kind of legal contract between a vendor and its customer which defines the quality of service that the vendor promises to provide to its customers in exchange for their subscription and payment [79]. If the vendor fails to provide the level of service to its customers that have been defined in their SLA, then the vendor will be penalized, and they will have to provide compensation to the customers that are also defined in the SLA. In other words, SLA is viewed as an important component of a vendor's legal contract.

Nevertheless, an SLA, also a kind of legal contract, has the inevitable problem of being vague and full of legal jargon that customers with different backgrounds and knowledge do not comprehend fully. Companies always prefer to talk in company-related or legal jargon, while customers who are laypersons prefer talking in everyday natural language [1]. For instance, generally, customers will make complaints such as "The service has been very slow and irregular for the last couple of months." These kinds of complaints might be genuine but present vague messages as, obviously, the customers cannot speak in legal or technical jargon. Due to this communication gap, when customers are trying to get further inquiries or support for their case, that leaves the customers even more puzzled. In that scenario, the customers who are not receiving proper service, as well as proper support from the company, are not getting any better help from the company-drafted SLA.

1.4.3. Remedies for Overcoming Confusion in Fuzzy Contracts

On the contrary, the main reason why smart contracts are becoming influential in the legal system is due to their explicitness, modernness, and innovative nature [128]. Although a conventional paper contract has always prevailed in the legal world since its origin for its enforceability, it still lacks a plethora of opportunities and advantages a smart contract can provide. Smart contracts are well suited for agreements without the presence of any third party or central authority. In contrast to traditional contracts, smart contracts are enforced by the blockchain system [176]. Hence, there would be no need for expensive court systems. This way, contracts become way cheaper as more peer-to-peer transactions can be governed

by smart contracts rather than by trust.

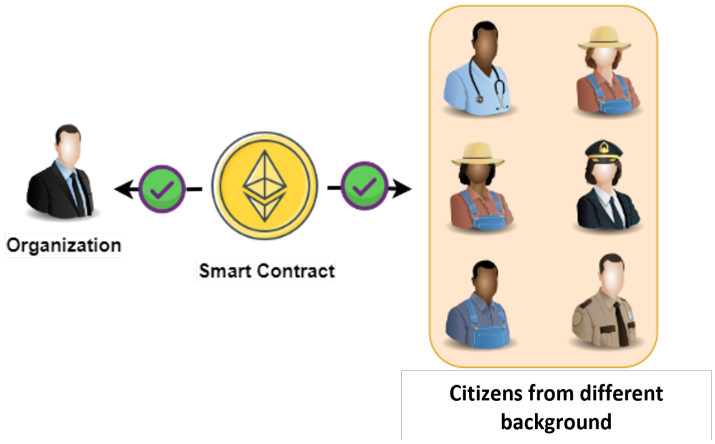

FIGURE 1.5. Unlike traditional legal contracts, which lack self-executability and are inherently vague in nature, a smart contract is an explicit computer program that resides in blockchain and is run when the specific and precise predetermined conditions are met. Due to this reason, even if the customers are from various backgrounds with diverse real-world knowledge, a smart contract does not create multiple interpretations for multiple people due to its explicit nature.

It can be challenging when there are contracts between organizations from two different countries with different languages and legislation. Researching and visiting different court systems can be very costly, and the judicial systems of one country will have limited power over companies from other countries. Nonetheless, blockchain-based smart contracts will not face these difficulties as they would not differentiate between any countries or their legislation and judicial practice. Enforcement of conventional contracts through a centralized authority such as a court system is not only very costly but also brings uncertainty to the result. There might always be that probability where lawyers will intentionally reveal some esoteric, vague, fuzzy, and ambiguous loophole concealed in the conventional contracts that entirely void the contract [56]. Even when the contract seems unquestionable and indisputable from the surface, the involved contracting parties rely on their court system's good faith to make sure that the contract is enforced.

TABLE 1.1. Applications of blockchain-based smart contract [144]

Features	**How can a Smart contract replace a Traditional legal contract?**
Explicit	The smart contract is written in the programming language, which is explicit and unambiguous and is able to be executed objectively.
Incorruptible	Once the traditional legal contract is changed into a smart contract, everything that is stored in a smart contract is incorruptible and immutable because blockchain is cryptographically secured.
Disintermediation	The blockchain and smart contract completely remove the trusted third party/arbiter/mediator.
Decentralization	The legal contract would not be controlled by a central authority.
Consensus	To maintain the integrity of the contract, all the parties and participants must come to a consensus.
Distributed	The ledger is maintained by all the participants and validators in the network, hence more secure.
Transparent	All entities, processes, and events that occur throughout the contract are transparent to all the participants in the network.
Faster settlement	Eliminating the intermediaries will settle everything faster and make the process cheaper as the parties involved in a legal contract do not have to pay extra legal fees to the lawyers.

During the COVID-19 pandemic, there were various cases of eviction of tenants by landlords in many states within the United States and in other countries as well [162]. Before letting someone live in a residence, landlords and tenants have signed a contract where they agree that if the rent is not paid on the due date, the landlord has the right to take action against the tenant in the form of eviction. Nonetheless, there were cases of tenants being evicted by their landlords during the pandemic, even when the tenants were willing to pay the rent. This means that a paper contract with the government seal or signature stamp

that can be torn apart at any time is not sensible and preferable when compared with smart contracts.

In the coming years, conventional paper legal contracts will definitely be replaced by smart contracts due to their faster settlement process, higher efficiency, and less vulnerability to legal loopholes [165]. In addition, smart contracts are less expensive, and they can reach across borders just as easily as within borders.

1.4.4. Narrowing the Focus for Research Motivation

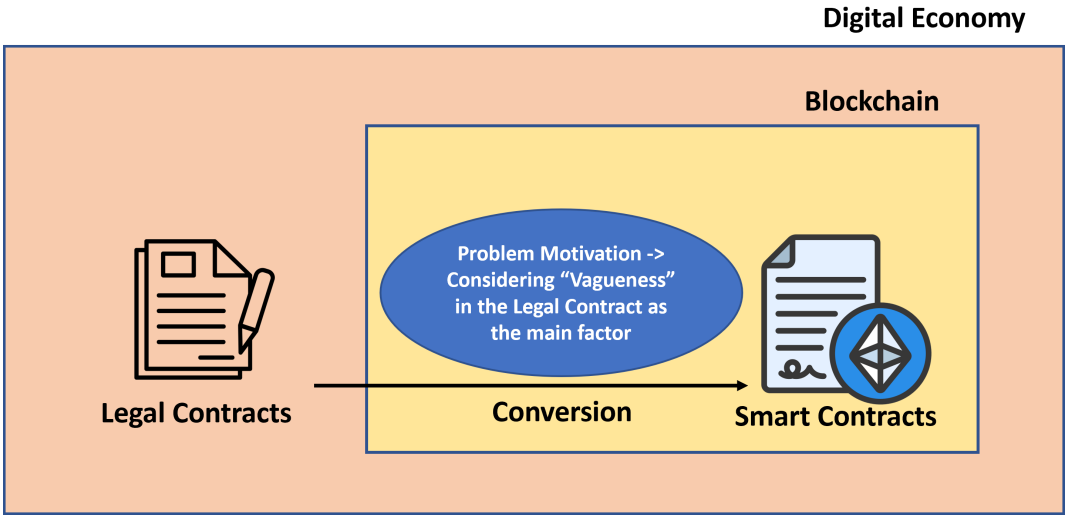

FIGURE 1.6. Due to the fact that digital entities inside the digital economy will be facing plenty of issues in using traditional legal contracts for the exchange of digital services or digital assets, there is a significant need to convert the vague legal contracts written in the natural language to the blockchain-based smart contracts. In this dissertation, we focus precisely on the medium that we have applied to do so, which is also the fundamental research motivation and definition.

Contracts have evolved through time, as also stated previously [165]. During all these years, contracts and policies have taken the forms of property lease and rental agreements, service-level agreements, software licensing agreements, non-disclosure agreement, end user license agreement, digital goods and services agreement, virtual currency exchange agreement, and so on [147]. It is also a fact that the majority of the time, these contracts have

always been vague, fuzzy, and incomprehensible [165].

For example, in the modern digital economy such as metaverse [38], there will be a presence of plenty of digital entities that will own digital assets. These digital assets owned by digital entities inside the digital economy environment will lead to various activities and transactions among the entities. In the metaverse, it will be challenging for digital entities to use traditional legal contracts in natural language, which has a plethora of problems, especially vagueness and lack of self-executability being the most. Therefore, there is an urgent need to convert vague legal contracts to blockchain-based smart contracts to make them vagueness-free and self-executable. Hence, with the use of several formal methods, proof-of-concept methods, and artificial intelligence methods, including fuzzy logic, vague legal contracts can be converted to blockchain-based smart contracts, as shown in figure [166]. In addition, blockchain also has the capability to preserve the integrity of generated smart contracts and artificial intelligence algorithms and maintains security from many kinds of attacks.

In spite of this overall example above, to narrow the focus of this dissertation's motivation, **we only emphasize the conversion processes of the vague legal contracts to the blockchain-based smart contracts**.

1.5. The Research Problems

The principal objective of this dissertation is to introduce a novel methodology that can be implemented by meticulously studying the nature of the traditional vague legal contracts and transforming them into blockchain-based smart contracts to see their performance in the blockchain network. To fulfill this objective, we have formulated the following research questions (RQs), which our work has addressed and answered:

- RQ1: How can we improve the worsening communication and trust between the legal contract drafters that draft vague contracts and policies with legal jargon and the consumers with limited real-world knowledge with no experience in the legal background? As explained above, this situation needs to be addressed in order for

blockchain-based smart contracts to prosper as the favored replacement to existing traditional vague legal contracts.

- RQ2: In what manner can we quantify and measure the interpretations since vague contracts can readily generate multiple interpretations among various individuals? It is essential to tackle this problem as without analyzing the multiple interpretations the traditional vague contract generates, it will only bring more uncertainty further.
- RQ3: In situations where customers/consumers demand compensation for inadequate service from vendors despite not completely comprehending the legal contracts, how is the certainty or confidence level of such claims determined and computed? Although, oftentimes, the victims of vague contracts and policies are the consumers, yet to be certain about their claims, the certainty or confidence level of such claims is crucial to be determined to ensure precision and equity.
- RQ4: How can we design an architectural model where a smart contract that consumers' limited real-world knowledge into perspective accounts for the vagueness factor and allows the smart contract to make decisions based on the fuzzy linguistic descriptors? This situation requires attention as a truly smart contract will only be completely smart if the contract can understand the vague and fuzzy linguistic descriptors as inputs that customers/consumers are accustomed to while communicating.

1.6. Summary of Specific Contributions

A brief summary of the specific contributions of this dissertation is provided below:

(1) The novel analytical examination and analysis has been delivered with a unique architectural model to aim for the elimination of vagueness, fuzziness, and legal jargon present in the traditional legal contracts for the translation and transformation into the blockchain-based smart contracts [165].

(2) The architecture with a methodology has been introduced that is able to study the vagueness of the traditional legal contracts for the generation of the possible human interpretations, quantification, and conversion of the generated possible interpre-

tations into smart contracts to measure the vagueness index with implementation results and observation [167].

(3) A novel architectural model with the methodology designed to classify the vague words from non-vague words in service-level agreements using an effective machine learning algorithm and then generating possible interpretations from the vague words to learn which service-level agreement when translated into the smart contract is vague and uncertain has been portrayed with experimental results and observations [164].

(4) A flexible mathematical model based on fuzzy logic [186] that helps in understanding and quantifying the certainty level [36] of the customers' claims on compensation based on their complaints has been portrayed with an example.

(5) A successful implementation of the architectural model is presented that solves the problem of the customers who are victims of the vague contracts and arduous bureaucratic system by incorporating the idea of fuzzy logic inside blockchain-based smart contract that can make claiming compensation an easy task [166].

(6) An exhaustive discussion regarding the unprecedented security risks and introduction to novel architecture for the security of digital economy such as metaverse by ensuring multimodal secured auditing has been made [163] .

1.7. Organization of the Dissertation

The rest of the dissertation is organized as follows:

- Chapter 2: This chapter presents the thorough background and existing literature surveys related to this dissertation.
- Chapter 3: We examine the vagueness index of each interpretation and the behavior of the translated interpretation when translated into the blockchain in this chapter.
- Chapter 4: In this chapter, we inspect the vagueness and uncertainty levels in SLAs from popular ISP vendors by translating them into smart contracts.
- Chapter 5: We apply the weighted fuzzy reasoning technique to study the certainty level of claims made by consumers in this chapter.

- Chapter 6: We incorporate the notion of fuzzy logic inside a smart contract in this chapter and study the behavior of these fuzzy logic-based smart contracts in blockchain networks.
- Chapter 7: This chapter presents a security analysis by portraying a security model for smart contracts and the digital economy and highlighting significant security vulnerabilities and their prevention remedies.
- Chapter 8: Finally, we conclude the dissertation in this chapter by summarizing the contributions of each chapter, challenges faced during the implementations, and possible directions for future work.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW[1]

2.1. Background Definitions

To begin with, let us define and explain some of the fundamental terminologies briefly that we will often be using in this dissertation.

Definition 2.1. **Blockchain.** Blockchain is a shared, distributed, decentralized, and immutable digital ledger that aids the process of recording and tracking transactions and assets [127]. An asset can be either tangible or intangible, and digital [19]. For instance, real estate properties, cash, and cars can be tangible assets that can be recorded and tracked inside the blockchain. On the other hand, intangible and/or digital assets can be digital and intellectual properties, copyrights, patents, non-fungible tokens, and so on. Basically, blockchain is a network of the "chain or sequence of blocks" which is maintained by the peers that use a consensus protocol in the network to achieve a distributed agreement about the ledger's state [191].

Definition 2.2. **Ethereum.** Ethereum is a popular decentralized blockchain that has the functionality of a smart contract [26]. It was co-founded by V. Buterin, G. Wood, C. Hoskinson, A.D. Iorio, and J. Lubin in 2013. The consensus mechanism used in the Ethereum blockchain used to be proof-of-work (PoW) but recently changed to proof-of-stake (PoS) in 2022 [92].

Definition 2.3. **Smart contract.** A smart contract is a piece of code or programs that reside in the Ethereum blockchain, which is self-executed when the predetermined conditions are triggered [26]. The concept of a smart contract was first introduced by Nick Szabo in the early 1990s [159]. A smart contract is itself an Ethereum account and also has a balance

[1]Portions of this chapter are reproduced from K. Upadhyay, R. Dantu, Y. He, A. Salau and S. Badruddoja, "Paradigm Shift from Paper Contracts to Smart Contracts," 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), Atlanta, GA, USA, 2021, pp. 261-268, doi: 10.1109/TPSISA52974.2021.00029. © 2021 IEEE. Reprinted, with permission.

that can send transactions over the blockchain network when executed.

Definition 2.4. **Ether.** Ether (ETH) is a native cryptocurrency or token for the Ethereum blockchain that can be used as a method of payment for purchase and sales, investment, collateral, etc., in the Ethereum blockchain [26]. As of 2023, Ether is the second-largest cryptocurrency by market capitalization, after Bitcoin [50].

Definition 2.5. **Transaction.** A transaction in the Ethereum blockchain is an action initiated by an externally-owned account (EOA) that is cryptographically signed. It is a process of transferring Ethereum-based assets from one address to another [26].

Definition 2.6. **Gas cost.** Gas cost is the cost required to execute a transaction on the Ethereum blockchain. Gas cost is paid in Ethereum's native currency, Ether [26].

Definition 2.7. **Deployment cost.** Deployment cost is the cost required to deploy the smart contract onto the Ethereum blockchain. It is paid in Ethereum's native currency, Ether [26].

Definition 2.8. **Fuzzy logic.** Fuzzy logic is a sub-topic of the explicit artificial intelligence method, and it is a mathematical approach that was first introduced by Lotfi Zadeh in 1965 to represent vagueness, uncertainty, and imprecise information [186]. Unlike Boolean logic, it is a logic that allows a variable to have truth values 0 and 1 instead of just being true or false.

Definition 2.9. **Linguistic variable.** A linguistic variable or a linguistic descriptor is a variable whose values are not in numbers but are in natural language words and sentences [186]. For example, the linguistic descriptors for the word "reasonable" can be *extremely* reasonable or *rather* reasonable or *not at all* reasonable.

Definition 2.10. **Membership function.** Membership function is a type of function in fuzzy logic that describes and provides the information of fuzziness of an element [186]. Membership function represents "degrees of truth" of something in a fuzzy set [189]. If the

membership value of an element is 0, then it is not a member of the fuzzy set, whereas if the membership value is 1, then it is completely a member of the fuzzy set.

2.2. Brief Overview of Blockchain Technology

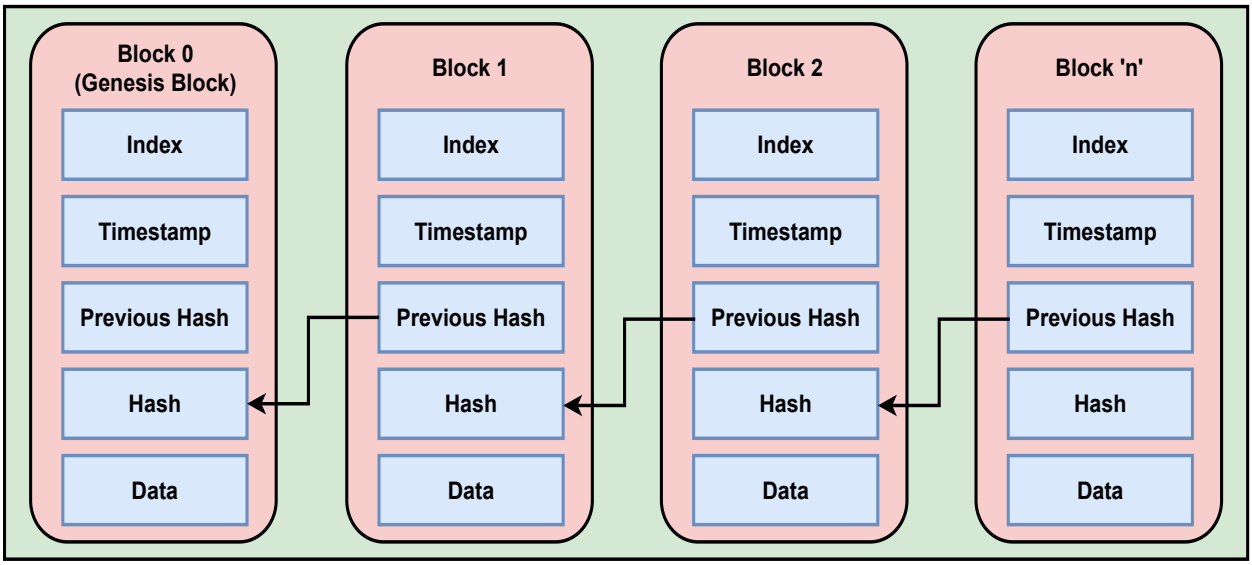

FIGURE 2.1. An illustration of a high-level Blockchain system showcasing the sequence of a chain of blocks [192].

Blockchain is a digital ledger of transactions that is shared and distributed to all the peers that are responsible for managing and confirming the lawfulness of each subsequent batch of transactions, called blocks. These subsequent blocks in the blockchain are ordered chronologically as each new block is added to the previous one. Blockchain is known to be immutable and tamper-resistant because of how the next block is linked with the previous block and also due to its distributive nature.

As this digital ledger is distributed to all the peers or nodes in the network, even if a malicious hacker who has control over one node wanted to tamper with the data in the block, all other nodes would still have the previous data that would not agree with the newly changed data.

Blockchain also has other core features such as decentralized, instant settlement, consensus-based, transparent, and no single point of failure.

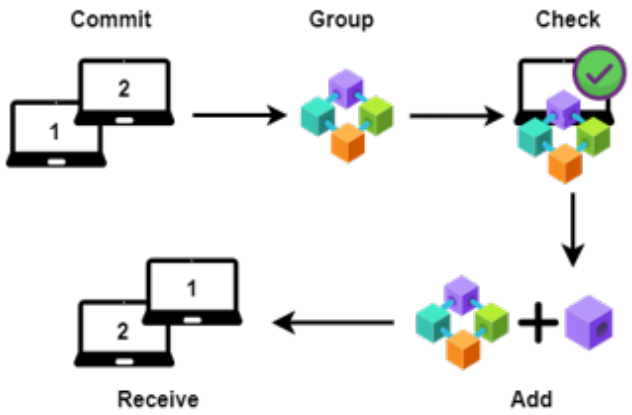

FIGURE 2.2. An illustration of a high-level mechanism of smart contract with respect to the blockchain [159].

Below here, we will briefly explain the step-wise mechanism of the blockchain system:

(1) Creating the transaction: When a sender creates and signs the transaction with their cryptographic keys, that transaction will be relayed to the recipient's wallet, which will need verifying peers or nodes for their verification and approval. If the verifying peers do not see anything suspicious, then the receiving party will see an update in their wallet. This change in the state will be recorded in the subsequent block instantly.

(2) Logging and compiling transaction into the block: The block is encapsulated with the timestamp, date, sender's and receiver's address, the transaction amount, and an encrypted hash of the sender's digital signature. All this information is recorded inside the block while a transaction is logged and compiled into the block.

(3) Distributing the block to peers: After the block is ready, it is distributed to all the peers or nodes throughout the network for verification purposes. As the blockchain is known to be decentralized, these peers inside the network follow a particular consensus mechanism (for example, proof-of-work, proof-of-stake, etc.) on the state of the ledger for verification purposes.

(4) Verifying the block: The peers or popularly known as "miners" in the case of Bitcoin

blockchain [123], depending on the method of a consensus protocol, are required to verify the current block. Based on the amount of work done in proof-of-work or on the amount of stake they own in proof-of-stake, the winner node or peer in the network verifies the block and, in return rewarded with the fees in the form of the native cryptocurrency.

(5) Linking the block: Once the verification of the block is completed and the winning peer are rewarded for their verification work based on their work or their ownership of stake, the block is completed and receives a new timestamp including its unique hash and is ready to be linked with the sequence or chain of previous blocks. The most important feature from a security perspective comes at this moment when the block records the hash of the last block in the chain to create the immutable and tamper-resistant state and proper sequential arrangement of the blockchain. Finally, this update is shared with other nodes or peers on the network. This correctness of the ledger is verified and ensured by the identical hashes.

2.3. Evolution of Legal Contracts

2.3.1. Traditional Paper Contract

Traditional paper contracts are the most common type of legal contract we see in our everyday lives [108]. The agreements between the parties, name of the parties, date, clauses/section, and the signatures of the parties are written on a paper that also includes a lawful governing seal, usually from a rubber stamp. The whole content of the contract is written in natural language by a person, usually by a lawyer, according to what the parties agree for that states their terms and conditions. In traditional paper contracts, the involved parties and the middlemen, usually lawyers and attorneys, need to meet in person to inform them about the terms and conditions of the agreement. When the parties have to make some changes to their existing contract, they meet again with their middlemen and create a new draft of the contract. Once all involved parties agree on the new draft, they sign the contract [108]. In this type of contract, the cost of the attorneys is usually very costly. Other expenses such as paper materials, printing, rubber stamps, several copies of the contract for

each party, and travel costs to meet the parties are also involved, which ultimately increases the final price in the agreement's implementation.

In case when the agreements set out in the contract are not met and the contract is violated, the involved parties have to go to the central authority, i.e., the court system. Here, the legal judge in the court system acts as the arbiter who settles the dispute between the parties.

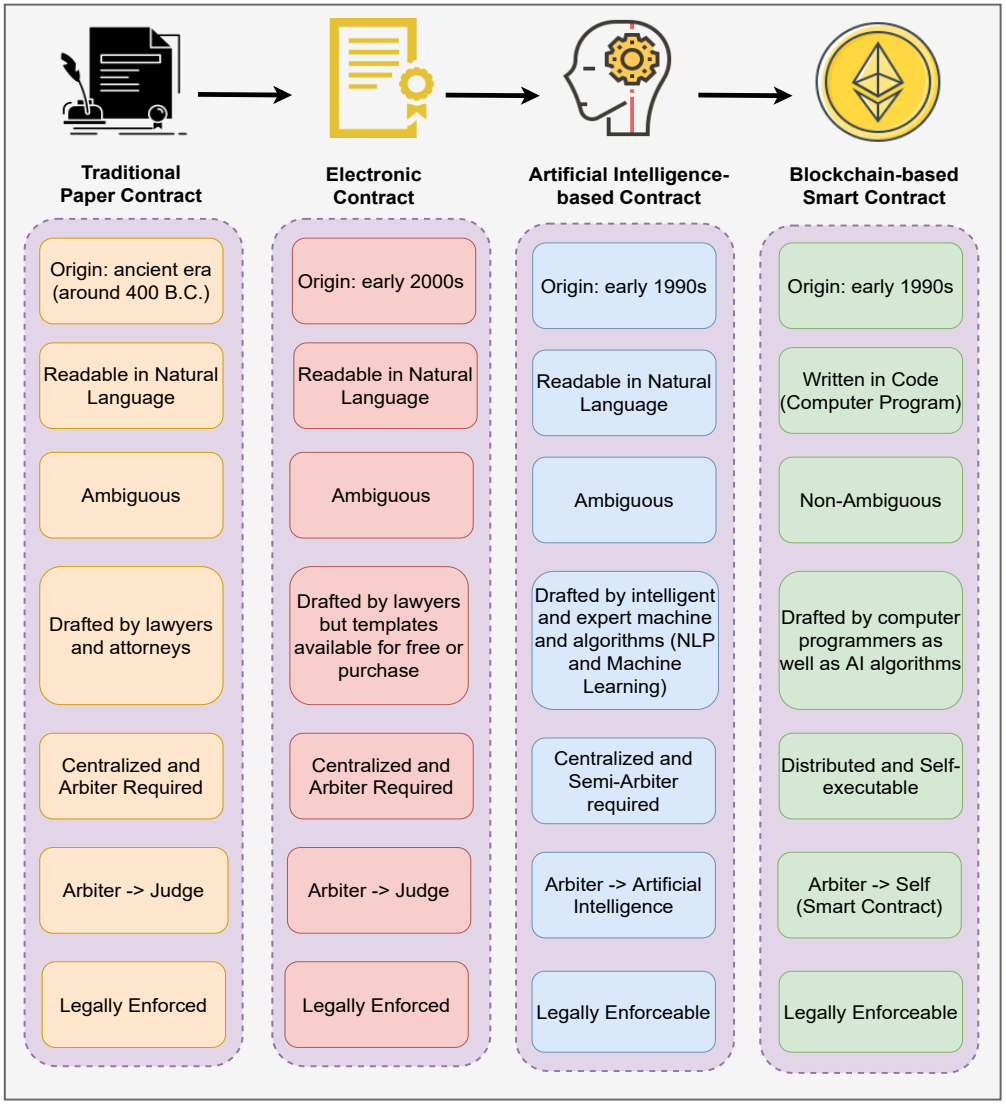

FIGURE 2.3. Paradigm shift of the contract from one stage to the other, where the traditional paper contract is the most primitive kind and blockchain-based smart contract is the most advanced and self-executable, hence, one of the disruptive technologies.

2.3.2. Electronic Contract

People started realizing that traditional paper contracts are a lot more expensive and consume more time when drafting. Even for a slight change in the process of drafting the contract, all the involved parties had to meet in person and their hired attorney for the signatures. After the arrival of the internet and personal computers for regular households were popular, in the early 2000s, electronic contracts were created [48]. Traditional contracts and electronic contracts were basically the same. Still, the major difference between them is that depending on the different needs and uses of the parties, ready-made templates of the various kinds of contracts are already available on the internet. The parties getting into an agreement just need to choose a template from the available templates, fill in their details, and attach their digital signatures [99]. Although the parties just have to download or buy the existing templates, which saves a lot of time compared to traditional paper contracts, the parties can also modify the template according to their needs if they have to. Despite being an electronic contract, this kind of contract is still written in natural language. However, as the whole process has more automation than paper contracts, the probability of human error is lesser. The advantage of an electronic contract is low transaction costs and other miscellaneous costs such as paper and printing.

Despite being an electronic contract, only the drafting and signing of the contract is automated, but settling the dispute still remains as primitive as a traditional legal contract where the involved parties must visit the judicial system in case of dispute settlement.

2.3.3. Artificial Intelligence-Based Contract

Artificial intelligence (AI) has been gaining popularity unlike any other science and has touched almost every sector since its arrival, including law. It is a type of technology that can mimic and replace human behavior. Law firms have been using the help of AI for contract drafting and management. Softwares like Document Assembly Programs [96] have existed since the 1990s. This software allows the lawyers to fill in a pre-coded questionnaire that includes or excludes specific language based on their responses. Once the lawyer answers all questions, the software will generate a final document. While these programs save

a lot of time, on the other hand, these programs do not allow lawyers to modify the coding to adapt to the specific needs of their clients. There has been an increasing demand for AI contract drafting in the field of law these recent years [111], [126], [100]. The contract drafting software based on AI learns from the past and similar contracts. It scans previous documents, identifies essential terms and phrases that include abundant legal jargon, and drafts a suitable legal contract template in just a few seconds. The most significant advantage of AI-driven contract software is that it learns the whole legal contract document by analyzing its subjects, word patterns, writing style, IF/ELSE agreements, different sections, and clauses, etc., and creates a similar legal contract accurately without the intervention of any third party such as lawyers and attorneys.

Moreover, using AI with the law does not only remove lawyers as contract drafting middlemen but also removes the judge from the court of law itself as the demand for prediction of trial outcomes through data analytics as AI intelligence has been able to predict outcomes with increasing accuracy. Scientists and researchers have been using Natural language processing (NLP), Machine learning, and Deep learning to enhance expertise and intelligence by creating AI algorithms to replace the third and centralized entity, i.e., the legal judge from the judicial system, for the dispute settlement processes. AI algorithms have been used extensively to find the settlement area between the involved parties, reducing the need for human contact and increasing the dispute settlement process. The usage of these AI algorithms allows the parties to save their time by settling directly, and these AI-based settlements are far more consistent and uniform as it does not leave any room for errors and human biases [69], [12]. However, two more features are missing even when AI is used for legal contracts, i.e., distributed nature and self-execution of the contract.

2.3.4. Blockchain-Based Smart Contract

So far, we have discussed how a simple and conventional paper contract originated and how it was drafted and functions. Later, the conventional contract was converted to an electronic or digital contract. Despite saving a lot of time and resources, electronic contracts still required a middleman to settle the disputes, if there were any. As the technology became

more advanced, with the rise of AI and its branches like NLP, machine learning and deep learning were able to learn from the past and were capable of both drafting new contracts as well as settling disputes as arbiters without any human intervention, increasing the efficiency. Still, there was one important part missing all along with these transformations, i.e., automation and self-execution of the contract.

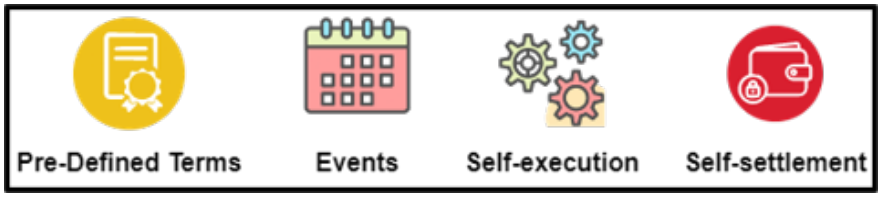

FIGURE 2.4. A blockchain-based smart contract is a computer program that has pre-defined terms with different events and is capable of self-execution, and self-settlement [159].

As mentioned earlier, blockchain is a distributed ledger system that is decentralized, immutable, and cryptographically secured. A smart contract is a concept in the blockchain, which is a computer code that resides inside the blockchain that has all the IF/ELSE statements and agreements between the involved parties of the contract. Due to the features and characteristics of this blockchain-based smart contract, the contract is distributed, decentralized, and secured. As a result, this makes the smart contract self-executable which does not require any outsiders or third parties or any arbiters in case of disputes.

Unlike traditional paper and electronic contracts, a smart contract is the most novel and technologically advanced contract, which is a computer program intended to execute and enforce automatically. The concept of a smart contract was first introduced by Nick Szabo in the early 1990s [159]. The smart contract runs on the Ethereum blockchain. This contract is a collection of code, i.e., functions and data, i.e., state that resides at a specific address on the Ethereum blockchain [26]. A smart contract itself is a type of Ethereum account. It has a balance, and it can send transactions over the blockchain network when triggered. Hence, it reduces the need for trusted intermediates. As it is self-executed and

self-enforced, a smart contract is not controlled by the involved parties. They are instead deployed to the blockchain network and run as they are programmed. Involved parties or user accounts can then interact with the deployed smart contract by sending transactions that execute a specific function defined inside the smart contract.

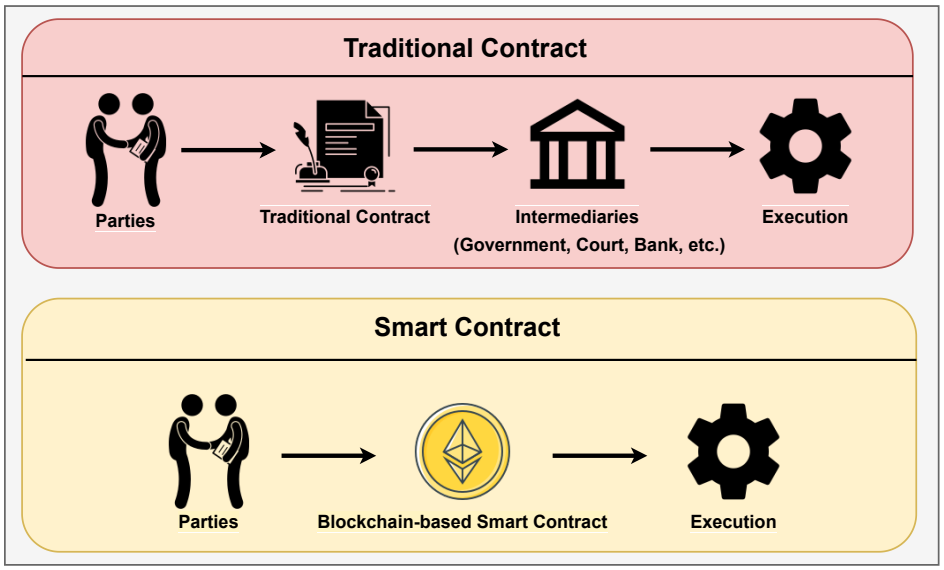

FIGURE 2.5. Traditional contract Vs. Smart contract, where the traditional contract needs to involve at least one intermediate party for execution, but the smart contract is self-executable without the need for an intermediate.

A smart contract is designed in such as way that if one of the two parties violates the contract, then with its self-executable feature, the contract gets triggered, and the violating party is penalized automatically. For instance, there are two parties A and B inside the smart contract that is programmed for a rental agreement. Assuming, A being the landlord and B being the tenant, if B is unwilling or fails to pay his/her rent by the due date, A does not have to seek the arbiter or a third-party AI-based expert system in this case. Since smart contracts are self-executing by nature, once the tenant violates the contract by not paying the rent at the proper time, the money from the tenant's account can be automatically transferred to the landlord's account.

It would be impossible for anyone meddling with the contract to modify anything in-

side the contract as it is cryptographically secured and immutable. Not only is it immutable, but all the nodes or participants will also know about the transactions and logs inside the blockchain as it is distributed throughout the network. However, it does not mean that everyone in the blockchain network would know that the tenant could not pay rent, and he/she has to suffer from embarrassment. Since everything inside the blockchain is encrypted, the involved parties (accounts) will remain anonymous. In addition, blockchain-based smart contracts completely eliminate the trust factor because it follows a peer-to-peer network architecture [84], [119]. Hence, the involved parties do not have to be concerned with the central (third-party) figure, such as a human judge in courts, as it usually introduces biases.

In Table 2.1, we present several kinds of smart contracts written in their respective programming languages in different blockchain platforms.

TABLE 2.1. Smart contracts in other popular blockchain platforms

Blockchain Platform	**Smart Contract**	**Programming Language**
Ethereum	Ethereum Virtual Machine (EVM) [78]	Solidity
Bitcoin	Bitcoin Script/Bitcoin Contracts [23]	C++, Custom Stack-based Language
Binance Smart Chain	Binance Smart Contract [32]	Solidity (Compatible with EVM)
Polkadot	Substrate Smart Contracts [89]	Rust
Cardano	Plutus [102]	Haskell
NEO	NEO Smart Contract [58]	C#, Java, Python

2.4. Adoption and Legal Enforcement of a Smart Contract

Smart contracts have been popular within a short period. With the advent of blockchain and smart contracts, only a few people were using smart contracts. However, in these recent years, the smart contract has been increasing its scope around as many areas

as possible. Trading activities, mortgages and loan systems, record storage, insurance, supply chain management, and crowdfunding are famous use-case examples of smart contracts. Although the smart contract is still not as mature as traditional paper contracts, people and even government of many nations have started to realize that smart contracts have been offering solutions to the existing legal and security challenges that have an abundance of loopholes [72].

For instance, in 2017, a provincial government in Switzerland launched the issuance of digital IDs that runs on the Ethereum blockchain[2]. Similarly, the government of Chile started using the Ethereum blockchain to track the data to maintain accountability and integrity in the energy department. The government of Georgia converted its traditional land titles registry to blockchain in 2017. In 2021, El Salvador became the first nation to recognize Bitcoin as a legal tender and established a law recognizing Bitcoin as a legitimate payment[3]. In the same way, countries such as Estonia, UK, UAE, Brazil, Sweden, Singapore, Portugal, Malta, Nigeria, etc., are also adopting blockchain and smart contract technology [61].

Just like in a traditional paper legal contract or an electronic/digital, a smart contract also has the same elements and features. These elements are mutual agreement, consideration, competent parties, genuine consent, and, finally, legally enforceable. Similarly, in a smart contract, there will be one party that offers and another party that accepts in exchange for a benefit. The involved parties in the smart contract have to reach an agreement. As mentioned earlier, in exchange for a benefit from the other party, each party gives up something of value. Also, it is significant for the parties in the smart contract to be competent, as the smart contract can only be enforced when the involved parties are qualified. The smart contract also has the feature of genuine assent, as all involved parties in the smart contract must engage in the agreement independently. Finally, the smart contract has a lawful and legal purpose, although it is written in the code and not in the natural language

[2]https://irishtechnews.ie/global-blockchain-adoption-which-countries-are-leading-the-charge/

[3]https://cointelegraph.com/news/5-countries-leading-the-blockchain-adoption

like traditional paper contracts. Hence, the smart contract has exactly the same features as the paper contract. In addition, it has more technological features that are even more advantageous to us compared to paper contracts. Some of these advantages are that smart contracts are decentralized, distributed, and immutable, and settlement occurs faster.

Whether a smart contract is legally enforceable or not depends on whether the smart contract meets the requirement of a valid legal contract. This will also depend on what law applies and the jurisdiction in which the enforcement is called [72]. Since the smart contract is a relatively new and emerging technology, it may not have been evaluated and tested by regional or national law. Nevertheless, smart contracts' enforceability should not be ruled out simply because they are written entirely in computer code for automation and self-execution. As long as a smart contract behaves like a traditional paper legal contract and complies with national/provincial/state law, the current legal system should not have any issue in adopting the smart contract.

The following are the significant requirements, also named by us as Five A's, that a smart contract needs to fulfill to be adopted by the current legal system for legal enforceability [165].

2.4.1. Admissibility

The term admissible means that something can be accepted. For a smart contract to be admissible in a court of law, it must prove that it is valid in the proceeding or comply with the law. A smart contract should be admissible in a court of law just like a traditional paper contract, as the smart contract also behaves the same way the paper contract does. Additionally, the smart contract also has all the major components that make it a legally enforceable contract. Hence, for this reason, a smart contract describes the information and has the characteristics that are pertinent to a resolution of issues in any kind of judicial proceedings so that a judge or jury can consider such information and characteristics to make a decision.

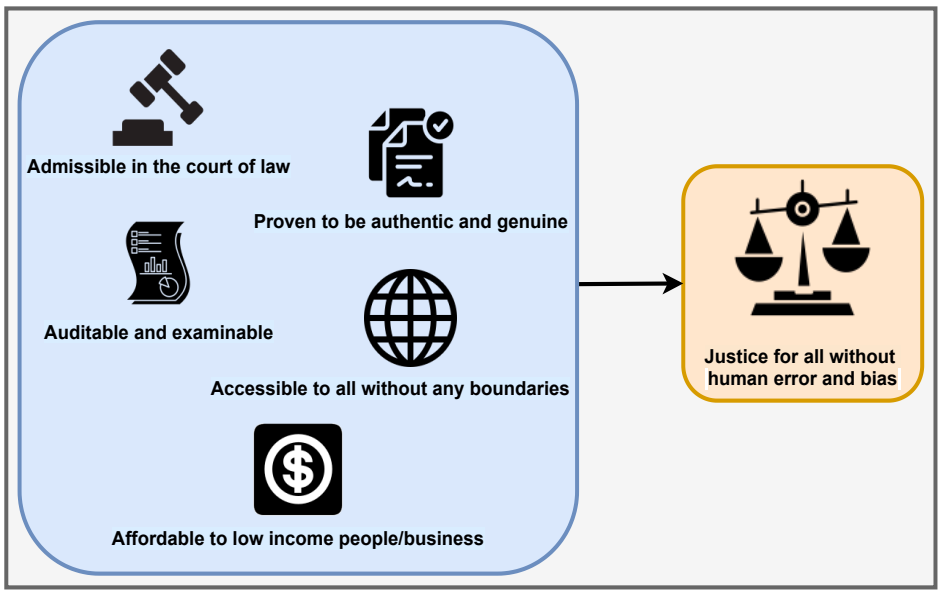

FIGURE 2.6. Major requirements, also known as 5 A's for Smart contract's enforceability. These 5 A's prove why a smart contract provides justice for all without any human errors and biases and is perfectly legally enforceable [165].

2.4.2. Authenticity

Many documents must fulfill the criteria of authenticity in a court of law for them to be legally enforceable. Authenticity defines the process by which the information and characteristics of a smart contract are proven to be veritable and legitimate. For anything to be legitimate and legally enforceable, it should be genuine and not a forgery. One of the major traits of the smart contract is immutability which is the biggest proponent of authenticity because once the smart contract is written and deployed onto the blockchain, it can neither be modified nor changed to maintain its integrity and legitimacy.

2.4.3. Auditability

The auditability of the smart contract or any kind of document is a core part of legal compliance from the judicial perspective. This feature of auditability and the audit logs enables the court to examine and verify when the smart contract was created, deployed into the blockchain, signed, and used to make a transaction. In addition, the smart contract and the transactions made via the smart contract give involved parties detailed and tamper-proof

timestamped audit logs of every event.

2.4.4. Accessibility

The contract law should not, in any case, be out of reach of the parties, and there should be no sense of discrimination. The terms and agreements in the legal contract should be fair and unbiased all the time. For example, when multinational companies are legally bounded by traditional paper or electronic contracts, the companies' corresponding government will have limited power over companies from different governments. However, if these multinational companies had been using smart contracts, there would be no discrimination and differentiation between any country and their legislation and judicial practice. Hence, smart contracts ensure their reach to everyone equally.

2.4.5. Affordability

Hiring a contract lawyer to review our written agreements in paper contracts in the contracting process is one of the principal steps, as the words and formats used in the paper contract need to be very specific and must follow a certain pattern to be legally binding. Although hiring and working with a contract attorney will probably ensure that the agreements are admissible in court and are legal, it is still very pricey. Depending on the situation, when a lawyer or an attorney is hired just for the review of the paper contract, the price can be unreasonably high, ranging from at least $500 to $1000 [73]. To make it worse, if people actually hire an attorney to draft and negotiate the contract for them, the price can be even exorbitant.

On the other hand, the development and deployment cost of the smart contract is expensive as well [66]. Moreover, proper auditing and testing of the smart contract are costly as it requires people with special expertise and background. In addition, when deployed to the main net and when making transactions with the smart contract, the incurred gas fee and transaction fee can be excessively high too. Currently, it may seem that the smart contract is not fit for small and mid-sized businesses because of the development and deployment cost of the smart contract. Although the development cost and deployment cost of the

smart contract is also expensive currently, it is inevitably true that in the future, the cost of adopting the smart contract will plummet as these days, the cost of cryptocurrencies are just exaggeratingly high due to immature market of cryptocurrencies and other factors.

2.5. Active Research Topics in Smart Contracts and Emerging Technologies in Legal Aspects

In spite of smart contracts being considered as one of the emerging technologies, there are other technologies as well where researchers are actively collaborating with each other to experiment and amalgamate smart contracts with other disciplines [16], [17], [155], [148], [122]. The smart contract has been providing immense benefits, but there are still various challenges on how to derive the smart contract from the legal contract as typically, regular paper contracts are written in natural language and hence create a high risk of vagueness, whereas a smart contract is a piece of code or a computer program. Hence, the derivation of a complete smart contract from a vague legal contract is still one of the major challenges.

There are various groundbreaking pieces of research being conducted in the field of Artificial intelligence (AI) and law using formal models of legal texts and legal reasoning as well [16], [12], [84]. One of the major roles of formal models is to remove vagueness, as regular legal contracts are written in natural language. As a result, there are no parenthesis or brackets; hence, the scope of connectives such as "and" and "or" can be vague. There are other words and phrases that are used in legal contracts as well, which are vague. For example, words such as "unless", "reasonable", "may", "can", etc., are capable of several interpretations [167], [74], [166]. Therefore, a lot of novel legal research includes the usage of propositional logic, fuzzy logic, and AI that attempts to understand, interpret and resolve the vagueness and fuzziness of legal contracts [166].

2.5.1. Natural Language Processing

Natural language processing (NLP) manipulates the natural language, such as text or speech, by a computer program. NLP is a subsection of Artificial intelligence (AI) that does not only help computers to understand human language but also to interpret it. NLP has roots in disciplines such as computer science and computational linguistics. Recently,

computational law research using NLP has been a hot topic as computational law involves analyzing natural language-based data and documents such as legal contracts in a considerable quantity. Therefore, modern machines and programs can analyze more language-based data and documents than humans consistently without any fatigue and bias.

There has been a massive increase in the demand for software development for the automation of tasks due to the growth of legislation. Presently, an analyst or an attorney is required who is expensive to hire to draft and interpret the law in legal activities. Nevertheless, there is always an issue of fuzziness and vagueness in the legal documents and contracts as they are written in plain natural language, which creates multiple interpretations for multiple parties involved in the legal action. For instance, the word "book" has numerous meanings. One is the verb that means to reserve, and the other is the noun that means something to read from. Attorneys overlook these issues intentionally or unintentionally when they draft and analyze the contracts as they review and analyze thousands of legal contracts full of vague words and legal jargon. Instead, researchers are using NLP so that they can pinpoint the specific vague terms and provide correct revisions for improvement [122]. Furthermore, NLP experts are trying to create a computational model to generate smart codes from the analysis of legal contracts, using NLP and Blockchain-based smart contracts so that they don't leave room for vagueness.

2.5.2. Machine Learning and Deep Learning

As NLP technologies have been involved more in attempting to review, analyze, interpret, and generate the logic for the smart contract's development, more research is going on on the security side of the smart contracts where machine learning has been used [12], [17], [148], [122]. Just like NLP, machine learning is also considered to be a subset of AI. Machine learning is defined as a branch of AI and computer science that focuses on using the available data and algorithms to imitate the way humans learn by improving their learning accuracy eventually. On the other hand, deep learning is a subset of machine learning and AI that is a neural network with three or more layers where these neural networks simulate the human brain's behavior. Deep learning algorithms are more modern and accurate for

learning something than machine learning algorithms but require more data to learn.

In recent years, hackers and malicious attackers have not only been exploiting vulnerabilities in web-based systems but also in blockchain-based smart contracts, which has resulted in huge economic and financial losses. For that reason, to find out and detect these vulnerabilities of the smart contracts, the researchers have used an analysis model that uses machine learning extensively [17]. These studies have successfully shown that their analysis model can predict various types of vulnerabilities, particularly in smart contracts of Ethereum blockchain written in Solidity language, such as access control, arithmetic, denial of service, re-entrancy, etc., with accuracy, precision, and recall with more than 90% [16].

Machine learning has not only been used just for prediction and detection of smart contract vulnerabilities but also has been used for legal contracts management [182]. Machine learning helps in identifying and analyzing the clauses and other relevant data in the contracts. Besides, it also has been allowing business companies to review thousands of contracts quickly by classifying the contract according to its relevancy, classifying the clause, pinpointing a significant part of the clause, and learning more about new clauses.

Nevertheless, machine learning and deep learning typically take a lot of computer processing power and memory. On the other hand, blockchain costs a lot for any processing, storing, or computer processing power as well. Since anything inside the blockchain costs money and is usually expensive, the cost factor of using machine learning and deep learning inside the smart contract or the blockchain still remains a significant challenge.

2.5.3. Internet of Things (IoT)

The Internet of Things, also known as IoT, is a large number of devices connected to the internet to share data with each other. These internet-connected devices use sensors to gather data and communicate with each other so that humans can improve their living and working lifestyles. One popular example of IoT is a smart home that automatically adjusts heating and lighting to a smart factory that monitors industrial machines to look for problems and then automatically adjusts to avoid failures.

IoT establishes an excellent combination with blockchain-based smart contracts, par-

ticularly when it comes to business, financial, and legal transactions, as they are traditionally authorized by a third party, such as a bank or a court, making the transaction process complex and time-consuming. When smart contracts are used with IoT, it will solve a plethora of problems, such as the publishing of secure software updates as URLs on the blockchain that includes cryptographic hash that IoT devices can validate and allowing automatic payments to everyone on the IoT network and ensuring of micropayments made between the IoT devices as well, and sending of accurate information on food temperature for frozen items to the blockchain network by the IoT sensors so that the data can be analyzed among stakeholders to ensure the quality and freshness of the frozen foods. IoT also helps in security vulnerabilities by allowing data sharing more securely across stakeholders, automating transactions, verifying identification and authentication, and reducing costs by disintermediating mediators when merged with blockchain-based smart contracts [17], [182]. For instance, the status of the IoT network will be improved by allowing devices to register and validate themselves, self-executing contracts, and reducing the threat of cyber attack since there would be no central system to attack [17]. Therefore, when combined with IoT, blockchain-based smart contracts will benefit us immensely.

2.6. Related Literature Review

Although there has been profound research going on for smart contracts in a substantial manner in recent years, the study on "smart legal contracts" has not been so thorough. Despite the fact there has been extensive research on vagueness, legal contracts, and smart contracts separately, there has not been any study on the relationship between legal contracts and smart contracts. Smart contract and vagueness has been studied in [74], but the author does not have any methodology to classify the interpretations of legal contracts and smart contracts based on the vagueness level. There is only a superficial classification of vagueness from a linguist's perspective made in [3] by the author, which was not enough as our study covered more aspects than just a linguistic point of view. In [158], the author explains how a contract can be computed and how it can be converted into code but lacks the research and discussion of vagueness and concepts of a smart contract. In [64], the author

talks about the rules by which various sequences of the events trigger particular sequences of state transitions in the relationship between the entities in which vagueness has not been discussed. In [155], the author talks about blockchain being used for drafting and probating wills and making the contract transparent and secure, yet we cannot find an explanation of the relationship between a legal contract and a smart contract based on the vagueness. In [42], the author takes vagueness into account by encoding contract metadata, but the consideration of actual clauses is completely ruled out.

Similarly, although there have been several types of research on vagueness and types of vagueness separately, there has not been any research so far on various kinds of legal contracts and SLAs and how we can convert these legal contracts and SLAs into smart contracts, considering vagueness as the main challenge. There has been a study done on how an SLA can be converted into a smart contract that can be used in the blockchain to reduce manual effort to claim compensations in [150]; however, the authors have not described the vague and legal jargons that we see in the SLAs and how that vagueness was considered while converting the SLA into the smart contract. In [169], the authors talk about the SLA management system but lack research on how we can convert an SLA into a smart contract. Also, only the basic functions of the SLA Management System have been studied. Likewise, the authors talk about how they proposed a new SLA management framework that uses two-level blockchain architecture and how an SLA is transformed into a smart contract in [168] but fail to include the concept of vague requirements that can cause issues while writing a smart SLA. In [10], the authors have proposed a blockchain-based method to assess SLA compliance but have ruled out the vagueness found in the SLA. Correspondingly, in [125], the authors have proposed a system that uses blockchain, which claims the compensation process can be kept safe and reliable but again lacks the discussion of the vague nature of SLA.

Table 2.2 here provides a concise comparison between the contributions of several existing works and the original research presented in the subsequent chapters of this dissertation, focusing on the key topics that are relevant to the subject matter.

TABLE 2.2. Comparison of few contributions from existing work and this dissertation

Topics Covered (↓) in Lit. Surveys (→)	**[74]**	**[3]**	**[158]**	**[64]**	**[155]**	**[42]**	**[150]**	**[169]**	**[10]**	**[125]**	**This Dissertation**
Legal Contacts	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Vagueness	✓	✓	✓	✓	×	✓	×	×	×	×	✓
Interpretation	×	×	×	×	×	✓	×	×	×	×	✓
Quantification of uncertainty	×	×	×	×	×	×	×	×	×	×	✓
Reasoning with linguistic variables	×	×	×	×	×	×	×	×	×	×	✓
Fuzzy Logic	×	×	×	×	×	×	×	×	✓	✓	✓
Blockchain and Smart contracts	✓	×	×	×	✓	×	✓	✓	×	✓	✓
Security	×	×	×	×	✓	×	×	×	×	×	✓

CHAPTER 3

STUDY OF CROWDFUNDING CONTRACT'S VAGUENESS AND TRANSLATION INTO SMART CONTRACT[1]

3.1. Introduction

A contract is an agreement that is in written or spoken form. It settles an agreement or a dispute between one or more parties since it is intended to be enforceable by law. It can be classified into different types [43]. However, in this chapter, we will only be focusing on traditional legal contracts, which are usually in paper form, or in some cases, in electronic form. Fundamentally, this type of contract contains do's and don'ts under different clauses. As we now all know that a contract involves one or more parties, so, consequently, a team of lawyers is also involved since a contract is involved in legal cases. These lawyers can be considered as *middlemen* since they are the ones who try to arrange and decide the best possible situation for all the parties who are involved in the legal contract.

On the other hand, a Smart contract (SC) is a computer program that is self-executable, self-enforced, and managed by a blockchain [26]. The computer program comprises the explicit and precise set of rules under which the parties of that smart contract agree to interact with each other. If and when the predefined rules written in the smart contract are met, the agreement is self-executed and enforced.

Thus, the main problem definition of this chapter is how we can convert or translate a "dumb" legal contract that is full of vagueness into an accurate *smart* legal contract that can be applied and used in the Ethereum blockchain. To perform the experiment and to measure the accuracy of a derived smart contract from a traditional legal contract, we have specifically taken a general crowdfunding legal contract because of its application in blockchain and the long set of vague terms and conditions. In this chapter, we discuss the

[1]This chapter is presented in its entirety from K. Upadhyay, R. Dantu, Z. Zaccagni and S. Badruddoja, "Is Your Legal Contract Ambiguous? Convert to a Smart Legal Contract," 2020 IEEE International Conference on Blockchain (Blockchain), Rhodes, Greece, 2020, pp. 273-280, doi: 10.1109/Blockchain50366.2020.00041.

steps that were taken in order to translate a legal contract into a smart contract considering all the ambiguities and vague terms present in a legal contract. This chapter also discusses the different interpretations of a legal contract that people can have based on their knowledge and experience and how those multiple interpretations can have an effect on the accuracy of the translated smart contract.

3.2. Contributions

The main contributions of this chapter are as follows:

- We investigate the legal contract's vagueness by generating all possible interpretations a contract has and convert into separate control flow graphs.
- We translate the generated control flow graphs of all interpretations into separate smart contracts for each interpretation for the Ethereum-based blockchain.
- We find the vagueness of each translated smart contract based on their performance.
- We use McCabe's cyclomatic complexity [116] to generate the vagueness index based on the complexity of the control flow graph of each interpretation.
- Finally, we identify the vaguest and most accurate translated smart contract based on its performance and vagueness index.

3.3. Relationship between a Traditional Legal Contract and a Smart Contract

Since a legal contract consists of a plethora of vague and legal words, it results in various different interpretations. For instance, a person who is reading a legal contract might perceive it in a different way than the other person who is reading the same legal contract. The main reason for the multiple interpretations of the people reading the same legal contract comes from the vagueness of the words used in it and how the meanings of those words can be perceived [3]. Fig. 3.1 shows that several versions of smart contracts can be mapped or converted from a legal contract as legal (natural) language can result in different interpretations and understandings for different people. It also explains the relationship between a legal contract and the generated smart contracts from the same legal contract can have one too many relationships.

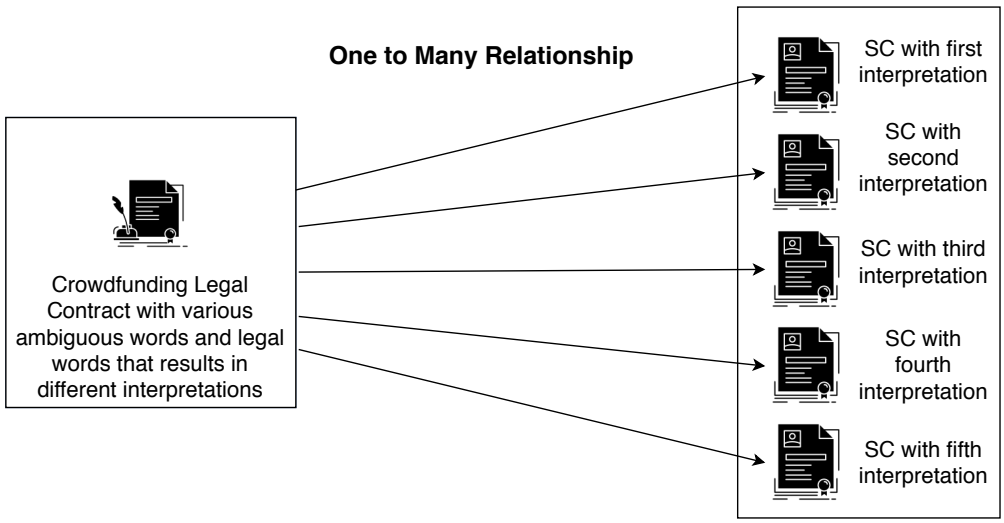

FIGURE 3.1. One to many relationship between a legal contract and smart contract

3.4. Experimental Setup

The tools and materials that we have used for this work are listed below: i) Ropsten Test Network [190], ii) Solidity Programming Language 0.5.3 [47], iii) Remix Web IDE [85], iv) Metamask [107], v) Node.js [161], vi) Truffle [173], vii) Ganache-CLI [106], viii) Web3 [175], ix) HD Wallet [49], x) Google Chrome in Incognito Mode [143], and xi) A Crowdfunding Legal Contract [67] xii) An Employment Agreement Legal Contract [132].

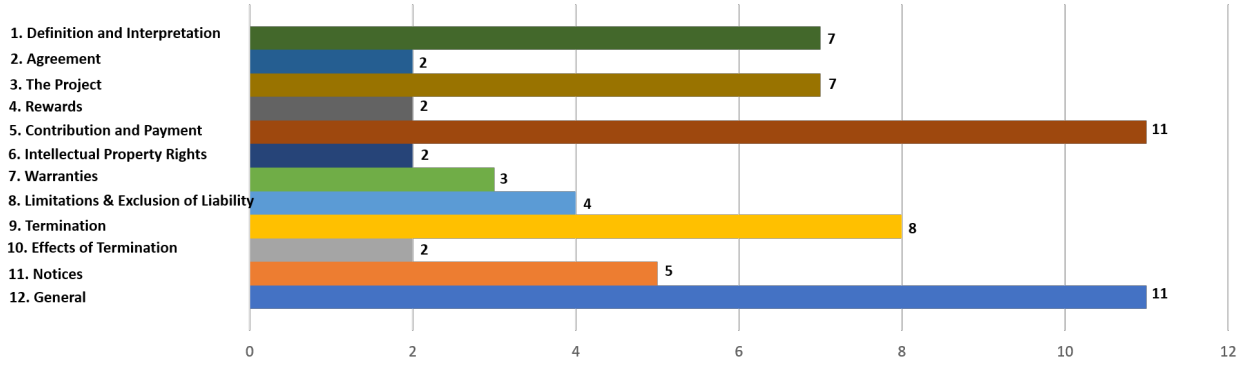

FIGURE 3.2. Number of times vague words and phrases were found in each clause with "Contribution and Payment" and "General" being the highest.

Fig. 3.3 shows there are four different fundamental phases in this chapter. The first phase is the selection of a legal contract. For this project, we have selected a regular crowdfunding legal contract as a test contract. The second phase talks about having different interpretations of the same legal contract. In the third phase of the project, we translate all possible interpretations derived from the vague crowdfunding legal contract into a respective smart contract. In the fourth and final phase, we find out which interpretation of the smart contract is the most vague and accurate.

Oftentimes, we have heard people and companies suing each other because of the lack of understanding of the terms and conditions in the contract. The only reason a legal contract is making everyone's life difficult is because of the way it is written, i.e., with many vague terms and jargon words [3]. As a result, it is obvious for different people to perceive the same contract in different ways. Hence, people who are reading a legal contract might have different interpretations of each other, as shown in Fig. 3.1. Another objective of this study was to create all possible interpretations people might have when reading a legal contract and convert all those interpretations into a smart contract, and finally find out the vaguest and most accurate smart contract among them. This crowdfunding legal contract was taken as a test sample from Cloudset Solutions from Coherence Design [67]. However, since the crowdfunding legal contract is several pages long and has 12 clauses in total, we have only taken one particular clause, i.e., "Contribution and Payment" (Clause number 5), into consideration for testing. The reason behind selecting only this particular clause out of all 12 clauses is that the number of vague words and phrases in this clause was more in numbers compared to other clauses, as we can see in Fig. 3.2 and this clause also constantly revolved around the idea and mechanism of how crowdfunding works and involved more transactions. The other reason to select this particular clause among other clauses is because of the more number of permissive and vague words and phrases used in that clause such as "may", "otherwise", "time to time", "is not intended" and "might". From Fig. 3.2, we can also see that out of 64 vague words and phrases found in the legal contract, this

clause has 11 of them, i.e., 17.18%.

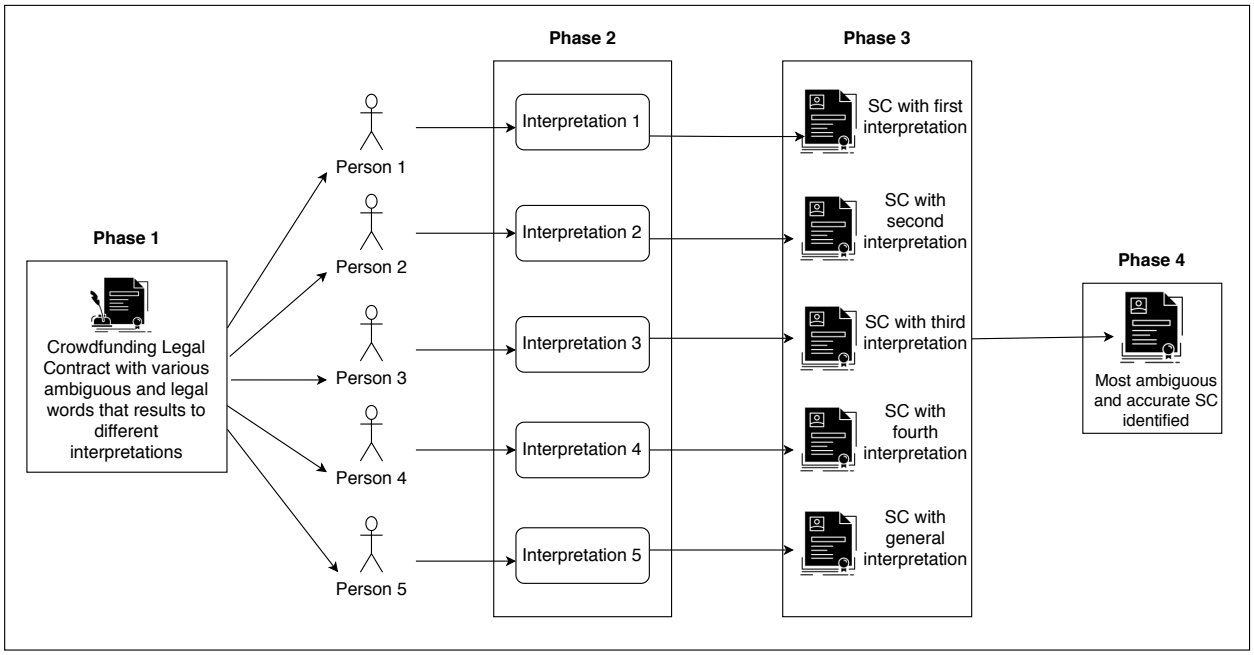

FIGURE 3.3. Selection of a legal contract in the first phase, generation of all possible interpretations of the selected legal contract in the second phase, translation of all possible interpretations derived from the vague legal contract into their respective smart contract, and identification of the vaguest as well as accurate smart contract in the fourth phase.

The control flow graph shown in Fig. 3.4 was generated from the fifth clause called "Contribution and Payment". This clause says that in a crowdfunding platform, a *'developer'* who is seeking monetary aid receives money from *sponsors* once the sponsors like his/her idea. However, this clause also states the rules of payment to the developer. It uses statements as *"All contribution amount are stated exclusive of VAT, unless the context requires otherwise"* and *"If the Sponsor does not pay any amount properly due to the Developer under or in connection with the Agreement, the Developer may charge the Sponsor interest on the overdue amount."* However, the words like *"otherwise"*, *"properly"* and *"may"* do not give clear and specific instructions hence, result in multiple interpretations. The word "may" itself could mean *"yes"* or *"no"*. Fig.3.4 explains the steps of Clause 5 with additional possible steps that

arise from these vague words. Hence, from the control flow graph from Fig. 3.4 and these two vague statements, we have created and categorized four further possible interpretations and shown them in their respective control flow graphs in Fig. 3.5.

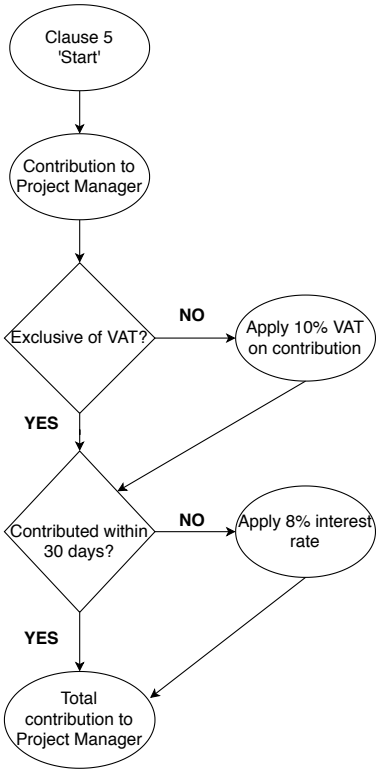

FIGURE 3.4. Control flow graph of the events from a clause "Contribution and Payment" from Crowdfunding Contract (General/Root Interpretation).

As shown in the Fig. 3.5, we can see that Fig. 3.4 's control flow graph can be further categorized into four different interpretations from where we can create four different control flow graphs. This is only possible due to the words such as *"may"* and *"properly"* present in Clause 5 of the crowdfunding legal contract, which has vague and multiple meanings. If mandatory words such as *"must"* or *"will"* were present instead of *"may"* and *"requires otherwise"*, then we would only have one control flow graph and no other variations because of it's preciseness.

3.6. Results

As we have generated a maximum of five different interpretations in total, including Root/General Interpretation, we also have measured the metrics for each interpretation.

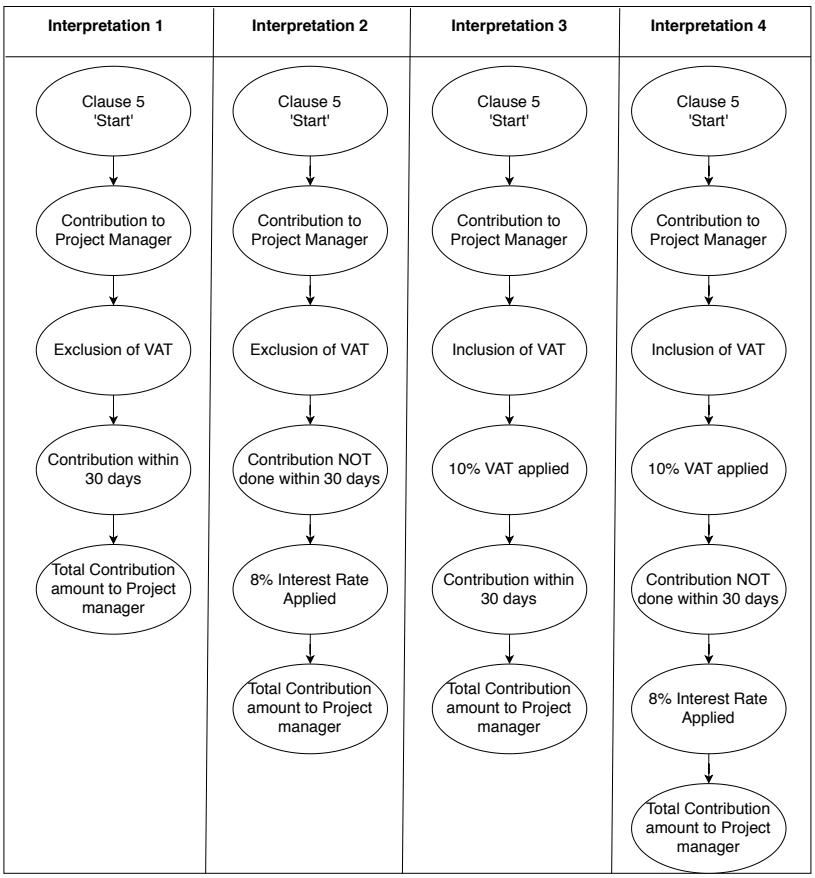

FIGURE 3.5. The variation in control flow graphs showing multiple interpretations from Fig. 3.4's control flow graph.

3.6.1. Performance of Smart Contracts by Each Interpretation

(1) Smart contract with Interpretation 1:

- Transaction fee of 100 transactions of Interpretation 1: 25 deployments of the same smart contract with Interpretation 1 were performed, in addition to 75 transactions in Ropsten Testnet. The deployment cost is constant. It is approximately 0.0007 ethers and is constant until the end. The deployment cost was much higher than the transaction cost. Although there were a few discrepancies in the transactions cost, the ethers that it consumed to run in Ropsten Testnet are very similar. Apart from the deploy function, there is only one function that took more than 0.0001 ethers, i.e., withdraw() function. This function is used when the sponsors contribute a payment to the developer and

when the developer is ready to withdraw the payment.

- Total time taken for each transaction of smart contract with Interpretation 1: The lowest time taken by one of the transactions was 9 seconds. On the other hand, a transaction took 1463 seconds, which is approximately 25 minutes. However, the average time taken by all these 100 transactions was 155 seconds, which is approximately 3 minutes. The time these transactions take depends on various factors. If the function is too complex and has a greater number of parameters, then it takes more time. Also, if the test net gets busy at its peak time, then it takes more time to be registered.

(2) Smart contract with Interpretation 2:

- Transaction fee of 100 transactions of Interpretation 2: The transaction fees and deployment cost of 100 transactions for interpretation 2. The highest transaction fee is 0.0006 ethers. Out of 100 transactions, 25 transactions have the same amount of fees, i.e., 0.0006 ethers. Since these 25 transactions were deployment costs, therefore the fees were much higher compared to other transaction costs.
- Total time taken for each transaction of Smart contract with Interpretation 2: The lowest time for a transaction to register taken was 9 seconds. The highest time for a transaction to register was 1549 seconds. And the average time for all 100 transactions was 96 seconds.

(3) Smart contract with Interpretation 3:

- Transaction fee of 100 transactions of Interpretation 3: This data of transaction fees for the smart contract with Interpretation 3 was exactly as same as for Smart contract Interpretation 2. The highest transaction fee is 0.0006 ethers. Out of 100 transactions, 25 transactions that are the deployment costs have the same amount of fees, i.e., 0.0006 ethers, which is the same as the previous case from Interpretation 2. The only reason behind the costs being the same is that the smart contract complexity for both interpretations 2 and 3 is also similar.

- Total time taken for each transaction of smart contract with Interpretation 3: The time taken to register keeps on varying as the peak rate of Rospten Testnet varies. Whenever the network is too busy, it usually takes more time to register the transactions. The lowest time taken for a transaction to register taken was 5 seconds. The highest time for a transaction to register was 1703 seconds. And the average time for all 100 transactions was 89 seconds.

(4) Smart contract with Interpretation 4:

- Transaction fee of 100 transactions of Interpretation 4: Although this data for transaction fees for the smart contract with Interpretation 4 is very much similar to previous interpretations except Interpretation 1, the highest transaction fees, in this case, is a bit more than previous interpretations 2 and 3.
- Total time taken for each transaction of smart contract with Interpretation 4: The lowest time for a transaction to register taken was 1 second. The highest time for a transaction to register was 390 seconds. And the average time for all 100 transactions was 55 seconds, which is much lesser than Interpretation 3. The reason for the variance in time taken to deploy was the fluctuations in the peak rate of the Ropsten Testnet.

(5) Smart contract with General Interpretation (Root Interpretation):

- Transaction fee of 100 transactions of General Interpretation: This smart contract with General Interpretation is the kind of smart contract where most people perceive the legal contract in a more practical way in the real world. This is the case of how a clause looks like in General Interpretations where there are lots of branches of *"yes"* and *"no"*. 25 transaction fees that are the deployment costs, were the highest compared to all interpretations, the highest transaction fees. The highest transaction fee here is more than 0.0007 ethers. This is the first sign of a smart contract, with this interpretation type being vague compared to the other interpretations. The more vague an interpretation is, the more complex it becomes. As a result, the more complex an interpre-

tation is, the more costly it is in terms of fees. This means that vagueness, complexity, and cost have a direct relationship.

- Total time taken for each transaction of smart contract with General Interpretation: The lowest time for a transaction to register taken was 1 second. The highest time for a transaction to register was 671 seconds. And the average time for all 100 transactions was 80 seconds. The reason for this variance in time taken to deploy is the same as in previous cases, i.e., the fluctuations in the peak rate of the Ropsten testnet.

3.6.2. Comparison of Average Transaction Fees between Different Interpretations of Smart Contract

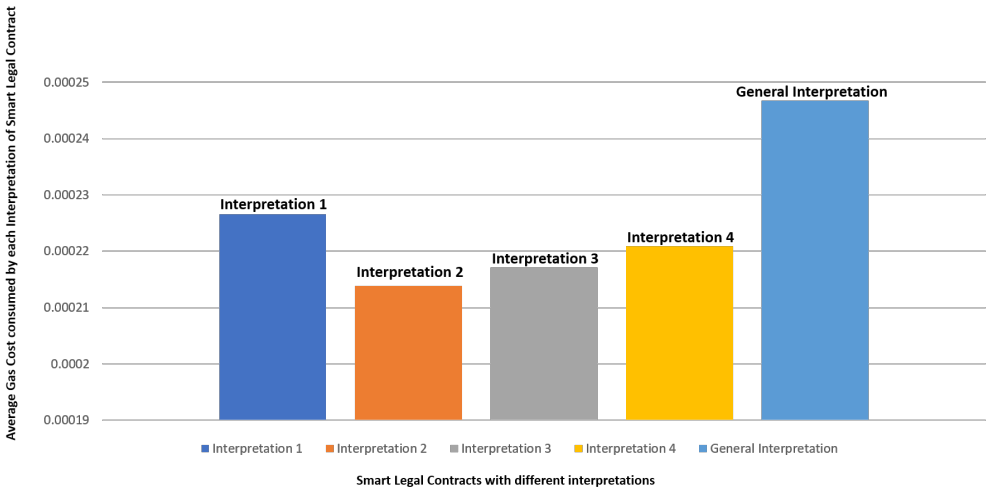

FIGURE 3.6. Comparison of average transaction cost by 5 different interpretations of Smart contract to find out the complexity of each Smart contract.

In Fig. 3.6, a comparison of all smart contracts with their respective interpretations has been made. 500 transactions were performed, 100 for each interpretation. As we can see in Fig. 3.6, Interpretation 1 consumed approximately 0.00023 ethers. Interpretation 2 has consumed the least, i.e., slightly more than 0.00021 ethers. Interpretation 3 has always been very similar to Interpretation 2 in all aspects. Even the average transaction cost is similar, i.e., slightly more than what Interpretation 2 costs. Interpretation 4 has consumed approximately 0.00022 ethers. However, smart contract with General Interpretation has consumed

the most among all, consuming slightly more than 0.000245 ethers. As demonstrated in Fig. 3.6, since the General or Root Interpretation is more vague and complex, the consumption rate is higher compared to other interpretations.

The reason smart contract with General Interpretation consumed more gas for transaction fees than the rest of smart contracts is because it is more complex and has more lines of code. And the only reason it is more complex is because it is more vague. We also discuss finding out the most vague interpretation previous section, where we calculate the vagueness index based on the complexity level of each smart contract to strengthen our observations and conclusion. From this transaction fee consumption pattern, we can say that the smart contract with General Interpretation is much more vague and complex contract than smart contracts with the other four interpretations.

TABLE 3.1. Complexity measure of Crowdfunding Smart contracts

Type of Smart contract	**Complexity Measure (Vagueness Index)**
Interpretation 1	1
Interpretation 2	1
Interpretation 3	1
Interpretation 4	1
Root Interpretation	3

3.6.3. Measurement of Complexity and Vagueness Index of Each Smart Contract

We have also calculated and measured the complexity of all five different interpretations. We have used McCabe's cyclomatic complexity in order to find the complexity of each interpretation. The relationship between complexity and vagueness is directly proportional, whereas vagueness and accuracy are inversely related. The more complex an interpretation is, the more vague it becomes. We have used the control flow graphs from Fig. 3.4 and Fig. 3.8 to calculate the complexity. To evaluate the complexity, we used McCabe's cyclomatic

complexity. The cyclomatic complexity is defined in [116], which measures the complexities and the total number of linearly independent paths of a program.

(1) $$C = N_e - N_n + 2 * N_{cc}$$

Where, C is the complexity, N_e is the number of edges of the control flow graph, N_n is the number of nodes of the control flow graph, and N_{cc} is the number of connected components.

As we can see in Table 3.1, smart contracts with Interpretations 1, 2, 3, and 4 have the same complexity measure, i.e., 1. This means that when measuring the vagueness index of smart contracts with Interpretations 1, 2, 3, and 4, we found that they are equally vague at the same level. However, the complexity measure for the smart contract with General Interpretation is 3. Hence, the vagueness index for General Interpretation is three times more than that of the other four smart contracts. Therefore, from Fig. 3.4, Fig. 3.8, Fig. 3.6 and Table 3.1, we have measured the complexity level of each smart contract with five different interpretations and found out that the most vague one is the smart contract with General Interpretation which makes it less accurate. In other words, smart contracts with Interpretations 1, 2, and 3 are more accurate compared to the smart contract with General Interpretation.

Not only we calculated McCabe's cyclomatic complexity for Crowdfunding Legal Contract, but we also took an Employment Agreement Contract, and we performed our test on it to find the complexity measure and vagueness index to see if the level of vagueness is also greater in General Interpretation for Employment Agreement Contract. From the Employment Agreement Contract, we generated a general interpretation along with 10 more different interpretations for test purposes, although even more interpretations could be generated. The more vagueness is in the legal contract, the more interpretations can be generated from it.

From Table 3.2, we can see that the smart contract with the general interpretation has a vagueness index of 5. On the contrary, the smart contract with the rest of the in-

terpretations have the same level of vagueness index, hence have equal vagueness index. Therefore, we can conclude the smart contract with the General Interpretation always has higher vagueness index compared to other interpretations because it comprises words and phrases with multiple meanings and is full of vagueness or least accurate.

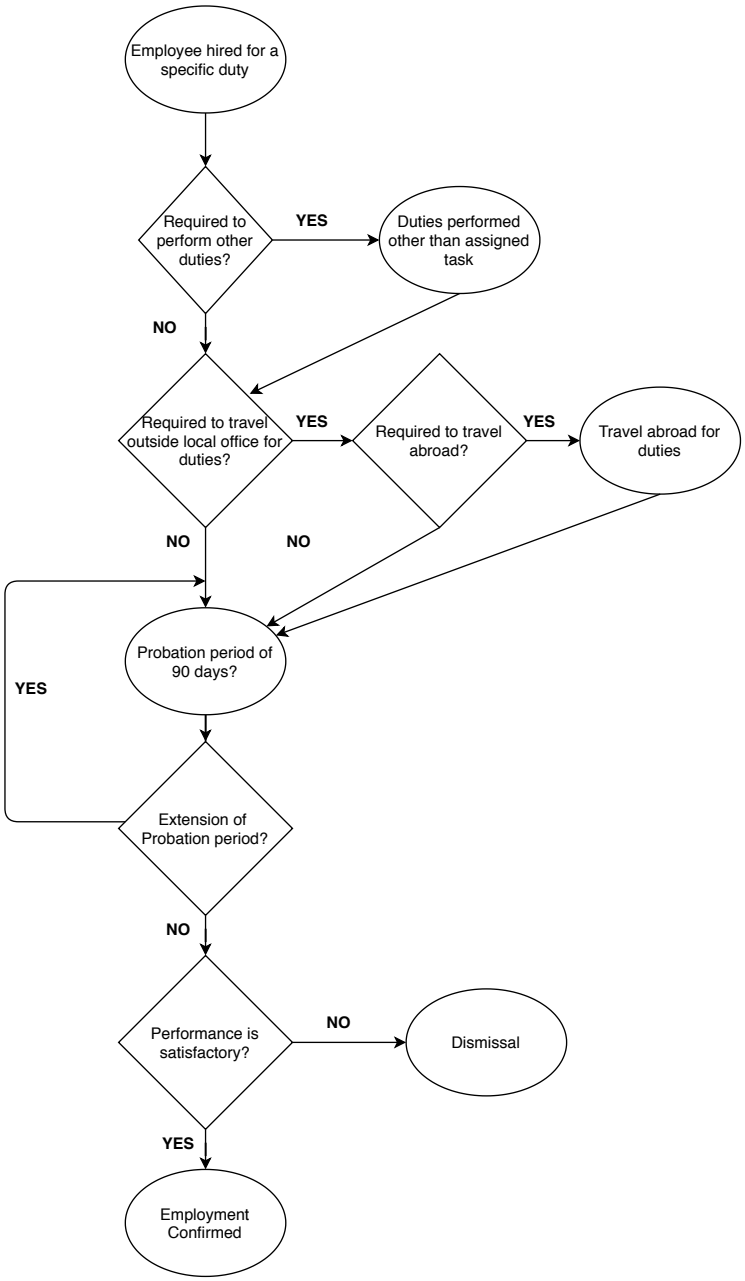

FIGURE 3.7. Control flow graph of the events from Employment Agreement Contract (Root Interpretation).

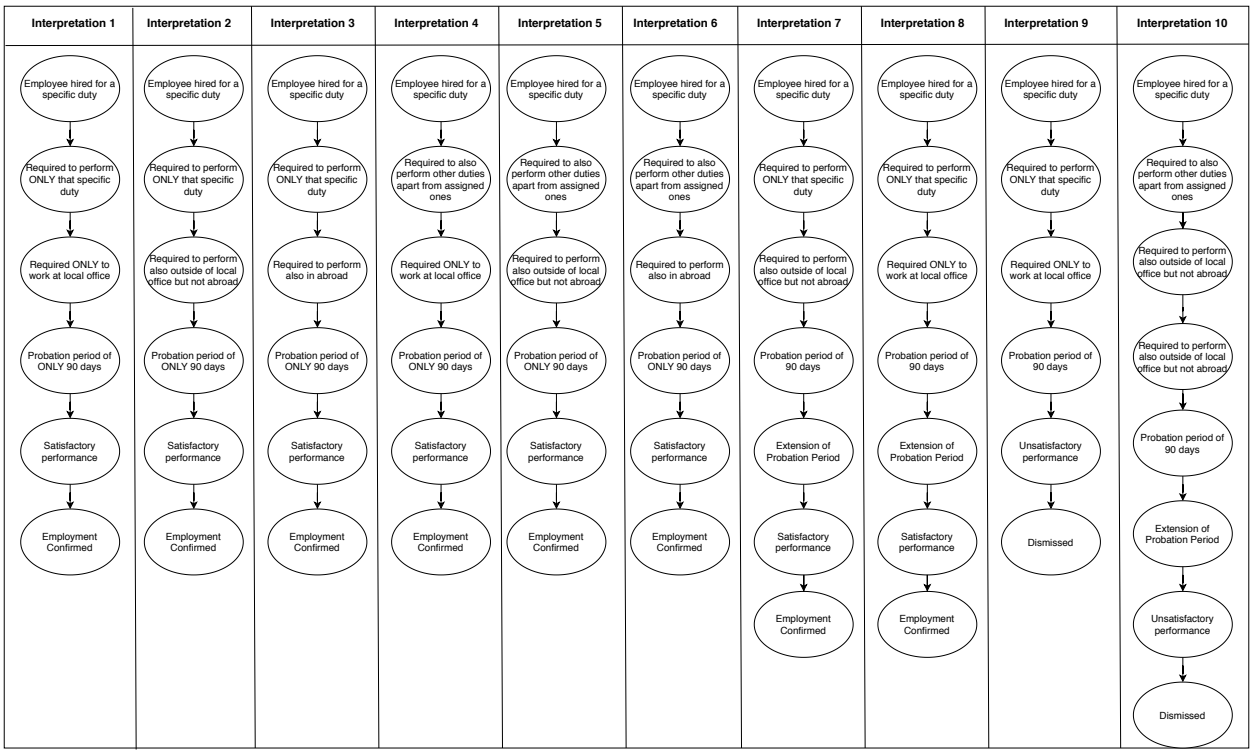

FIGURE 3.8. The variation in control flow graphs showing multiple interpretations from Fig. 3.7's control flow graph.

In both contracts, the contract with general interpretation scored highest, meaning the contract with general interpretation is more vague than any other interpretations in any given contract. Our observations and the comparison between the vagueness index from Table 3.1, Table 3.2, Fig. 3.7 and Fig. 3.8, shows that the employment agreement smart contract is more vague than the crowdfunding smart contract because of the higher vagueness index.

3.6.4. Total Translation Percentage of a Legal Contract

There are altogether of 12 clauses in our test crowdfunding legal contract. 4 out of 12 clauses have been successfully converted into the smart contract. In other words, we can say that we were able to convert 33.33% of the total contract into the smart contract, as shown in Fig. 3.9. The clauses that have been converted are "Agreement", "The Project", "Rewards", and "Contribution and Payment" which revolves around the idea and mechanism of the crowdfunding process.

TABLE 3.2. Complexity measure of Employment Agreement Smart contracts

Type of Smart contract	**Complexity Measure (Vagueness Index)**
Interpretation 1	1
Interpretation 2	1
Interpretation 3	1
Interpretation 4	1
Interpretation 5	1
Interpretation 6	1
Interpretation 7	1
Interpretation 8	1
Interpretation 9	1
Interpretation 10	1
Root Interpretation	5

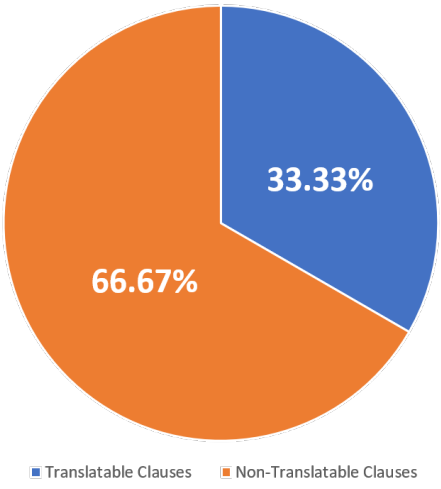

FIGURE 3.9. Total translation percentage of a whole Crowdfunding Legal Contract into Smart contract.

The clauses that were not converted were not related to the mechanism and function-

ing of a crowdfunding process and hence did not contribute much when it came to writing the smart contract. The whole contract cannot always be converted into a smart contract as the activities, events, and other major aspects in a legal contract also include physical and non-transactional activities. In that case, we only take the subset of the legal contract and convert the convertible subset into the code, i.e., smart contract.

3.7. Conclusion

In this chapter, we introduced a novel study on the relationship between a vague legal contract and a smart contract. We also created all possible interpretations from a vague legal contract and then evaluated and compared different metrics that helped us to ultimately find the most vague as well as accurate interpretation. By assessing the transaction fees and vagueness index of all the possible interpretations of the smart contract, we were able to strengthen our final conclusion and point out whether a given interpretation of a smart contract was accurate or vague. We also compared two legal contracts and found which contract was more vague than the other. We also studied the total translation rate of a traditional legal contract into a smart contract and what type of clauses are more likely to be converted to computer code easily. The main purpose of this chapter is to study how a legal contract in the real world has been affecting people's lives in different ways by being vague and vague and how we can convert a given legal contract into a smart contract and leverage blockchain technology to make the work efficient and effective.

CHAPTER 4

ANALYSIS OF THE CONVERSION PROCESS OF TRADITIONAL SERVICE-LEVEL AGREEMENT (SLA) OF ISP VENDORS INTO SMART CONTRACT[1]

4.1. Introduction

A service-level agreement (SLA) is a legal contract between a vendor and its customer which defines the quality of service that the vendor promises to provide to its customers in exchange for their subscription and payment. If the vendor fails to provide the level of service to its customers that have been defined in their SLA, then the vendor will be penalized, and they will have to provide compensation to the customers that are also defined in the SLA. In other words, SLA is viewed as an important component of a technology vendor's legal contract.

However, since an SLA is also a type of traditional legal contract [179], it is full of vague terms and legal jargon that makes it hard for the vendor's customer to understand the precise meaning. Oftentimes, we hear and see many reviews, news, and incidents where customers complain against ISPs about not getting their internet service in exchange for what they are paying for [140], [91], [60]. We have also heard customers spending their time and money to request compensation and service credit from their vendors. However, due to the vague and equivocal nature of the SLA, it becomes difficult for the customers to get their refunds back.

A legal contract is vague when a specific term, word, phrase, or definition is not precise and hence results in multiple meanings [3]. Since most SLAs are vague and lack a precise set of metrics by which the service is measured as well as the indemnification clause results in multiple interpretations when multiple people from different linguistic backgrounds and experiences read them. As a result, the customers always have a hard time getting their

[1]This chapter is presented in its entirety from K. Upadhyay, R. Dantu, Y. He, S. Badruddoja and A. Salau, "Can't Understand SLAs? Use the Smart Contract," 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), Atlanta, GA, USA, 2021, pp. 129-136, doi: 10.1109/TPSISA52974.2021.00015.

compensations back from the vendors when they do not get the service they have subscribed for due to the absence of self-enforcement property and precise usage of words.

On the contrary, a Smart legal contract or a Smart contract (SC) is a kind of contract where the agreements are self-enforcing and are embedded in computer code that is managed by the blockchain [26]. There are a clear and precise set of rules under which the parties involved in the smart contract agree to interact with each other. If and when the predefined rules that are written in the smart contract as code are met, there will be automatic enforcement of the agreements.

Vagueness is an important issue when formalizing contractual clauses, and we propose a formal method to find out vague terms in SLA contracts using machine learning and then convert those vague SLA contracts into Ethereum-based smart contracts. Thus, the main problem definition of this chapter is how we can analyze and compare the vague nature of different SLAs, particularly broadband vendors' SLAs that are full of vague words that result in multiple interpretations for different people. Besides, we also discuss how we can convert these vague SLA contracts into non-vague and smart contracts that can be used in Ethereum-based Blockchain as the Blockchain is decentralized and distributed and also eliminates the need for middlemen such as lawyers and legal attorneys.

4.2. Contributions

The chapter's main focus is contract interpretation due to the vagueness present in a contract. The main contributions of this chapter are as follows:

- We investigate the vague nature of SLAs by taking six samples of real SLAs from different popular vendors of the same industry, i.e., six different SLAs from six different internet service providers (ISPs) that are AT&T [130], Verizon [135], Spectrum [133], T-Mobile [134], Ziply Fiber [136], and CenturyLink [131].
- We use machine learning to train the model from the SLAs of AT&T, Verizon, Spectrum, and T-Mobile as training dataset to detect vague words from the SLAs of Ziply Fiber and CenturyLink as testing dataset. *The SLAs for training and testing were chosen randomly.*

- We manually transform Ziply Fiber and CenturyLink's vague SLAs into their respective control graph and further derive all possible special case interpretations from the root control graphs.
- We translate the generated control flow graphs of both vendors' SLAs and all interpretations into their corresponding smart contracts for each interpretation for Ethereum-based Blockchain.
- We measure the vagueness of each translated smart contract based on their performance in the Ethereum-based Blockchain network and compare which vendor's SLA is more vague out of the two test SLAs.
- Additionally, we measure the vagueness by using Shannon's Entropy [152] and McCabe's cyclomatic complexity [116] to generate the vagueness index for each interpretation of a smart contract of both vendors' (Ziply Fiber and CenturyLink) SLAs based on the complexity of the control flow graph of each interpretation for both test SLAs.
- We compare the performance of the smart contracts of both broadband vendors, i.e., Ziply Fiber and CenturyLink, and we then identify their most vague as well as accurate interpretation of the smart contract.
- Finally, we identify which vendor's smart contract is more vague in general among Ziply Fiber and CenturyLink.

4.3. Relationship between a Traditional Service-Level Agreement (SLA) and a Smart Contract

A service-level agreement (SLA) is written by a vendor, but it is also written so that the customers can measure that the service they are getting is how it is exactly defined in the SLA. Unfortunately, an SLA consists of affluent ambiguous, vague, and fuzzy legal terms. Hence, SLA results in various interpretations when different customers read them because of their different experiences and knowledge. An SLA drafted by the legal department of a vendor is written in such a way that it is full of jargon terms that only the people who are involved in legal aspects can understand the SLA.

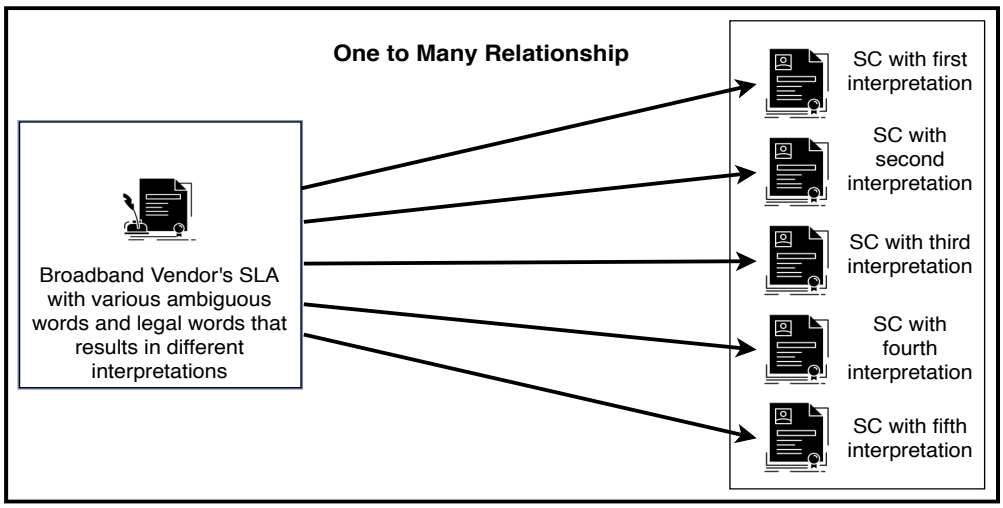

FIGURE 4.1. One to many relationship between an SLA and smart contract

Also, the service metrics that are defined in the SLA, which describes how much service and what kind of service the customers are expected to get after they subscribe for it, need to be clearer for the customers to understand. In addition, even though the service metrics are written as clearly as possible, there will still be plenty of words such as *"may", "might", "reasonable", "best efforts", "most likely"*, and so on in the indemnities section when it comes to giving the compensation back to the customers for bad service [151]. Therefore, as shown in Fig. 4.1, due to the presence of vague words and structure in the SLA contract, different people perceive the same contract differently.

These kinds of vague terms as well as the way the service metrics are defined in the SLA, create multiple interpretations. For example, one customer from different background and experience might understand the same SLA differently than the other customer who reads it. The main cause of these multiple interpretations from the same SLA is the way it is drafted and the vague words contained in it. On the other hand, a smart contract is clear, precise, and straightforward. In Fig. 4.1, we can see the one-to-many relationships between an SLA contract and a smart contract. This figure describes the type of relationship between a traditional SLA contract and a smart contract and how one SLA can be interpreted in various ways due to the vague words present in it. Hence, several different versions of the smart contract can be translated from a vague SLA, which is written in vague natural

language. The more vague an SLA is, the more interpretations it will have and the more possibility of generation of different interpretations of smart contracts.

4.4. Experimental Setup

The tools and materials that we have used for this project are listed below:
i) Ropsten Test Network, ii) Solidity Programming Language, iii) Remix Web IDE, iv) Metamask, v) Node.js, vi) Truffle, vii) Ganache-CLI, viii) Web3, ix) HD Wallet, x) Google Chrome in Incognito Mode, xi) Python 3, Anaconda and Jupyter Notebook [146] xii) Support Vector Machine [44], xiii) AT&T's SLA, Verizon's SLA, Spectrum's SLA and T-Mobile's SLA as training SLA, and xiv) Ziply Fiber's SLA and CenturyLink's SLA as test SLA.

4.5. Methodology

We have divided our entire methodology into six phases, as shown in Fig. 4.2 and Fig. 4.3.

In our first, second, and third phases, as shown in Fig. 4.2, we read the texts in the two SLAs and use binary classification to classify the vague words from non-vague words by using machine learning. Apart from being a part of future work and research, the reason machine learning is used instead of manual hand-picking of vague words and phrases is that we wanted to automate the extraction process of vague words and phrases and evaluate the performance. Therefore, in our first phase, we gather different SLAs from different vendors but from the same industry so that we can create a training dataset for the machine to learn the kind of vague words being used in the SLAs. We have gathered six different SLAs from six different popular ISP (broadband) vendors, which are AT&T [130], Verizon [135], Spectrum [133], T-Mobile [134], Ziply Fiber [136], and CenturyLink [131]. The reason all the SLAs from ISP vendors were chosen and not mix from other vendors such as insurance companies was primarily to get unbiased results and to translate the SLAs of ISP into the Ethereum-based smart contracts so that customers would benefit from automated compensation system and would not have to face any difficulty to get the indemnities, penalties, and compensations when they do not get their services as they were promised in the SLA contract.

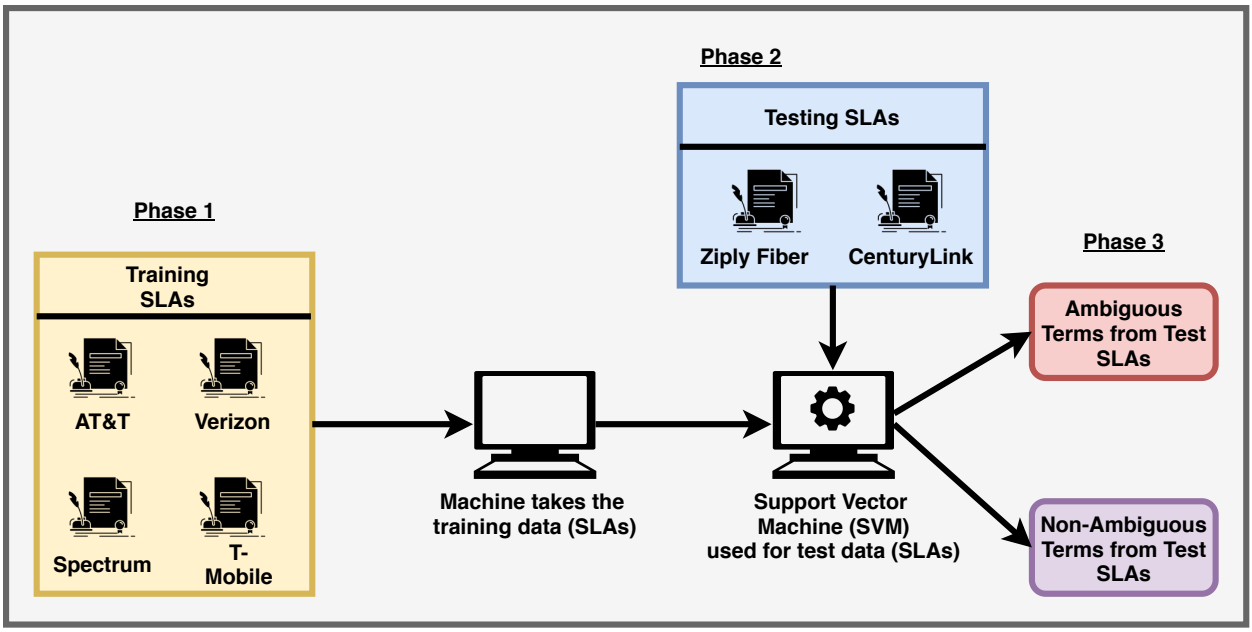

FIGURE 4.2. Selection of six SLAs from six different vendors from the same industry and categorizing them into train and test data sets in Phase 1 and 2 and using Support Vector Machine (SVM) to detect and classify vague and non-vague terms from test data, i.e., Ziply Fiber and CenturyLink in Phase 3.

We have categorized the SLAs of AT&T, Verizon, Spectrum, and T-Mobile as the training dataset, while Ziply Fiber and CenturyLink were categorized as testing dataset. There were no hard-and-fast rules to decide what SLAs will be as training dataset and what SLAs will be testing dataset. The selection of both the training and testing SLAs is done randomly. We created a script to read all texts in the SLA documents, tokenize all the words present in the documents, and finally prepared the training dataset by labeling the tokens manually as vague (1) or non-vague (0).

In Phase 2, we classify the SLAs of Ziply Fiber and CenturyLink as test SLAs meaning all the words extracted from these two SLAs were used to prepare the test dataset. Support Vector Machine (SVM) [44] was used to train the model for it to perform binary classification and detect the vague words from non-vague words as shown in Phase 3 of Fig. 4.2. After experimenting and testing with other common machine learning algorithms such as Random Forest, Decision Tree, and kNN, we got the highest accuracy from SVM. As a result, we

decided to use SVM for binary classification of vague words in test data, i.e., tokens from Ziply Fiber's and CenturyLink's SLA contracts.

As shown in Phase 4 of the Fig. 4.3, after we finish detecting all the possible vague words and phrases in our two test SLAs (Ziply Fiber and CenturyLink) using machine learning, we manually generate different possible interpretations from those machine detected vague terms as shown in Phase 4. One of the main objectives of this study was to create as many as possible human interpretations people will have while reading the SLAs of the ISP vendors, convert all the interpretations into the Ethereum-based smart contract, and finally find out which version or the interpretation of the smart contract is more vague and accurate along with finding which SLA in average is more vague.

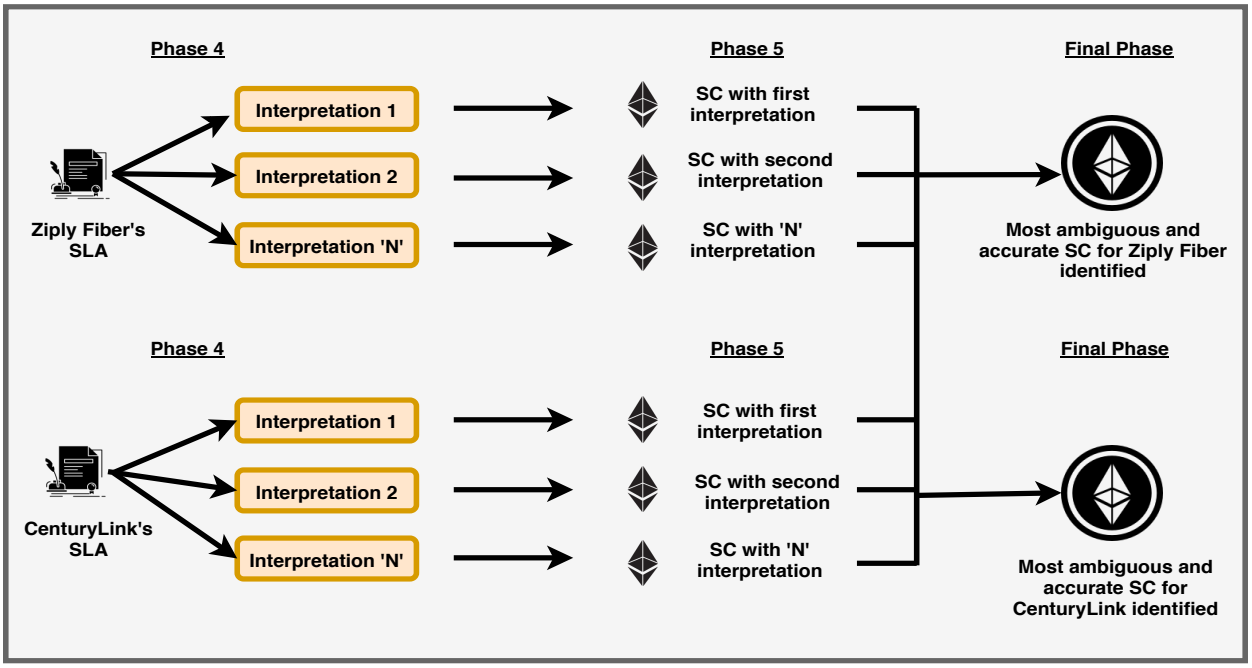

FIGURE 4.3. Generation of all possible different interpretations of both test SLAs in Phase 4, translation of all generated interpretations from both vague test SLAs into their respective smart contracts in Phase 5, and comparison and identification of the most vague and accurate interpretation from each test SLA along with the most vague and accurate SLA out of the two in Final Phase.

The classification or detection accuracy of the model while classifying the vague words in Ziply Fiber's SLA was 85% and in CenturyLink's SLA was 79%. Although increasing the accuracy is our top priority and part of our future work, we have considered only those vague words that the machine has detected successfully to generate various interpretations for translating those interpretations into their corresponding smart contracts. We translate all those generated interpretations from the vague words that were detected using machine learning into their respective smart contracts, as shown in Phase 5 of Fig. 4.3. Finally, as shown in Phase 6 or the final phase in Fig. 4.3, we perform various tests of the translated smart contracts of all the interpretations of both SLAs, and we find out what interpretation of each SLA and what SLA as a whole is the most vague as well as the most accurate one.

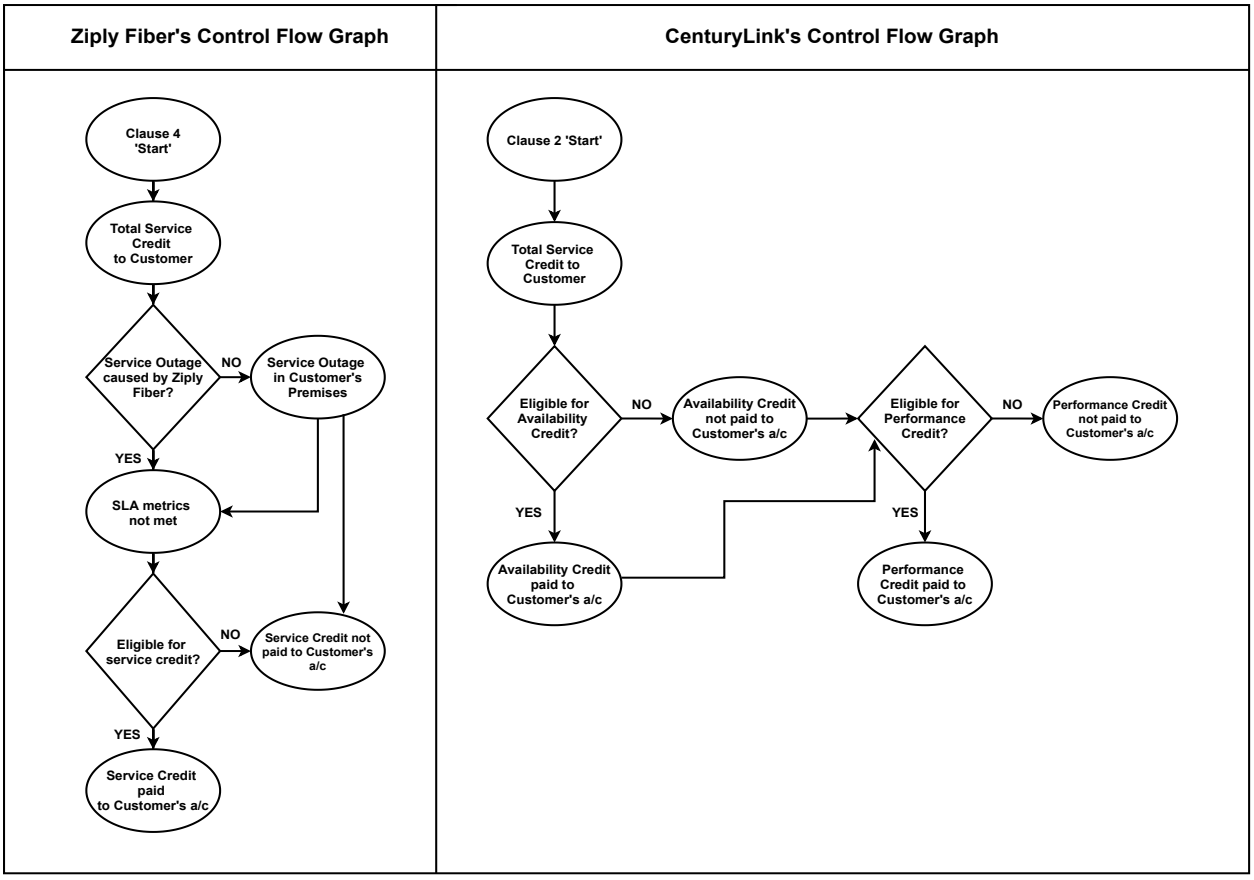

FIGURE 4.4. From these CFGs, 5 other special case interpretations for Ziply Fiber's SLA and 4 special case interpretations for CenturyLink's SLA will be generated.

Fig. 4.4, 4.5, and 4.6 describe Phase 4 of our methodology in more detail. The figure that we see in Fig. 4.4 is the control flow graphs that we generated from Ziply Fiber's SLA and CenturyLink's SLA, considering only the vague words that the machine detected after classifying the vague from non-vague terms. We manually generated control flow graphs of these vendors so that we could also generate all possible special case interpretations from these control graphs. The control graphs in Fig. 4.4 explain how vague the SLA of Ziply Fiber and CenturyLink is by portraying multiple branches in the control graph. We have named this version of the control graph as **root control graphs** as this was our first step to derive the control flow graph from Ziply Fiber's and CenturyLink's SLA contract. The control graphs from Fig. 4.5 and 4.4 are named as **special case interpretation control graphs** as these are generated further from the root control graphs.

The sentences present in Ziply Fiber's SLA contract such as *"In the event of a Service Outage, Customer may be entitled to a credit against the applicable On-Net Service MRC"* and *"Credits do not apply to Service Outages caused, in whole or in part, by one or more of the following."* increases the degree of vagueness. Here, the words such as *"may"* and *"in whole or in part"* lead to more than one interpretation of the whole SLA of Ziply Fiber because these are permissive terms. It also describes different actions and events that might take place depending on the understanding of the customers who read the SLA. Hence, we first generate the root control graph of Ziply Fiber's SLA as shown in the left column of Fig. 4.4 as well as CenturyLink's root control graph as shown in the right-column of Fig. 4.4. Hence, this would be the case of how the SLA will look where there are multiple *"yes"* and *"no"* because of the involvement of vague words, which results in multiple branches in the control flow graph.

From Fig. 4.4, we further generate more special case interpretations. All 5 of them for Ziply Fiber are shown in Fig. 4.5. Hence, Fig. 4.5 shows how we have generated other interpretations further from Fig. 4.4. Similarly, Fig. 4.6 shows the vague nature of CenturyLink's SLA as well by portraying multiple possible branches the decisions, events, and actions can have.

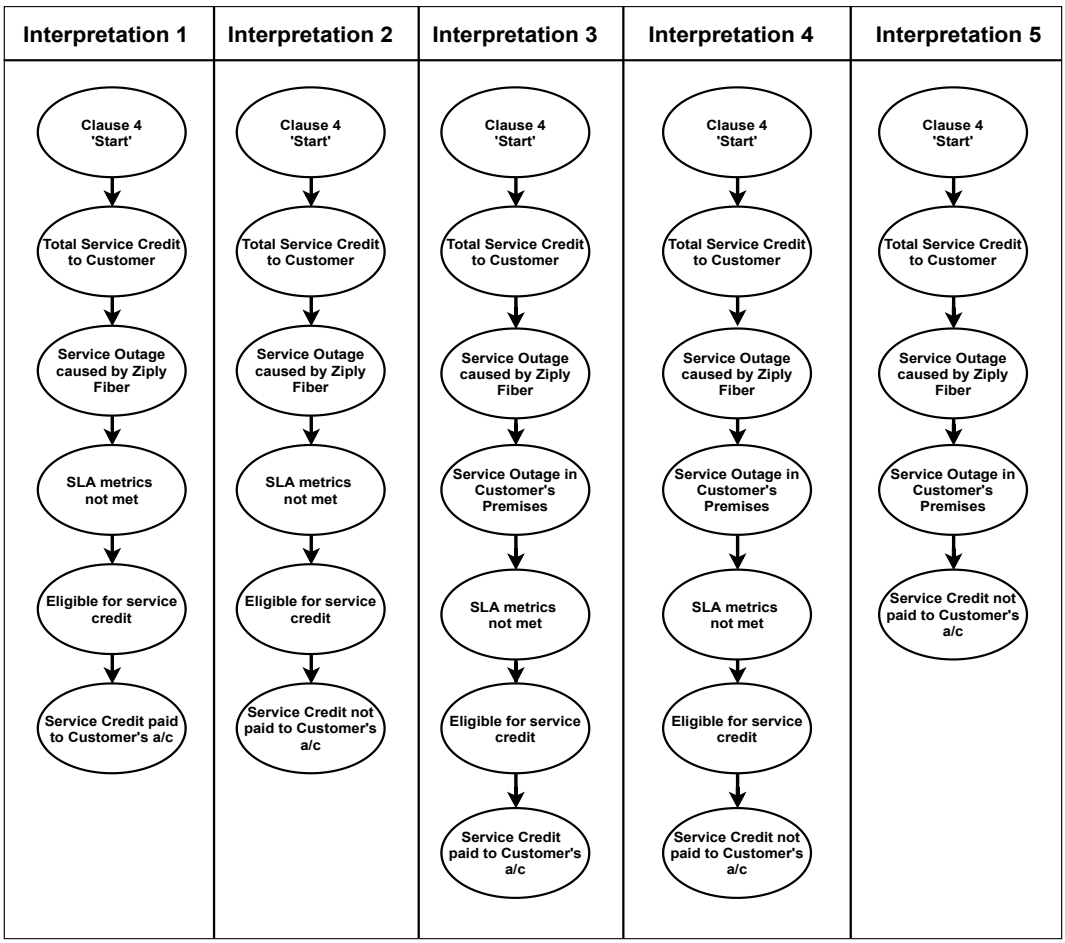

FIGURE 4.5. Derivation of five special cases of interpretation from Ziply Fiber's control flow graph.

Fig. 4.6 only has four interpretations, as CenturyLink's SLA had fewer nodes, edges, and connected nodes. It was the first control graph that we derived from the SLA. The sentence such as *"If Service performance falls below the thresholds provided in Table 2.0 and CenturyLink is unable to rectify the performance of the Service(s) at the Affected UNI within 30 business days then Customer may be eligible for a Performance Credit for Service degradation subject to the rules and exclusions provided in this agreement."* along with other vague sentences are used in CenturyLink's SLA, which allows customers to form multiple interpretations further that we have discussed in Fig. 4.6. In Fig. 4.6, we have four possible special case interpretations that can be generated from Fig. 4.4's root control flow graph (right-column). This further generation of interpretations was possible due to the usage of

permissive and vague words in CenturyLink's SLA. This case is similar to the case of Ziply Fiber's SLA.

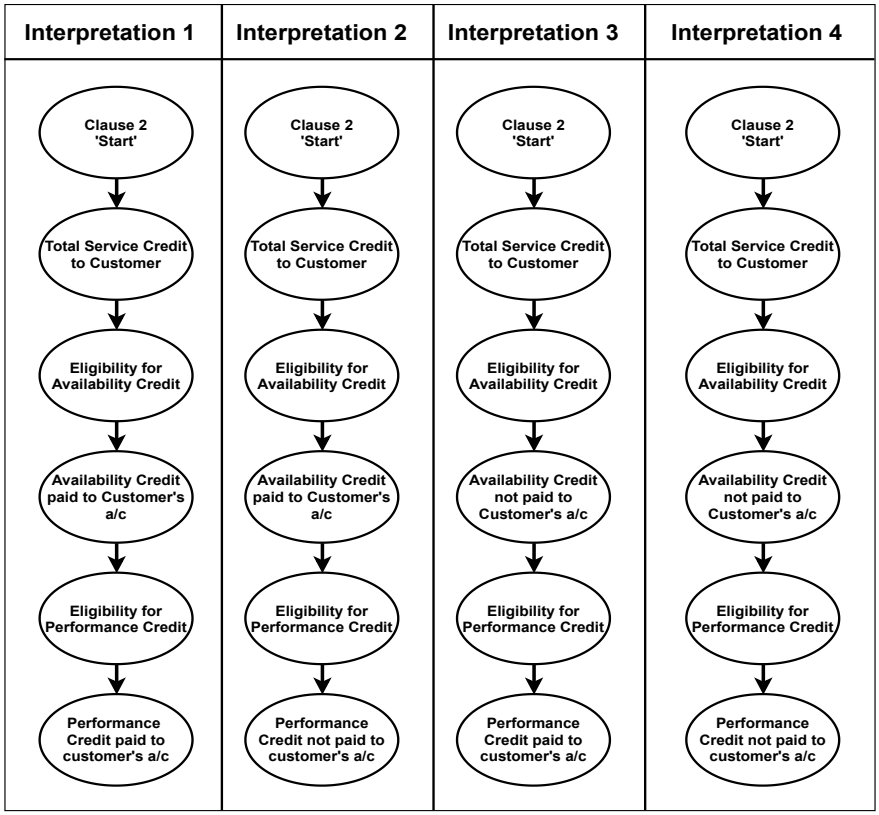

FIGURE 4.6. Derivation of five special cases of interpretation from centuryLink's Control flow graph.

If mandatory words had been used instead of permissive and vague words, then the root control graphs in Fig. 4.4 would be more straightforward without different branches. Once we derived and generated all possible special cases interpretations further from Ziply Fiber and CenturyLink's root control flow graph as shown in Fig. 4.4, 4.5 and 4.6, we translated both the root control graphs and special case control graphs from both vendors into their respective smart contracts.

4.6. Results

4.6.1. Vagueness and Complexity Measurement by Smart Contract Deployment

We translated the root control graphs from Fig. 4.4 to analyze which vendor has a more vague SLA in general. We translated the control flow graph of Ziply Fiber and CenturyLink into their respective SLA and deployed their smart contract in Ropsten Testnet 10 times each. As we can see in the Fig. 4.7, the TXN cost of Ziply Fiber was 0.031516123 ETH.

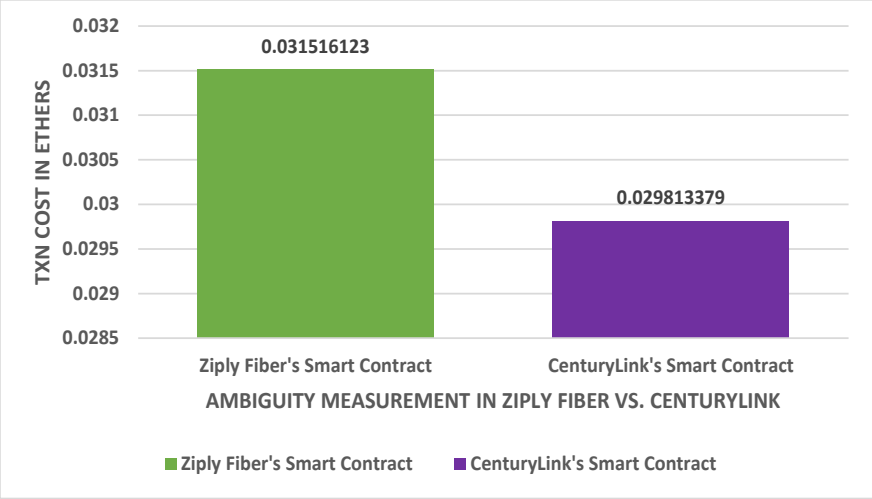

FIGURE 4.7. Comparison of the vagueness in SLAs from two different vendors which shows that Ziply Fiber's SLA is more vague than CenturyLink's SLA.

However, as the size of the control flow graph for CenturyLink was small and had less number of interpretations compared to Ziply Fiber, the TXN cost for CenturyLink was just 0.029813379 ETH. Then we deployed all five special case interpretations of the smart contract of Ziply Fiber 10 different times in Ropsten Testnet. We have made the comparison of transaction (TXN) costs of all smart contracts with their respective interpretations. Fig. 4.8 shows the average of all the registered TXN costs of all five special case interpretations of Ziply Fiber's smart contract in Ropsten Testnet. Interpretation 1 had the average TXN costs of 0.024063215 ethers (ETH). Similarly, Interpretation 2 had average TXN cost of 0.021482481 ETH. Likewise, Interpretation 3 and 4 had 0.020106104 ETH and 0.025117192 ETH, respectively. Interpretation 5's average TXN cost was the lowest because of its con-

trol flow graph size, i.e., 0.014882172 ETH. We observed that all these TXN costs of their respective interpretations correlate to the size of control graphs as well.

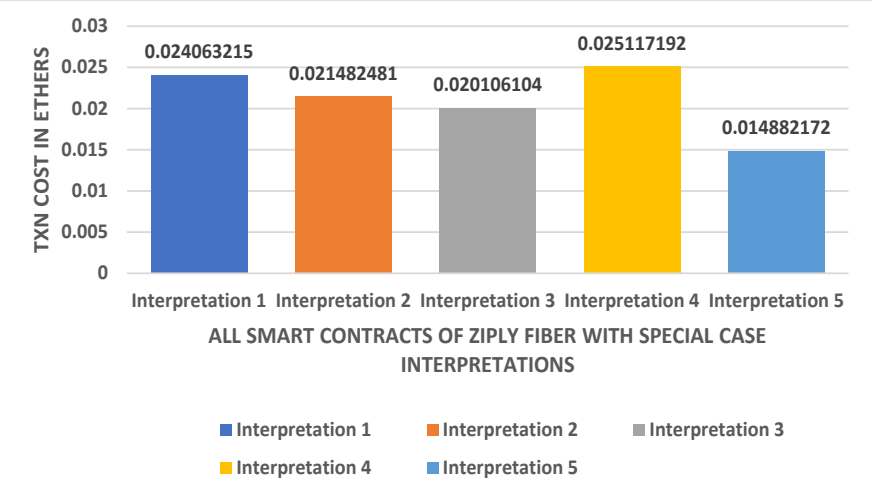

FIGURE 4.8. Comparison of average TXN cost of 5 different special case interpretations of Ziply Fiber's Smart contract for the measurement of vagueness and complexity.

Similarly, we deployed all four special case interpretations of the smart contract of CenturyLink 10 different times as well in Ropsten Testnet. If we take a look at Fig. 4.9, we can see that the average TXN cost of Interpretation 1 is 0.01572939 ETH. Likewise, the TXN cost of Interpretation 2, 3 and 4 are 0.017547318 ETH, 0.013452181 ETH and 0.014446134 ETH respectively.

From our study, we found that the reason Ziply Fiber consumed more TXN cost than CenturyLink was that it is more vague. Vagueness is directly proportional to the complexity of the smart contract, which means if the vagueness of a certain interpretation rises, the lines of code along with the program complexity will also rise, which will result in the increment of the TXN and gas cost. As we can also see in Fig. 4.4 and 4.5, the control flow graph of Ziply Fiber was more complex and had more interpretations. The main reason for this was the vague nature of Ziply Fiber was more compared to CenturyLink's smart contract.

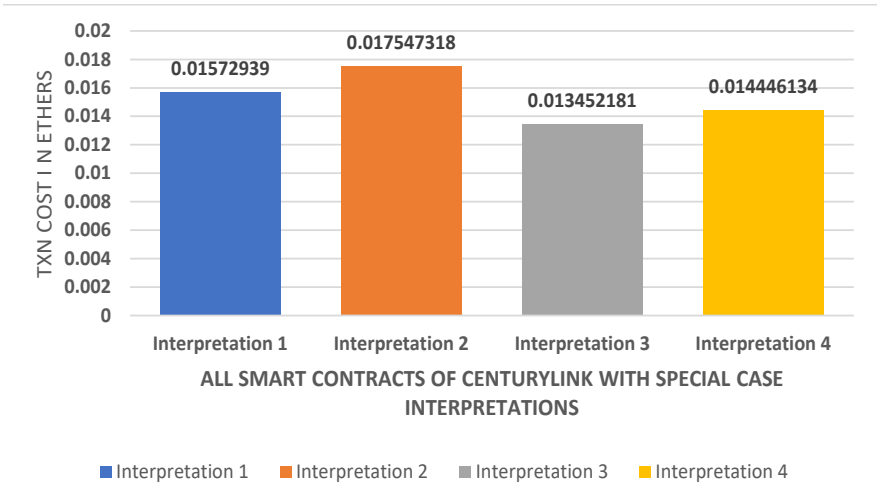

FIGURE 4.9. Comparison of average TXN cost of 4 different special case interpretations of CenturyLink's Smart contract for the measurement of vagueness and complexity.

Therefore, from this observation, we can say that if a particular interpretation is more vague in nature, it is more complex in the control graph as well. In addition, while translating the control graph into the smart contract, due to the SLA's vagueness as well as complexity, the smart contract of that very SLA consumed more TXN and gas cost as we can see in Fig. 4.7, 4.8 and 4.9.

4.6.2. Vagueness and Complexity Measurement by Entropy and Cyclomatic Complexity

To corroborate our evaluation of the proportional relationship between vagueness and TXN costs, we have also studied both entropy and cyclomatic complexity of Ziply Fiber's and CenturyLink's SLA along with respective interpretations, which helped us to find their respective vagueness indexes.

We have used Shannon's Entropy and McCabe's cyclomatic complexity to find the uncertainty and complexity of both vendors' SLAs. We have used the control flow graphs from Fig. 4.4, 4.5, and 4.6 to find the entropy and cyclomatic complexity. The Shannon's entropy measures the average level of information and uncertainty which is in variable's possible's outcomes. Similarly, cyclomatic complexity measures the complexities and the total number of linearly independent paths of a program.

TABLE 4.1. Entropy measurement of Ziply Fiber's SLA control graph and its Special Case Interpretations

Type of Smart contract	**Entropy Measure (Uncertainty Index) of Ziply Fiber's SLA**
Ziply Fiber's root SLA	1.6094
Interpretation 1	0
Interpretation 2	0
Interpretation 3	0
Interpretation 4	0
Interpretation 5	0

(1) Shannon's Entropy:

The Shannon's entropy [152] is defined as:

(2) $$H(X) = -\sum_{i=1}^{n} P(x_i) log P(x_i)$$

Where, $H(X)$ is the entropy of X, $\sum_{i=1}^{n}$ is the sum over variable's possible values, *log* is the natural logarithm, $x_1 ...., x_i$ are possible outcomes and $P(x_i)$ is the probability of the occurrence.

TABLE 4.2. Entropy measurement of CenturyLink's SLA control graph and its Special Case Interpretations

Type of Smart contract	**Entropy Measure (Uncertainty Index) of CenturyLink's SLA**
CenturyLink's root SLA	1.3863
Interpretation 1	0
Interpretation 2	0
Interpretation 3	0
Interpretation 4	0

We calculated Shannon's Entropy for both control flow graphs of Ziply Fiber and CenturyLink from Figure 4. To calculate Shannon's entropy, we considered the number of special case interpretations each SLA (root control graph) can generate. For example, the number of special case interpretations from Ziply Fiber (Fig. 4.5) is 5. Hence, each special case interpretation is assumed to have a 1/5 probability of occurrence. Likewise, the number of special case interpretations from CenturyLink (Fig. 4.6) is 4.

Therefore, in this scenario, each special case interpretation is assumed to have a 1/4 probability of occurrence. We found out that entropy for Ziply Fiber was 1.6094 and for CenturyLink was 1.3863, as shown in Table 4.1 and 4.2. From this, we can say that the control flow graph and hence, the nature of Ziply Fiber is more uncertain and vague than the SLA of CenturyLink.

(2) McCabe's Cyclomatic Complexity:

The Mccabe's Cyclomatic Complexity [116] is defined as:

(3)
$$C = N_e - N_n + 2 * N_{cc}$$

Where, C is the complexity, N_e is the number of edges of the control flow graph, N_n is the number of nodes of the control flow graph, and N_{cc} is the number of connected components.

Both entropy (uncertainty) and complexity (vagueness) of Ziply Fiber's root SLA is higher compared to CenturyLink's SLA. From our observations and evaluations from Fig. 4.4, 4.5, 4.6, 4.7, 4.8, and 4.9 and Table 4.1, 4.2, 4.3 and 4.4, we found that in both vendors' SLAs, smart contract of Ziply Fiber was more vague than smart contracts of CenturyLink.

TABLE 4.3. Complexity measurement of Ziply Fiber's SLA control graph and its Special Case Interpretations

Type of Smart contract	**Complexity Measure (Vagueness Index) of Ziply Fiber's SLA**
Ziply Fiber's root SLA	3
Interpretation 1	1
Interpretation 2	1
Interpretation 3	1
Interpretation 4	1
Interpretation 5	1

TABLE 4.4. Complexity measurement of CenturyLink's SLA control graph and its Special Case Interpretations

Type of Smart contract	**Complexity Measure (Vagueness Index) of CenturyLink's SLA**
CenturyLink's root SLA	2
Interpretation 1	1
Interpretation 2	1
Interpretation 3	1
Interpretation 4	1

4.7. Conclusion

We introduced a novel idea on how we can study the vague nature of legal contracts and service-level agreements (SLAs) of real vendors from the industry and how using smart contracts can help avoid the challenges of vagueness in traditional legal contracts. Regardless of how popular a vendor is, their SLAs can still be vague, imprecise, and vague, which can put a customer into a myriad of confusion and difficulty. In this chapter, we presented a fresh solution to an existing problem of vagueness in legal contracts by gathering real-

world SLAs of the top ISP vendors using a machine learning approach to train the model to detect vague words in legal contracts automatically. Since understanding a vague legal contract is difficult and it can create several different interpretations for several different people, we studied all the interpretations and their behaviors thoroughly from the SLAs of two different ISP vendors. We derived and generated all possible interpretations from the root SLA and then evaluated and compared different metrics, which helped us to find the most vague interpretation as well as the most vague vendor's SLA as a whole. We were also able to validate our final conclusion and decide whether a given interpretation of an SLA was accurate or vague by assessing the transaction fees and vagueness index of all possible interpretations of the SLA. Moreover, we also compared two different SLAs and found which one was more vague than the other. The main purpose of this chapter is to study how SLA contracts, even from popular vendors, can create confusion and different interpretations in different customers by being vague and how converting the traditional and vague SLAs into smart contracts can help us find the right interpretation of a legal contract.

CHAPTER 5

QUANTIFYING INTERPRETATION CERTAINTY VIA WEIGHTED FUZZY REASONING TECHNIQUE

5.1. Introduction

We have all been in that situation when we are constantly making complaints to the organization, especially internet service providers (ISPs), whenever their service gets constantly interrupted [112], [140], [60], [124]. For the subscription and the payment the customers have made to the ISPs, they are entitled to provide the customers with very good service in all aspects, at least what they have mentioned in their service-level agreement (SLA). However, often times customers end up not getting the kind of service that these ISPs and other sort of companies promise to them. In addition, when customers are exasperated with their service, the only help they can get is from the SLA of the company. As mentioned before, an SLA is a legal contract created by the vendor which defines the level of the service expected by the customer that describes the metrics by which their service is measured. An SLA also mentions that in case the services are not fulfilled to the customers properly, then customers are entitled to receive some form of compensation or remedies based on the measurement of provided metrics [154].

Nevertheless, as mentioned in previous chapters, companies always use vague and fuzzy words and legal jargon in their legal contracts and SLAs, while customers who are laypersons understand and prefer talking in everyday natural language. Due to this communication gap, when customers are trying to get further inquiries or support for their case, that leaves the customers even more puzzled. In that scenario, the customers who are not receiving proper service, as well as proper support from the company, are not getting any better help from the company-drafted SLA.

Hence, it is obvious that many companies' knowledge and many customers' complaints and compensation claiming interpretation always involve fuzzy concepts. Vagueness is a kind of phenomenon that is still an open research problem in the area of natural lan-

guage that has been studied throughout the years [21]. In this chapter, we have presented how with the application of Fuzzy Logic, we are able to quantify a dissatisfied customer's claim for compensation when they file complaints to the vendors based on their SLA. This method uses natural linguistic descriptors and can easily model the semantics of linguistic expressions. This mathematical model was inspired by the Mycin system [153] that was developed in the early 1970s at Stanford University by Edward Shortliffe and Bruce Buchanan which was used for the diagnosis of blood clotting disease and to help with bridging the gap between a physician's knowledge in medical diagnosis and patient's symptom manifestations.

5.2. Contributions

The main contributions of this chapter are as follows:

- We use and adapt from Shyi-Ming Chen's Weighted Fuzzy Reasoning Algorithm to quantify the certainty level of a claim made by a dissatisfied customer, that was originally used for the medical diagnosis [36].
- We generate the degree of truth for the knowledge representation.
- We investigate the concluded compensation based on the set of customers' complaints.
- We use similarity measures and the weighted vector method for the inclusion of the degree of importance.
- We illustrated with an example based on the weighted fuzzy reasoning technique how the confidence/certainty level of a customer's claim on compensation could be derived when the provided metrics of a company are not satisfactory.

5.3. Fuzzy Logic, Definitions, and Properties

Fuzzy logic was first proposed by Lotfi Zadeh in his 1965 paper, where he reflected on how to model logical reasoning with vague or imprecise statements [186]. Fuzzy logic is a method of reasoning that resembles human reasoning for representing and manipulating uncertain information. It has the ability to handle the concept of partial truth as in real life, as in real life, there are situations when the decision has to be made from inputs that

have gray areas. Unlike Boolean logic that only works on either "yes" or "no" or either "1" or "0", fuzzy logic is a many-valued logic that has truth values of variables in any real number between 0 and 1, inclusive [63]. It takes truth degrees as a mathematical basis on the model of the vagueness phenomenon. For instance, consider a statement "The weather is cold today." In Boolean logic's scenario, as it can only deal with either 1 or 0, if the word "cold" had to be quantified to understand and measure the above statement, then the value of the word "cold" only has two possible values, i.e., 1 or 0. In this above case, it would be a 1. However, every person has different knowledge representations and interpretations on different topics and, as a result of this, have different expressions in natural language. A person's tolerance and interpretation of cold weather can vary from another person who is used to even colder weather. Hence, instead of expressing natural language in either "is cold (1)" or "is not cold (0)", fuzzy logic allows the expression of natural language considering gray areas and degrees of truth using linguistic descriptors and their corresponding truth membership values. For example, "cold" can have different linguistic descriptors and corresponding membership values such as "extremely cold (1)", "very cold (0.85)", "somewhat cold (0.30)", or "not at all cold (0.00)".

With the kind of advantages Fuzzy logic has been offering, and by being a kind of explicit artificial intelligence model, we can see its application in numerous areas, ranging from anti-lock brakes and auto transmission developed by Nissan to a microwave oven developed by Mitsubishi Chemical, and many more [118], [115], [22], [172].

Following are the definitions of fuzzy sets, including the most common fuzzy relations and operations.

DEFINITION 5.1. A fuzzy set A of the universe of discourse U, $U = \{u_1, u_2, ..., u_n\}$ can be defined as a set of ordered pairs $\{(u_1, \mu_A(u_1)), (u_2, \mu_A(u_2)), ..., (u_n, \mu_A(u_n))\}$,
where, μ_A is the membership function of the fuzzy set A,

$\mu_A : U \longrightarrow [0, 1]$, and

$\mu_A(u_i)$ indicates the grade of membership of u_i in A;

$\forall u_i \in U$ the membership value $\mu_A(\mu_i)$ is a single value between zero and one [189].

DEFINITION 5.2. A fuzzy set A is empty if and only if its membership function is identically zero on the universe of discourse U [189].

DEFINITION 5.3. Let fuzzy sets A and B of the universe of discourse U and let y be the element of the universe. Let μ_A and μ_B be the membership values of element y in the fuzzy sets A and B, respectively. The union of the two fuzzy sets is denoted by:
$\mu_{A \cup B}(y) = max[\mu_A(y), \mu_B(y)]; \forall y \in U$
$\mu_{A \cup B} = \mu_A \cup \mu_B,$
where, $\cup$ represents maximum element in the set [189].

DEFINITION 5.4. Let fuzzy sets A and B of the universe of discourse U and let y be the element of the universe. Let μ_A and μ_B be the membership values of element y in the fuzzy sets A and B, respectively. The intersection of the two fuzzy sets is denoted by:
$\mu_{A \cap B}(y) = min[\mu_A(y), \mu_B(y)]; \forall y \in U$
$\mu_{A \cap B} = \mu_A \cap \mu_B,$
where, $\cap$ represents minimum element in the set [189].

DEFINITION 5.5. Let fuzzy sets A and B of the universe of discourse U and let y be the element of the universe. Let μ_A and μ_B be the membership values of element y in the fuzzy sets A and B, respectively. The complement of the fuzzy set A and B are respectively denoted by [189]:
$\mu_A(y) = 1 - \mu_A(y); \forall y \in U$
$\mu_B(y) = 1 - \mu_B(y); \forall y \in U$

5.4. Degree of Truth and Knowledge Representation

In the case of a dissatisfied customer due to inadequate services from a broadband vendor, knowledge can be represented after reading the SLA as follows:
IF complaints are filed based on different information, THEN concluded compensation ($CF = \mu_i$).

TABLE 5.1. Fuzzy quantifiers and their corresponding numerical intervals - Adapted from [36], [189], [187]

Fuzzy Quantifiers	**Numerical Intervals**
always	[1.00, 1.00]
very strong	[0.95, 0.99]
strong	[0.80, 0.94]
more or less strong	[0.65, 0.79]
medium	[0.45, 0.64]
more or less weak	[0.30, 0.44]
weak	[0.10, 0.29]
very weak	[0.01, 0.09]
no	[0.00, 0.00]

TABLE 5.2. Certainty levels and their corresponding numerical intervals - Adapted from [36], [189], [187]

Certainty levels	**Numerical Intervals**
absolutely certain	[1.00, 1.00]
extremely certain	[0.96, 0.99]
very certain	[0.86, 0.95]
pretty certain	[0.76, 0.85]
quite certain	[0.66, 0.75]
fairly certain	[0.56, 0.65]
more or less certain	[0.46, 0.55]
little certain	[0.30, 0.45]
very little certain	[0.16, 0.29]
hardly certain	[0.01, 0.15]
absolutely uncertain	[0.00, 0.00]

For example, let U be a set of complaints filed based on different metrics in service-level agreements (SLA), U = {Performance, Operation, Availability, Latency, Jitter, Maintenance}, and Compensation be a concluded outcome, then this knowledge can be represented by the rule R_1 as follows:

R_1: IF {no Performance Λ no Operation Λ very weak Availability Λ very strong Latency Λ always Jitter Λ very weak Maintenance},

THEN Compensation (CF = 0.90).

According to Table 5.1 and Table 5.2 in this chapter, the rule R_n can be written as follows:

R_1: IF {(Performance, 0.00), (Operation, 0.00), (Availability, 0.05), (Latency, 0.97), (Jitter, 1.00), (Maintenance, 0.05)},

THEN Compensation (CF= 0.90).

where D_1 = {(Performance, 0.00), (Operation, 0.00), (Availability, 0.05), (Latency, 0.97), (Jitter, 1.00), (Maintenance, 0.05)},

THEN Compensation (CF= 0.90).

Here D_1 is a fuzzy set of the universe U, where U = {Performance, Operation, Availability, Latency, Jitter, Maintenance}.

5.5. Similarity Measures and Degrees of Importance

(4) $$T(x, y) = 1 - |x - y|,$$

where $T(x, y) \in [0, 1]$.

The larger the values of $T(x, y)$, the higher the similarity between x and y.

Let U be the universe of discourse and let A and B the two fuzzy sets of U, i.e., U = {u_1, u_2, ..., u_p},

A = {(u_1, a_1), (u_2, a_2), ..., (u_p, a_p)},

B = {(u_1, b_1), (u_2, b_2), ..., (u_p, b_p)},

where,

$a_i \in [0, 1]$, $b_i \in [0, 1]$, and $1 \leq i \leq p$. By using the vector representation method, A and B can be represented by the vectors $\overline{A}$ and $\overline{B}$ respectively, where

$\overline{A} = \langle a_1, a_2, ..., a_p \rangle$,

$\overline{B} = \langle b_1, b_2, ..., b_p \rangle$.

Assuming that each ν_i in U has a different degree of importance and the importance of ν_i is w_i, where $w_i \in [0, 1]$ and $1 \leq i \leq p$, then the degree of importance of each u_i in U can be described by a weighted vector $\overline{W}$, where

$\overline{W} = \langle w_1, w_2, ..., w_p \rangle$.

The degree of similarity between the fuzzy sets A and B can be measured by the similarity function F, $F(\overline{A}, \overline{B}, \overline{W}) \in [0, 1]$, where

$$F(\overline{A}, \overline{B}, \overline{W}) = \sum_{j=1}^{p} \left[T(a_j, b_j) * \frac{W_j}{\sum_{k=1}^{p} * W_k} \right] \tag{5}$$

The larger the value of $F(\overline{A}, \overline{B}, \overline{W})$, the higher the similarity between the fuzzy sets A and B.

5.6. A Weighted Fuzzy Reasoning Technique

In this section, we present a weighted fuzzy reasoning technique based on the similarity function F which is adapted from Shyi-Ming Chen's paper [36].

Let U be a set of complaints filed based on different information in SLA and V be a set of concluded compensation, where

$U = \{m_1, m_2, ..., m_p\}$,

$V = \{d_1, d_2, ..., d_n\}$.

Assuming that the knowledge base contains the following fuzzy production rule:

R_i: IF D_i THEN d_i $(CF = \mu_i)$,

where $D_i = \{(m_j, t_{ij} \mid t_{ij} \in [0, 1], 1 \leq j \leq p\}$, $\mu_i \in [0, 1]$, and $1 \leq i \leq n$, and assuming that the M is the set of customer's complaints, where $M = \{(m_j, x_j \mid x_j \in [0, 1], 1 \leq j \leq p\}$. D_i and M are the fuzzy sets of U, where $U = \{m_1, m_2, ..., m_p\}$. By using vector representation method, D_i and M can be represented by the vectors $\overline{D_i}$ and $\overline{M}$, respectively, where

$\overline{D_i} = \langle t_{i1}, t_{i2}, ..., t_{ip} \rangle$,

$\overline{M} = \langle x_1, x_2, ..., x_p \rangle$.

Let $\overline{W_i}$ be the weighted vector of the complaints appearing in D_i, where $\overline{W_i} = \langle w_{i1}, w_{i2}, ..., w_{ip} \rangle$. The weights are used to emphasize the importance of individual element in the similarity calculation as the elements inside the set have different characteristics and features. By applying above equation, we get [36]:

$$F(\overline{M}, \overline{D_i}, \overline{W_i}) = \sum_{j=1}^{p} \left[T(x_j, t_{ij}) * \frac{W_{ij}}{\sum_{k=1}^{p} * W_{ik}} \right], \tag{6}$$

where,

$F(\overline{M}, \overline{D_i}, \overline{W_i}) \in [0, 1]$. The larger the value of $F(\overline{M}, \overline{D_i}, \overline{W_i})$, the higher the similarity between M and D_i.

Let λ be a threshold value. If $F(\overline{M}, \overline{D_i}, \overline{W_i}) \geq \lambda$, then the rule R_i can be triggered. The threshold helps to prioritize the rules in the situations where multiple rules are triggered simultaneously. This indicates that the customer might get the compensation d_i with the degree of certainty of about c_i, where $c_i = F(\overline{M}, \overline{D_i}, \overline{W_i}) * \mu_i$ and $c_i \in [0, 1]$. The larger the value of c_i, the higher the possibility that the customer might get the compensation d_i. If $F(\overline{M}, \overline{D_i}, \overline{W_i}) < \lambda$, then the rule R_i cannot be activated and therefore, will be discarded.

In the following, we use an example to illustrate the quantification of interpretation, where the result of any arithmetic operation is represented by 2 digits of significant numbers.

EXAMPLE 5.6. Let U be a set of metrics, V be a set of concluded compensation, and M be a set of customer's complaints, where,

$U = \{m_1, m_2, m_3, m_4, m_5, m_6\}$,

$V = \{d_1, d_2, d_3, d_4, d_5, d_6\}$,

$M = \{(m_1, 0.20), (m_2, 0.50), (m_3, 0.20), (m_4, 0.85), (m_5, 0.95), (m_6, 0.00)\}$

Assume that the threshold value λ is 0.50 (i.e., $\lambda = 0.50$), and the knowledge base of a broadband or an ISP contains the following fuzzy production rules:

R_1: IF $\{(m_1, 0.90), (m_2, 0.70), (m_3, 0.70), (m_4, 0.05), (m_5, 0.00), (m_6, 0.50)\}$
THEN d_1 $(CF = 0.10)$

R_2: IF $\{(m_1, 0.50), (m_2, 0.70), (m_3, 0.85), (m_4, 0.70), (m_5, 0.50), (m_6, 0.50)\}$
THEN d_2 $(CF = 0.40)$

R_3: IF $\{(m_1, 0.50), (m_2, 0.50), (m_3, 0.70), (m_4, 0.70), (m_5, 0.85), (m_6, 0.85)\}$
THEN d_3 $(CF = 0.50)$

R_4: IF $\{(m_1, 0.40), (m_2, 0.50), (m_3, 0.70), (m_4, 0.85), (m_5, 0.70), (m_6, 0.40)\}$
THEN d_4 $(CF = 0.50)$

R_5: IF $\{(m_1, 0.20), (m_2, 0.40), (m_3, 0.60), (m_4, 0.80), (m_5, 0.85), (m_6, 0.40)\}$
THEN d_5 $(CF = 0.80)$

R_6: IF $\{(m_1, 0.05), (m_2, 0.05), (m_3, 0.40), (m_4, 0.85), (m_5, 0.85), (m_6, 0.20)\}$
THEN d_6 $(CF = 1.00)$

$\mu_1 = 0.10$, $\mu_2 = 0.40$, $\mu_3 = 0.50$, $\mu_4 = 0.50$, $\mu_5 = 0.80$, $\mu_6 = 1.00$

$D_1 = \{(m_1, 0.90)\}, \{(m_2, 0.70)\}, \{(m_3, 0.70)\}, \{(m_4, 0.05)\}, \{(m_6, 0.00)\}, \{(m_6, 0.50)\},$
$D_2 = \{(m_1, 0.50)\}, \{(m_2, 0.70)\}, \{(m_3, 0.85)\}, \{(m_4, 0.70)\}, \{(m_6, 0.50)\}, \{(m_6, 0.50)\},$
$D_3 = \{(m_1, 0.50)\}, \{(m_2, 0.50)\}, \{(m_3, 0.70)\}, \{(m_4, 0.70)\}, \{(m_6, 0.85)\}, \{(m_6, 0.85)\},$
$D_4 = \{(m_1, 0.40)\}, \{(m_2, 0.50)\}, \{(m_3, 0.70)\}, \{(m_4, 0.85)\}, \{(m_6, 0.70)\}, \{(m_6, 0.40)\},$
$D_5 = \{(m_1, 0.20)\}, \{(m_2, 0.40)\}, \{(m_3, 0.60)\}, \{(m_4, 0.80)\}, \{(m_6, 0.85)\}, \{(m_6, 0.40)\},$
$D_6 = \{(m_1, 0.05)\}, \{(m_2, 0.05)\}, \{(m_3, 0.40)\}, \{(m_4, 0.85)\}, \{(m_6, 0.85)\}, \{(m_6, 0.20)\}$

Based on the vector representation method, M, D_1, D_2, D_3, D_4, D_5, and D_6 can be represented by the vectors $\overline{M}$, $\overline{D_1}$, $\overline{D_2}$, $\overline{D_3}$, $\overline{D_4}$, $\overline{D_5}$, and $\overline{D_6}$ respectively, where

$\overline{M} = \langle 0.20, 0.50, 0.20, 0.85, 0.95, 0.00 \rangle$
$\overline{D_1} = \langle 0.90, 0.70, 0.70, 0.05, 0.00, 0.50 \rangle$
$\overline{D_2} = \langle 0.50, 0.70, 0.85, 0.70, 0.50, 0.50 \rangle$

$\overline{D_3} = \langle 0.50, 0.50, 0.70, 0.70, 0.85, 0.85 \rangle$

$\overline{D_4} = \langle 0.40, 0.50, 0.70, 0.85, 0.70, 0.40 \rangle$

$\overline{D_5} = \langle 0.20, 0.40, 0.60, 0.80, 0.85, 0.40 \rangle$

$\overline{D_6} = \langle 0.05, 0.05, 0.40, 0.85, 0.85, 0.20 \rangle$

Assuming that the weighted vectors of D_1, D_2, D_3, D_4, D_5, and D_6 are $\overline{W_1}$, $\overline{W_2}$, $\overline{W_3}$, $\overline{W_4}$, $\overline{W_5}$, and $\overline{W_6}$ respectively, where

$\overline{W_1} = \langle 0.00, 1.00, 0.10, 0.00, 0.20, 0.40 \rangle$

$\overline{W_2} = \langle 0.30, 0.00, 0.20, 1.00, 0.00, 1.00 \rangle$

$\overline{W_3} = \langle 1.00, 0.00, 0.00, 0.30, 0.00, 0.10 \rangle$

$\overline{W_4} = \langle 0.00, 0.20, 0.50, 0.10, 0.10, 0.00 \rangle$

$\overline{W_5} = \langle 0.50, 0.00, 0.40, 1.00, 0.00, 0.50 \rangle$

$\overline{W_6} = \langle 0.00, 0.30, 1.00, 1.00, 1.00, 1.00 \rangle$

when i = 1,

$$Q = M \cap D_1$$

$$= \{(m_1, 0.20), (m_2, 0.50), (m_3, 0.20), (m_4, 0.05), (m_5, 0.00), (m_6, 0.00)\}; \tag{7}$$

$$T = \{m_1, m_2, m_3, m_4\}.$$

Because $T \neq \varnothing$, we get $y_1 = F(\overline{M}, \overline{D_1}, \overline{W_1}) = 0.62$.

Since $0.62 > \lambda$, we obtain $c_1 = 0.62 * 0.10 \simeq 0.06$.

From Table 5.1, we can see that the corresponding certainty level of c_1 is "*hardly certain*".

when i = 2,

$$Q = M \cap D_2$$

$$= \{(m_1, 0.20), (m_2, 0.50), (m_3, 0.20), (m_4, 0.70), (m_5, 0.50)\}; \tag{8}$$

$$T = \{m_1, m_2, m_3, m_4, m_5\}.$$

Because $T \neq \varnothing$, we get $y_2 = F(\overline{M}, \overline{D_2}, \overline{W_2}) = 0.65$.

Since $0.65 > \lambda$, we obtain $c_2 = 0.65 * 0.40 \simeq 0.26$.

From Table 5.1, we can see that the corresponding certainty level of c_2 is "*very little certain*".

when i = 3,

$$\begin{aligned} Q &= M \cap D_3 \\ &= \{(m_1, 0.20), (m_2, 0.50), (m_3, 0.20), (m_4, 0.70), (m_5, 0.85)\}; \\ T &= \{m_1, m_2, m_3, m_4, m_5\}. \end{aligned} \tag{9}$$

Because $T \neq \varnothing$, we get $y_3 = F(\overline{M}, \overline{D_3}, \overline{W_3}) = 0.40$.

Since $0.40 < \lambda$, the rule R_3 cannot be activated and therefore, will be discarded.

when i = 4,

$$\begin{aligned} Q &= M \cap D_4 \\ &= \{(m_1, 0.20), (m_2, 0.50), (m_3, 0.20), (m_4, 0.85), (m_5, 0.70)\}; \\ T &= \{m_1, m_2, m_3, m_4, m_5\}. \end{aligned} \tag{10}$$

Because $T \neq \varnothing$, we get $y_4 = F(\overline{M}, \overline{D_4}, \overline{W_4}) = 1.44$.

Since $1.44 > \lambda$, we obtain $c_4 = 1.44 * 0.50 \simeq 0.72$.

From Table 5.1, we can see that the corresponding certainty level of c_4 is "*quite certain*".

when i = 5,

$$\begin{aligned} Q &= M \cap D_5 \\ &= \{(m_1, 0.20), (m_2, 0.40), (m_3, 0.20), (m_4, 0.80), (m_5, 0.85)\}; \\ T &= \{m_1, m_2, m_3, m_4, m_5\}. \end{aligned} \tag{11}$$

Because $T \neq \varnothing$, we get $y_5 = F(\overline{M}, \overline{D_5}, \overline{W_5}) = 0.82$.

Since $0.82 > \lambda$, we obtain $c_5 = 0.82 * 0.80 \simeq 0.66$.

From Table 5.1, we can see that the corresponding certainty level of c_5 is "*quite certain*".

$$
\begin{aligned}
&\textit{when } i = 6,\\
&\quad Q = M \cap D_6\\
&\quad = \{(m_1, 0.05), (m_2, 0.05), (m_3, 0.20), (m_4, 0.85), (m_5, 0.85), (m_6, 0.00)\};\\
&\quad T = \{m_1, m_2, m_3, m_4, m_5\}.
\end{aligned} \tag{12}
$$

Because $T \neq \varnothing$, we get $y_6 = F(\overline{M}, \overline{D_6}, \overline{W_6}) = 0.85$.

Since $0.85 > \lambda$, we obtain $c_6 = 0.85 * 1.00 \simeq 0.85$.

From Table 5.1, we can see that the corresponding certainty level of c_6 is "*pretty certain*".

Thus, we can obtain the following results:

(1) The customer might get the compensation d_1 with the degree of certainty of about 0.06 (hardly certain)

(2) The customer might get the compensation d_2 with the degree of certainty of about 0.26 (very little certain)

(3) The customer might get the compensation d_4 with the degree of certainty of about 0.72 (quite certain)

(4) The customer might get the compensation d_5 with the degree of certainty of about 0.66 (quite certain)

(5) The customer might get the compensation d_6 with the degree of certainty of about 0.85 (pretty certain)

5.7. Conclusion

In this chapter, a weighted fuzzy reasoning algorithm for handling consumers' claims on compensation was presented that was adapted from S.M. Chen's algorithm used for medical diagnosis [36]. This technique is flexible as it is able to take into account several sets of complaints filed based on different information in an SLA with different degrees of importance and weight. This algorithm efficiently handles the problem of vagueness in an

SLA and the communication gap due to different interpretations between a company and a customer. With a strong knowledge base that includes fuzzy quantifiers, certainty levels, and production rules, we can efficiently evaluate a customer's complaint and, with the help of approximate reasoning, can derive the customer's complaints of their complaint with a specific and crisp degree of certainty.

CHAPTER 6

DEFUZZIFICATION OF TRADITIONAL SERVICE-LEVEL AGREEMENT (SLA) INTO SMART CONTRACT[1]

6.1. Introduction

In our everyday life, when a customer makes a complaint against the company when the services that they are subscribed to are unsatisfactory, that is against the service-level agreement (SLA) [79], the complaint is made in natural language. For example, *"The service is slow and has been really bad for over a month now."* is the kind of complaint made by the customer to the company that has plenty of vagueness in the statement. The company takes advantage of the customer's lack of legal and contractual knowledge and tries to escape from making the compensation to the customer [71], [77], [39]. And even if the complaints are heard, it takes a long time for the customers to claim their compensation and get back their refund, which results in uninvited wastage of extra time and money just to get the compensations back [112], [140], [60], [124].

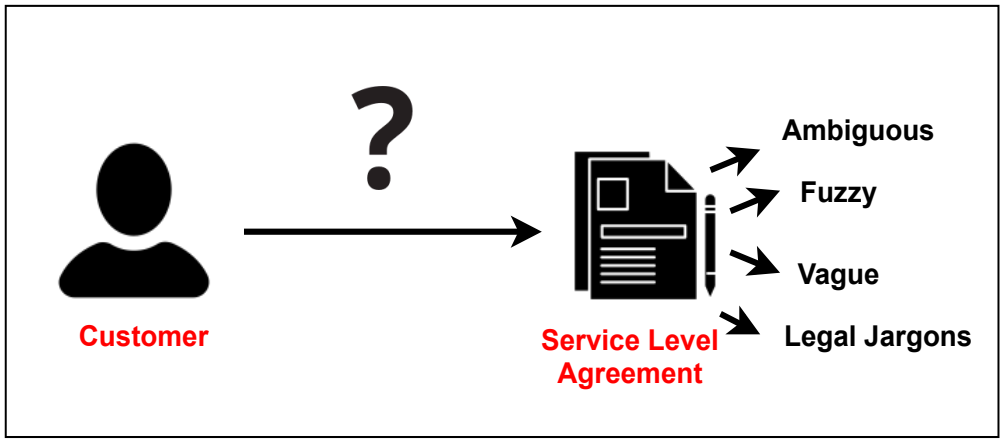

FIGURE 6.1. A layperson does not understand the ambiguity, fuzziness, vagueness, and legal jargon present in the legal contract or service-level agreement.

[1]This chapter is presented in its entirety from K. Upadhyay, R. Dantu, Y. He, A. Salau and S. Badruddoja, "Make Consumers Happy by Defuzzifying the Service-Level Agreements," 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), Atlanta, GA, USA, 2021, pp. 98-105, doi: 10.1109/TPSISA52974.2021.00011. © 2021 IEEE. Reprinted, with permission.

Ambiguity and vagueness are the phenomenons that many have tried to study in natural language. We find vagueness, fuzziness, and legal jargon abundantly that is beyond our comprehension in legal contracts [20], [141], [29], [110] and service-level agreements (SLAs). Fuzzy logic is an approach to computing something that is based on degrees of truth rather than the Boolean true or false (1 or 0). Natural language has many gray areas, and nothing can always be classified either as 1 or as 0. We have used fuzzy logic because it can model the semantics of linguistic expressions [188]. After all, fuzzy sets can capture their innate vagueness. Fuzzy logic is also much cheaper and quicker at the same time when implemented inside the smart contract [159] for the blockchain compared to machine learning due to the simplicity of fuzzy logic's [188], [86] rule-based system and an inference engine that makes the smart contact not only smart but also intelligent.

Despite the fact that there has been substantial research going on for the smart contracts in the present day [74], [3], [158], [150], [169], [42], [125], the study specifically on the vagueness in legal contracts and translation of the legal contracts to smart legal contracts considering vagueness in legal contracts as the main factor has not been exhaustive as mentioned in Chapter 2.

6.2. Contributions

The main focus of this chapter is the Ethereum-based smart contract that incorporates fuzzy logic, which is intelligent enough to handle the issues of linguistic vagueness present in the legal contracts and SLAs that will potentially create multiple interpretations. The main contributions of this chapter [166] are as follows:

- We take a real-life SLA from a popular telecommunication vendor, Spectrum's SLA [133], and find ambiguities and vagueness in it.
- We manually summarize the whole vague SLA into an IF and ELSE condition that would be the basis for our fuzzy logic-based smart contract.
- We incorporate Mamdani's Fuzzy Inference System [114] inside our Ethereum-based smart contract.

- We create three different smart contracts with the same architecture but with different numbers of linguistic descriptors and membership functions and perform several experiments for evaluation and analysis.
- Finally, we conclude that using fuzzy logic inside an Ethereum-based smart contract would be a novel and convenient method to handle the uncertainties and vagueness found in the clauses of the SLA that would also result in faster settlement in claiming compensation by the dissatisfied customers.

6.3. Experimental Setup

The tools and materials that we have used for this project are listed below:
i) Ropsten Test Network, ii) Solidity Programming Language, iii) Remix Web IDE, iv) Metamask, v) Node.js, vi) Truffle, vii) Ganache-CLI, viii) Web3, ix) HD Wallet, x) Google Chrome in Incognito Mode, xi) Python 3, Anaconda and Jupyter Notebook xii) Skfuzzy [129]

6.4. Methodology

When a customer is not satisfied with the services provided by their company, as mentioned in the company's service-level agreement (SLA), due to the lack of knowledge of legal jargon and vague words and phrases, it would be difficult for any customer to understand the SLA clearly and claim their compensation. A customer has to go through a lot of hassles even if they would have understood the vague legal words in the SLA. Hence, in this work, we have selected a real-life SLA from a popular telecommunication vendor, Spectrum Internet from Charter Communications, and studied the vagueness and ambiguities found in the SLA. We read the whole SLA and found out that there were not any metrics properly given for the customers discussing the performance and operation of the company. Furthermore, the compensation that was provided was absolutely not in favor of the customers who were experiencing the worst internet service. Since the whole SLA was vague and the metrics were not properly set out for the customers, we concentrated and summarized the whole SLA into one general fuzzy rule. The rule is: "*If the Performance*

and Operation are bad, then Compensation should be high." Here, *Performance* is the title of Clause 3 and *Operation* is the title of Clause 4. Since the basis for the calculation of compensation is the performance and operation of the company, as the SLA states, we have assumed our two inputs are Performance and Operation, and our output is Compensation. This fuzzy rule would be the basis for this work, where we create an Ethereum-based smart contract that has fuzzy logic implementation inside. We created a smart contract that has a fuzzy inference system in it so that the smart contract can actually be smart and can decide by itself the total compensation amount to be sent back to the customer's account based on the ratings provided by the customers. There is a possibility that customers can provide fake ratings which are very low to get a higher compensation amount [76]. However, that is the concern of this work, and since everything should be validated and everybody should come to a consensus in blockchain so, cheating by providing low ratings just to get higher compensation even if a customer is getting good service is not possible [185]. Our main focus in this work is the Ethereum-based smart contract itself that is smart and intelligent, which can understand and decode the natural human language and hedges by quantifying the linguistic variables and providing us a crisp value of compensation.

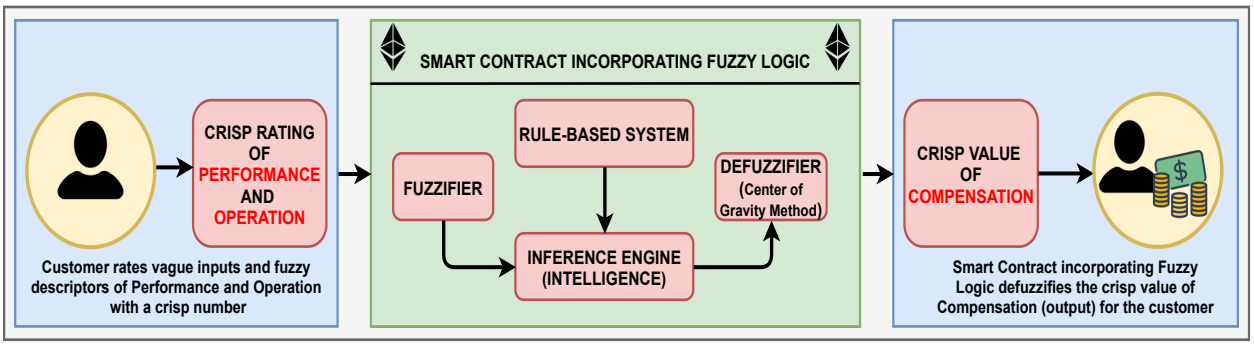

FIGURE 6.2. Our model architecture consists of three main phases where a dissatisfied user who wants to claim compensation, provides crisp ratings of the company to the fuzzy logic-based smart, which fuzzifies the inputs into linguistic variables for the generation and inference of rules, and finally defuzzifies the aggregated fuzzy output into the crisp value of compensation for the customer.

Our methodology comprises a smart contract that incorporates a fuzzy logic mechanism that has four major components and their respective functions in it [14]. They are *Fuzzifier, Rule-based System, Inference Engine, and Defuzzifier* as shown in Fig. 6.2. In our methodology, we have three main phases, which are explained below:

6.4.1. Inputs

First, the customer provides crisp ratings for the vague inputs, Performance, and Operation to the smart contract. This input is measured in percentages. For example, if the customer is highly satisfied with the performance of the company but somewhat satisfied with the operations of the company, they would rate Performance as 90% and Operation as 40% in the smart contract.

6.4.2. Components of Smart Contract

The ratings for two inputs provided by the dissatisfied customer will now be fetched by the smart contract, which performs fuzzy logic operations inside. We have four major components inside the smart contract, as further explained in detail below:

(1) Fuzzifier:

Fuzzifier is a component that is responsible for the process of converting the crisp inputs (ratings) for Performance and Operation provided by the user. The crisp ratings are converted into linguistic variables [57] and are assigned with the membership values. The source of the membership values is either the domain expert, intuition, or statistical analysis. In this work, the source of assignment of membership values is both domain expert and intuition. We have developed three different smart contracts called *SC 1, SC 2* and *SC 3* that have the same architecture but a different number of linguistic descriptors and hence a different number of membership values for each linguistic descriptor. In our SC 1, we have the least number of descriptors for inputs, i.e., three. We increase the number of descriptors for inputs to five for SC 2. Finally, we have eight descriptors for both inputs and output in SC 3.

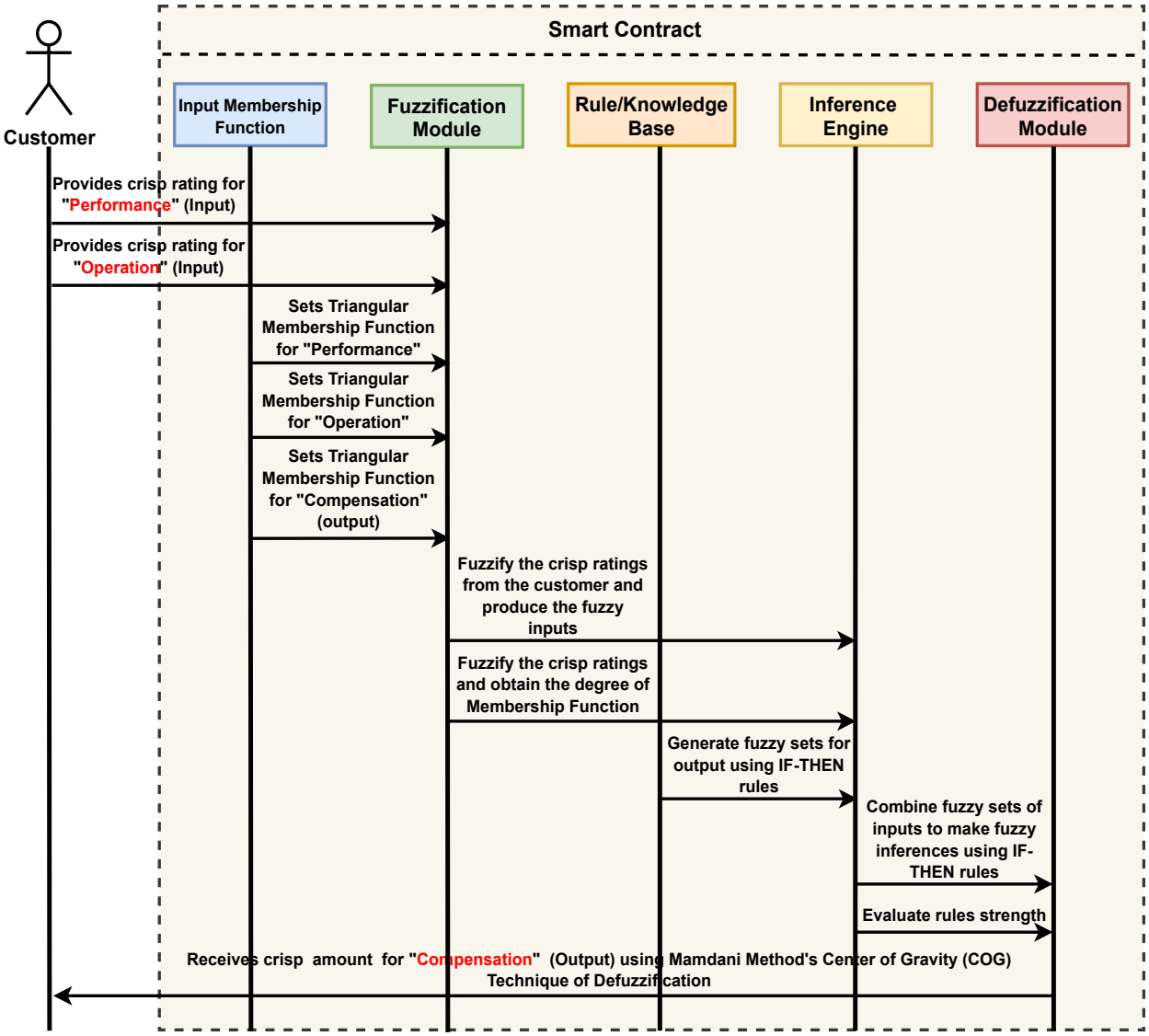

FIGURE 6.3. Implementation of Fuzzy Logic inside Smart contract which uses Triangular Membership Function [139] in order to solve the problem of contractual vagueness by fuzzifying the crisp inputs provided by the customer. With the help of a rule-based system and inference engine, the customer will get the correct amount of compensation or service credits without having to deal with the vagueness and fuzziness present in the SLA

Descriptors are fuzzy linguistic variables that describe the gray areas of the fuzzy inputs. The descriptors for inputs and output for each smart contract are provided below:

- *Smart contract with 3 descriptors (SC 1):*

  This smart contract has only 3 descriptors in its inputs and 5 descriptors in its output. We have also referred to this smart contract as "SC 1" in our figures below, as this was the first smart contract we developed and tested. The descriptors of the inputs and outputs are provided below:

  - Performance: {*poor, good, excellent*}
  - Operation: {*slow, acceptable, rapid*}
  - Compensation: {*very low, low, normal, high, very high*}

- *Smart contract with 5 descriptors (SC 2):*

  Our second smart contract has 5 descriptors in both its inputs and output. The descriptors in this smart contract have been stretched out to 5 for evaluation and further research purposes. We have referred to this smart contract as "SC 2". The descriptors of the inputs and outputs are provided below:

  - Performance: {*very poor, poor, good, very good, excellent*}
  - Operation: {*very slow, slow, acceptable, fast, rapid*}
  - Compensation: {*very low, low, normal, high, very high*}

- *Smart contract with 8 descriptors (SC 3):*

  Finally, our third smart contract has been even further stretched out to 8 descriptors in both inputs and outputs. The reason we also increased the number of descriptors in the output along with the inputs was that we did not want to have fewer descriptors in the output compared to the number of descriptors in the inputs. We have referred to this smart contract as "SC 3". The descriptors of the inputs and outputs for SC 3 are provided below.

  - Performance: {*extremely poor, very poor, poor, satisfactory, good, very good, extremely good, excellent*}
  - Operation: {*extremely slow, very slow, slow, mediocre, acceptable, fast, very fast, rapid*}
  - Compensation: {*extremely low, very low, low, insufficient, normal, high,*

very high, extremely high}

No hard and fast rule says anything about a specific name should be given to a descriptor, or there should be a specific number of descriptors in the inputs and output [157]. We have used the aforementioned descriptors because they are suitable for this research that involves service-level agreements (SLA). Although there are various membership functions [8] to assign membership values after the crisp ratings are converted into linguistic descriptors, in this fuzzification process, we have used triangular membership function. The membership values range from 0 to 1 and are denoted by μ.

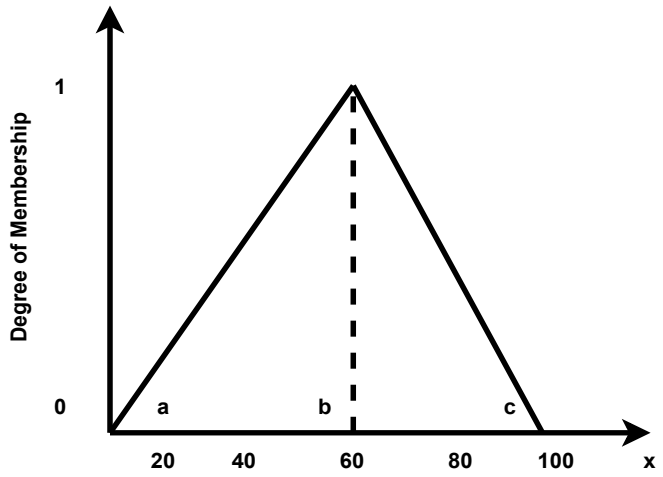

FIGURE 6.4. An example of a triangular membership function

The triangular membership function is defined as [139]:

$$\mu(x,a,b,c) = \begin{cases} 0, & \text{if } x < a \\ (x-a)/(b-a), & \text{if } a <= x <= b \\ (c-x)/(c-b), & \text{if } b < x <= c \\ 0, & \text{if } c < x \end{cases} \tag{13}$$

Where, $\mu(x,a,b,c)$ is the degree of membership of parameters a, b, and c.

For example, as we can see in the sequence diagram of Fig. 6.3, when a customer provides the crisp ratings for Performance and Operation, the Fuzzifier takes those crisp numbers to convert them into the linguistic descriptors we have mentioned earlier. For instance, if we take the case of SC 1, which has only three linguistic descriptors when a customer rates Performance as 40%, this crisp value will be converted into a fuzzy descriptor. Hence, for some people, 40% might be *poor*, and for others, the same rating of 40% might be *good*, which depends on people's experience and interpretation.

(2) Rule-based system:

Once the crisp values are fuzzified into descriptors and membership values are assigned for those corresponding descriptors, we construct fuzzy rules in a rule-based system that has IF, OR, AND, THEN with linguistic descriptors. These rules are very much similar to the rules from the Decision Tree [98]. Each rule has two parts that are antecedent and consequent. Any rule can have multiple antecedents and consequents. For instance, there are three rules formed in the rule-based system after the crisp inputs are fuzzified into descriptors. Here in this instance, the descriptors are *poor* and *slow*. The antecedent is the condition, and the result is the consequent. *IF the performance is "poor" OR operation is "slow"* is the antecedent, and *THEN compensation is "high"* is the consequent. There are n^2 number of rules inside the rule-based system, where n is the number of descriptors in inputs. As we can see in Fig. 6.5, this is the matrix of rules for SC 1 that has 9 rules altogether because of three descriptors for inputs in SC 1. Similarly, our SC 2 had 25 rules altogether because five different descriptors were assigned for each input and output. Likewise, we stored a total of 64 rules in our SC 3 because the inputs and output of SC 3 had eight descriptors in its inputs and output.

(3) Inference engine:

Once the fuzzy rules are created and stored in the rule-based system, we map the fuzzy rules into the membership graphs of all the parameters. This means that

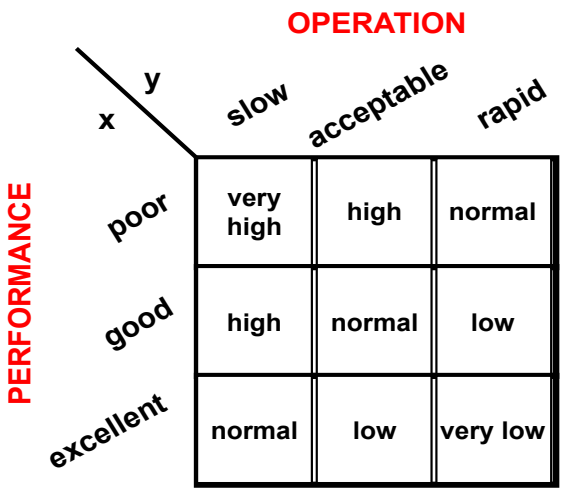

FIGURE 6.5. Matrix of 9 rules for SC 1 as SC 1 just has three descriptors for its first input, Performance, and three descriptors for its second input, Operation.

we map the antecedents of a rule to the consequents of the same rule.

Mapping is performed to all the rules in the rule-based system. Once the matrix is created, then the inference engine decides what the output will become when the crisp inputs are converted to the linguistic variables. For instance, as we can see in the matrix, when Performance is *poor*, and Operation is *slow*, the Compensation is *very high*. Similarly, when the Performance is *excellent*, but Operation is *acceptable*, then Compensation is *low*. Additionally, the Inference Engine also helps to measure the strength of the rules and select for the final phase, defuzzification. The membership values of the antecedents are conjoined together with the intersection operator (finding the MIN or minimum) since they are connected with AND in this work. If the antecedents had been connected with OR, then we would have used the union operator (finding the MAX or maximum). Once the membership values of the antecedents are compared, and the minimum values of each rule are selected, the minimum membership value would be the unit for measuring the strength of the rules.

(4) Defuzzifier:

Defuzzification is the final step of fuzzy logic and hence the final component of our smart contract as well. This component is responsible for making the final decision by selecting the rule that has the highest strength [35]. More importantly, this component is also responsible for finding the crisp value from the output of the aggregated fuzzy set. From Fig. 6.3, we can see that when the strength of rules is evaluated, the defuzzifier/defuzzification module transfers the fuzzy inference results back to the crisp value. This crisp value would be the final output. Although there are various defuzzification techniques, such as the Mean of Max method, Weighted sum method, Lambda-cut method, etc., we have used the Center of gravity (COG) method in this work [65]. The COG is defined as [184]:

$$X^* = \frac{\sum_{i=1}^{n} x_i * A_i}{\sum_{i=1}^{n} A_i} \tag{14}$$

Where, X^* is the crisp output, $\sum_{i=1}^{n}$ is the sum over variable's possible values, x_i is the center of an area, and A_i is the total area of the selected region.

From the sequence diagram in Fig. 6.3, we can also see that the final crisp value for Compensation is produced as output and sent back to the customer.

6.4.3. Output

Finally, the customer who provides the crisp ratings for the two inputs to the fuzzy logic-based smart contract, i.e., Performance and Operation, will get a crisp result back as Compensation. The Compensation is also measured in percentages, just like the inputs. For example, from Fig. 6.6, if SC 1 is implemented, we can see when the customer provides crisp ratings for Performance and Operation as 20% and 30%, respectively if the service is poor, the customer receives the Compensation as 60% of total expenses of his/her subscription to the current service. However, when the customer increases the crisp ratings for Performance and Operation to 50% and 60%, respectively, the customer receives the Compensation as 26.82%. The lower the customer ratings are, the higher the compensation is, and vice-versa. Nevertheless, in the Results section, we still discuss the accuracy of our defuzzification output in all three smart contracts.

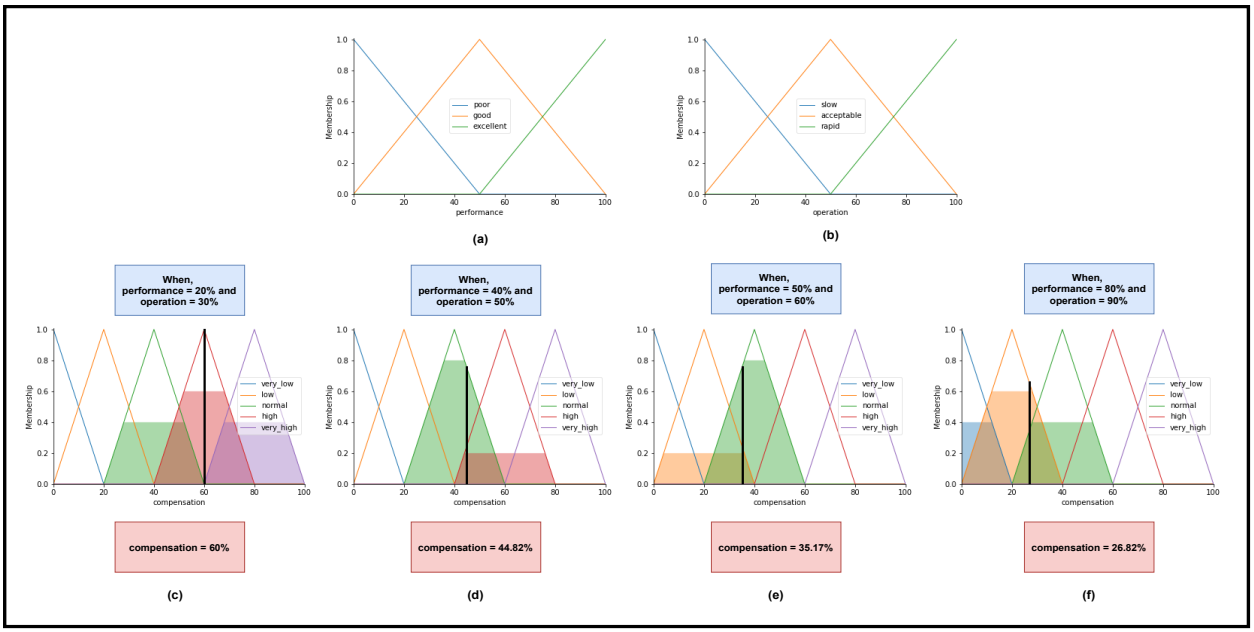

FIGURE 6.6. Performance and Operation are the inputs of the Smart contract incorporating Fuzzy Logic with 3 descriptors for inputs and 5 descriptors for output where the Triangular MF and Center of gravity method is used. Different series of inputs are provided to observe the varying nature of the output, i.e., Compensation.

6.5. Results

We have developed and tested three different types of smart contracts with the same architecture of fuzzy logic that we have discussed above in the Methodology section. The only way these three smart contracts differ is by the number of descriptors and their corresponding membership functions and matrix of rules. Although the employed technique of defuzzification is the same, i.e., the Center of gravity (COG) method, these three different kinds of smart contracts have a different number of descriptors. The reason we developed and tested three different kinds of smart contracts is to successfully evaluate and analyze the performance, accuracy, and impact of the varying number of descriptors in smart contracts when deployed into the Ethereum-based Blockchain. More is explained about these smart contracts and their respective descriptors in detail in the following subsections.

6.5.1. Defuzzification Results from Different Smart Contracts

In our SC 1, we got different results for Compensation when different values for inputs were provided as expected. Since the Compensation will be higher when the customer ratings are lower and will be lower when the customer ratings are higher, the inputs and output have an inverse relationship. As shown in Fig. 6.6 and Fig. 6.7, when the user provides the crisp rating of 20% and 30% in the smart contract for Performance and Operation, respectively, it gives us the output of 60% for Compensation. Similarly, SC 1 gives the Compensation of 45% when the Performance and Operation are 40% and 60%, respectively. When the inputs for Performance and Operation are 50% and 60%, respectively, SC 1 gives us the Compensation of 35%. Finally, when the inputs for Performance and Operation are increased to 80% and 90%, the Compensation reduces to 27%. We can also see in Fig. 6.7 that the compensations calculated by the SC 1 decrease when the ratings for Performance and Operation are increasing.

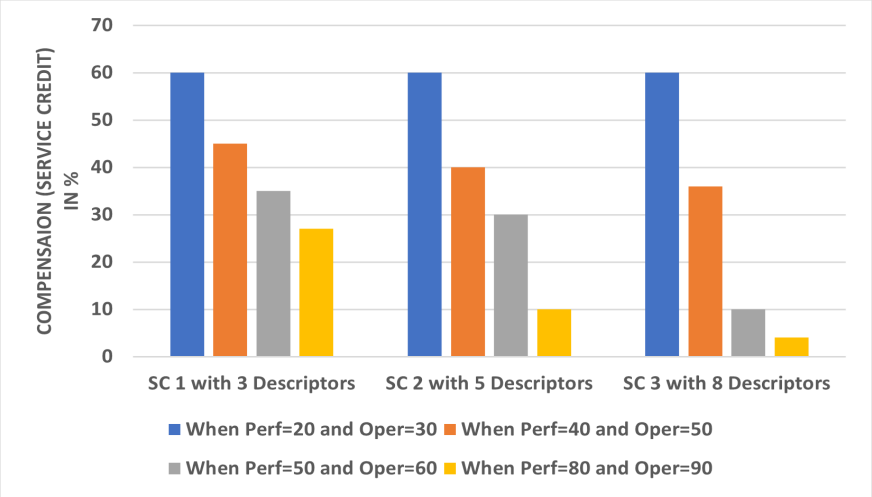

FIGURE 6.7. Defuzzification of the output of three different Smart contracts when different values of inputs are provided.

We provided different values of ratings as inputs again for SC 2. Although SC 2 has more descriptors for Performance and Operation, i.e., five, we can see that the Compensation is the same as 60% when the ratings are 20% and 30% respectively. However, when the ratings for Performance and Operation are increased to 40% and 50%, this time SC 2

gives us the Compensation of 40%. Similarly, when the input ratings are 50% and 60%, respectively, the Compensation is 30%. Finally, when there are high ratings for Performance and Operation, such as 80% and 90%, respectively, SC 2 gives us the Compensation of 10%.

Finally, for SC 3, we provided the same series of values of ratings as inputs again for SC 3 as we did for SC 1 and SC 2. From Fig. 6.7, we can see that when the ratings for Performance and Operation were 20% and 30% respectively, SC 3's output, i.e., Compensation is 60%. We can observe in the bar chart the values for the output are decreasing when provided the same values for inputs as SC 1 and SC 2. Similarly, the Compensation was outputted as 36% when the rating inputs were increased to 40% and 50%, respectively. Likewise, when the customer ratings were increased to 50% and 60%, respectively, the Compensation fell to 10%. Finally, when the customer ratings were at their highest, i.e., 80% and 90%, the Compensation output was just 4%.

From the bar chart, we can see that regardless of the kind of smart contract and the number of descriptors they have, when the two inputs, i.e., Performance and Operation are 20% and 30%, respectively, the final defuzzified crisp value for Compensation is same or at least similar in all three smart contracts, SC 1, SC 2 and SC 3.

6.5.2. Deployment Costs of Different Smart Contracts

We deployed all three smart contracts in Ropsten Testnet. We converted all ETH costs in United States Dollars (USD), and on August 16, 2:09 AM UTC, the conversion rate of 1 ETH was 3,315.44 USD. This data was provided by Morningstar for Currency and Coinbase for Cryptocurrency [11]. SC 1 had the lowest deployment cost, i.e., 14.02 USD. The reason SC 1 had the lowest deployment cost was that it was the lightest program among all smart contracts, as SC 1 only had three descriptors in its inputs, Performance and Operation. As a result of only three descriptors, the number of membership functions for each corresponding descriptor was also lesser. However, SC 2 had five descriptors for both inputs, and as a result, this smart contract had more membership functions assigned for its descriptors. Hence, the deployment cost for SC 2 was higher than SC 1, i.e., 21.24 USD. Finally, the highest deployment cost was for SC 3 because it had eight descriptors for inputs,

Performance, and Operation, as well as for the output, Compensation. The deployment cost for SC 3 was 30.11 USD.

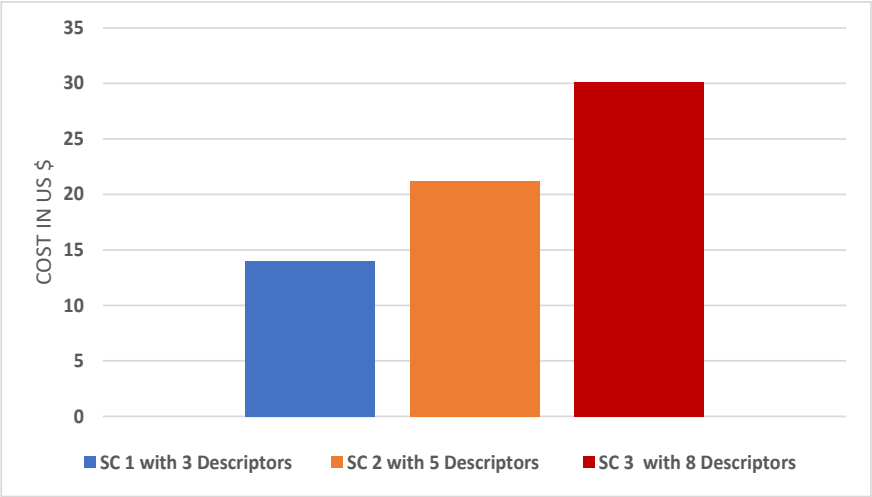

FIGURE 6.8. Deployment costs of SC 1, SC 2, and SC 3 in Ropsten Ethereum Testnet in USD.

The deployment cost of a smart contract depends on its size. Therefore, the larger the smart contract, the higher the deployment cost is. In this case, the size of the smart contract was affected by the number of descriptors and their corresponding membership functions in the smart contract. Additionally, we can also observe that we have a common ratio of 1.5 in this geometric series of deployment costs among the three smart contracts. The deployment cost of SC 2 is 1.5 times higher than the deployment cost of SC 1, and the same case for SC 3 and SC 2 as well. The reason we see this almost precise ratio between the deployment cost is the number of descriptors chosen for inputs in each smart contract, i,e, three, five, and eight.

6.5.3. Transaction Costs of Major Functions Used in Different Smart Contracts

We discuss the TXN costs incurred by the four major functions used in the smart contracts below.

- *Function for Performance:*

  This function is responsible for taking the crisp input from the customer for rating

the Performance of the company and then fuzzifying the crisp input using its membership functions depending on how many descriptors it has. In SC 1, the TXN cost for Performance was 1.29 USD. The TXN cost for Performance increases to 2.13 USD in SC 2 and increases further to 2.64 USD in SC 3. The reason TXN cost is getting higher and higher is due to the increasing number of descriptors and their corresponding membership functions in smart contracts.

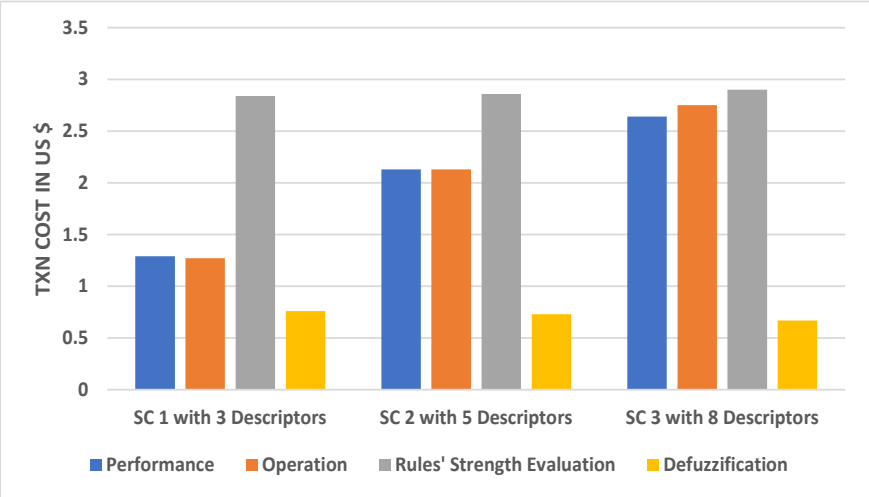

FIGURE 6.9. TXN costs of major functions used in SC 1, SC 2, and SC 3 in USD.

- *Function for Operation:*

  This is a similar function to Performance as Operation is our second input. Likewise, this function is responsible for taking the crisp input from the customer for rating the Operation of the company and then fuzzifying the crisp input using its membership functions depending on how many descriptors it has. Hence, the TXN costs in each smart contract for Operation is approximately the same as Performance, as we can see in Fig. 6.9. The TXN cost for Operation in SC 1, SC 2, and SC 3 were 1.27 USD, 2.13 USD, and 2.75 USD, respectively, in an increasing fashion.

- *Function for Rules' Strength Evaluation:*

  Depending on the crisp ratings given by the customer, this function checks and

selects the triggered rules in the rule-based system by calculating the membership values of each descriptor. Then, in the inference engine, when two antecedents are joined together using a conjunctional operator, i.e., AND operator/MIN operator/Intersection operator, it compares between two membership values each time and finds the minimum value to evaluate the strength of all selected rules. Hence, because of this function's complexity, the TXN cost incurred is the highest, as shown in Fig. 6.9. The TXN cost incurred by the functions that measure the strength of the rules is approximately the same in all three smart contracts. The TXN costs for SC 1, SC 2, and SC3 are 2.84 USD, 2.86 USD, and 2.90 USD, respectively. Even though the TXN cost for this function is highest in SC 3, higher in SC 2, and lowest in SC 1, there is not much significant difference in the TXN costs regardless of being from different smart contracts. The reason the TXN costs are almost similar in this case is that the number of rules this function checks to measure their strength is only four. Only four rules are selected for evaluation of their strength because there are only two different inputs. Since we only have two inputs, only four rules are triggered and then selected from n^2, where is the number of inputs. Hence, the level of complexity of this function, regardless of having a different number of descriptors in a different smart contract, is the same.

- *Function for Defuzzification:*

  This function is responsible for finding the crisp output from the aggregated fuzzy set. The defuzzification technique that we have used for this work is the Center of gravity (COG) method, as mentioned in the methodology section. The TXN cost of SC 1, SC 2, and SC 3 are 0.76 USD, 0.73 USD, and 0.67 USD, respectively. The TXN cost incurred by this function is also almost exactly similar because of the usage and implementation of the same method across all three smart contracts.

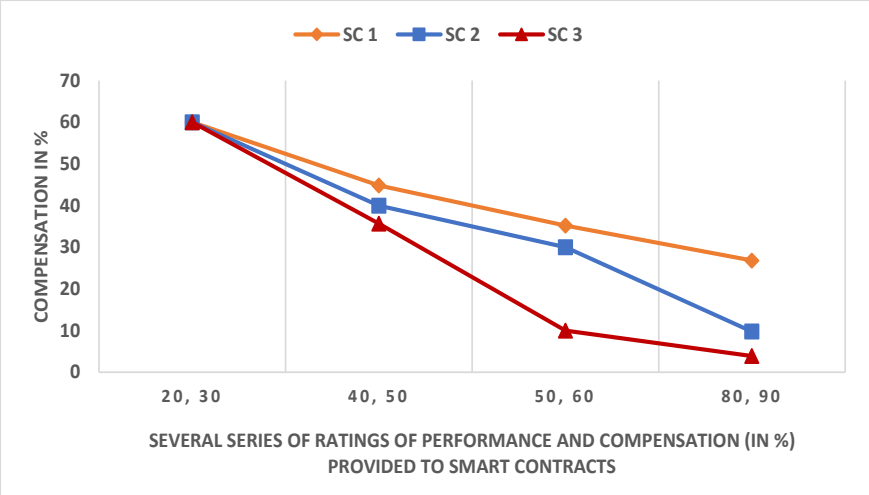

FIGURE 6.10. SC 3 proves to be most accurate and effective as it provides the most realistic and accurate output, whereas SC 1 with the least realistic output.

6.5.4. Effectiveness of the Smart Contract

From our observations in the Fig. 6.10, we concluded that SC 3, with the highest number of descriptors, has the highest accuracy compared to SC 1 and SC 2 because when the Performance and Operation are 80% and 90%, respectively, the value of Compensation in SC 1 is 26.82% which is extremely high and unusual in real life [40]. However, the output from SC 3 has the most accurate and realistic values of all defuzzified Compensation values compared to SC 1 and SC 2.

6.6. Conclusion

We introduced a novel idea on how we can translate a vague legal contract by using fuzzy logic inside the smart contract that would be smart and intelligent enough to consider various human interpretations by taking several linguistic variables and descriptors into account. No matter how popular a vendor is, their SLAs can still be incomplete and vague, which puts a customer into a myriad of confusion and trouble. In this chapter, we presented a fresh solution to an existing problem of vagueness in legal contracts by taking a real-world legal contract and using a cheaper and faster approach, i.e., fuzzy logic, to make the Ethereum-based smart contract smart and intelligent to decide the output based on several

sets of different inputs. We also created three different smart contracts that employ the same logic and architecture but have different sets of linguistic variables to evaluate the behavior, cost, and accuracy of each smart contract. The main purpose of this chapter is to study the gray areas of natural language that create the fuzziness and vagueness in legal contracts and how an Ethereum-based smart contract can be made even smarter and more intelligent to easily handle this problem of vagueness and multiple contract interpretations.

CHAPTER 7

SECURITY ANALYSIS AND MULTIMODAL AUDITING OF SMART CONTRACTS FOR DIGITAL ECONOMY[1]

7.1. Introduction

As pointed out in Chapter 1, we have already stepped into the fourth industrial revolution, where the digital economy is evolving at its best. Web3 has already taken over [70], and the technologies that are employed in today's digital economy are state-of-the-art and sophisticated such as blockchain and smart contracts, artificial intelligence, edge computing, internet of things, extended reality, 3D reconstruction, and so on. Due to this, the economy has gotten more and more customer-centric. In addition, there is enhanced social interaction, expanded access and inclusivity, greater economic opportunities, rich and immersive experiences, and more collaboration and innovation. One such revolutionizing example of the digital and virtual economy is metaverse.

The term "Metaverse" was first introduced by American science fiction author Neal Stephenson in his 1992 novel, Snow Crash [156]. Today, the metaverse is defined as a simulated digital environment that employs extended realities, blockchain, and artificial intelligence, along with ideas of social media, to create the scope and sphere for rich user interaction that mimics the real world [54].

Nonetheless, as the technology and hence, the industry gets increasingly high-end and sophisticated, the challenges it invites are also enormous. Especially in the scope of cybersecurity, the digital economy is even more vulnerable to various forms of cyber attacks such as ransomware, phishing, hacking, malware, etc., which can cause information theft, identity theft, cryptocurrency theft, and several other financial losses and data breaches. As the metaverse is a modern form of the digital economy, it has certainly been "living up to

[1]Portions of this chapter are reproduced from K. Upadhyay, R. Dantu, Y. He, S. Badruddoja and A. Salau, "Auditing Metaverse Requires Multimodal Deep Learning," 2022 IEEE 4th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications (TPS-ISA), Atlanta, GA, USA, 2022, pp. 39-46, doi: 10.1109/TPS-ISA56441.2022.00015.

the hype" of its consumers. Nevertheless, as it also has the potential to invite cybersecurity risks, modern problems will definitely need modern solutions, which we discuss in the subsequent sections. In this chapter, we have used the terms "digital economy" and "metaverse" interchangeably.

7.2. Contributions

- We outline the major technologies employed in the digital economy, such as metaverse, along with the current trends, and argue why a decentralized digital economy is better than a centralized digital economy.
- We present unprecedented security risks and vulnerabilities that modern digital economies such as the metaverse can invite.
- We provide a novel conceptual model with a secure multimodal approach for countermeasures that can tackle the security risks in the digital economy.
- We discuss the advantages of a manual secure multimodal approach for remediation strategies in detail.

7.3. Technologies Employed in Digital Economy

- Blockchain: In simple terms, the blockchain is a *distributed ledger technology* that comprises a continuously growing sequence of blocks linked together by cryptographic hashes. The blockchain reflects an immutable history of the states and sets of transactions throughout the life of the system. The distributed ledger is maintained by the peers running a consensus protocol in the system. Also, due to the decentralized, distributed, inherent, immutable, and secure nature of the blockchain, it offers the digital economy an effective way of securing its users' data and digital contents [68].
- Extended reality: Extended reality (XR) enables users in the digital economy to visualize and actively interact with 3D contents [180]. With XR, which comprises *Augmented reality* (AR), *Virtual reality* (VR), and *Mixed reality* (MR), can help to improve existing techniques or even facilitate novel approaches in medicine/healthcare,

education, sports, etc. with the co-existence and interoperability of the virtual world and the physical world [41].

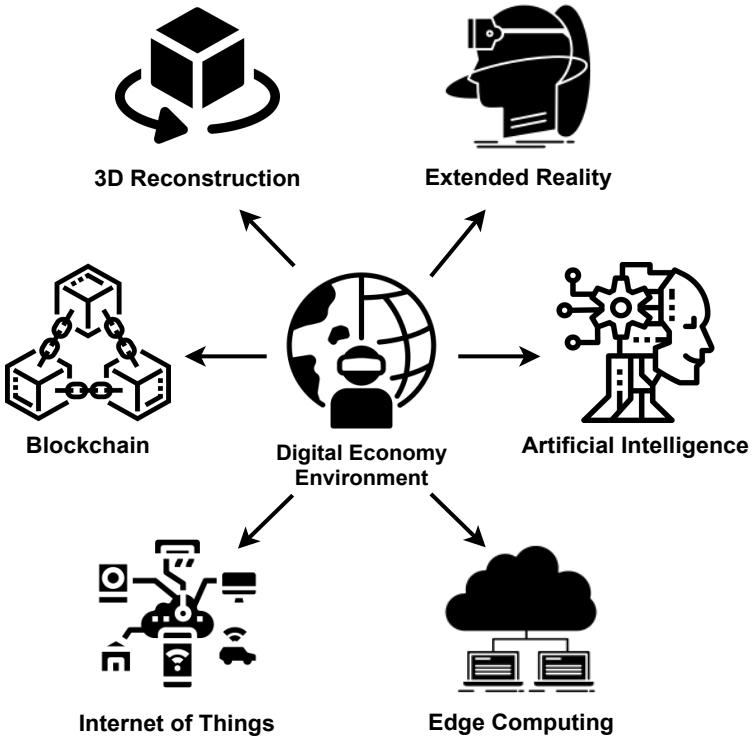

FIGURE 7.1. Digital Economy is composed of various advanced technologies such as Blockchain, 3D Reconstruction, Artificial intelligence, Extended Reality, IoT, and Edge Computing.

- Artificial intelligence: AI engines can serve various purposes within the digital economy, such as analyzing 3D scans and 2D images for the creation of user avatars, lifelike conversations and interactions with users, generating outputs, and even improving the virtual world based on learning experiences from users and so on [18].
- Internet of things: IoT can enable fast communication between the different components of the digital economy, i.e., seamless connectivity of the virtual world and the physical world devices. For example, data from IoT can improve context and awareness of the physical devices for extended reality applications with real-time communication between the virtual and physical worlds. There are numerous benefits of IoT [109] for the digital economy.

- 3D reconstruction: To realize an immersive virtual world that is similar in construction to the physical world in the digital economy, 3D reconstruction is fundamental. 3D models of real-life objects can be made from realistic images taken with 3D cameras, creating a digital representation of the real world for the virtual world. This digital representation of the real world in the virtual world is called Digital Twins [174].
- Edge computing: Edge computing is crucial in the development of the digital economy as it helps to reduce latency in users' interactions, activities, and engagements in the virtual world. It will facilitate the capabilities of extended reality technology, blockchain security, connectivity, and computing, among others [53].

7.4. Experiences in Digital Economy

- Social networking: Digital economies such as metaverse can improve the connectivity and experiences of social media users [55]. Naturally, humans are social beings, communicating and interacting with one another, whether online or offline. The metaverse will enable a smooth integration of the offline and online (social networks) lives into one, and because users can experience multiple virtual worlds in the metaverse, they will be able to enjoy and experience social life that may have been beyond their reach in real life.
- Gaming: Gaming is one of the fascinating experiences in the digital economy. With the current immersive experience that users already derive from games such as Minecraft, Call of Duty, Animal Crossing, etc., in the metaverse, the experience can only get better as users will be able to have a more realistic combination of the virtual and physical worlds [54].
- Trading: The digital economy opens a new world of trading for internet users, where they are allowed to buy and sell their digital assets. There are already platforms that provide access to such a trade, e.g., Decentraland, Sandbox, and many more. We will discuss more such platforms in Sec. 7.5.
- Events: With the metaverse as a virtual form of smart city, users can hold events

virtually, saving them the cost of physical and in-person meetings, especially in times of pandemic as we currently experience [55]. For instance, researchers can virtually hold a conference, with a teacher organizing a virtual classroom with the students experiencing the same or even better learning process.

- Remote working: The digital economy is bringing to reality novel techniques in medicine and health care, education, hospitality, and tourism, among others. For instance, in healthcare, the operations of *Doctors without Borders* are greatly improved as a surgeon can remotely participate in a medical procedure on a patient without having to be physically present with the patient [160].
- Entertainment: Users can enjoy not just the gaming experience of the metaverse, but even much more entertainment, like watching movies in an immersive 3D virtual world with VR/AR tools, attending live concerts, and many more [88].

7.5. Current Trends in Modern Digital Economy

- Decentraland: Decentraland [137] is a virtual reality platform that allows users to buy, sell, and manage their virtual property, i.e., real estate. In this digital economy, the native currency is known as MANA [7]. Users can create and then develop their own world as they desire and tour this metaverse from their phones, computers, or VR headsets. The cryptocurrency MANA uses an Ethereum-based blockchain as its platform but is expensive due to the high gas fee that is required for operating the blockchain. Decentraland Metaverse has been catching the attention of many users. As soon as Facebook changed its name to Meta, there was a staggering 400% rise in the price of MANA[2].
- Axie Infinity: Axie Infinity [52] is probably the most popular digital economy in the present day as it already has more than a quarter of a million users daily. The users, or players, have to own a native token called AXS so that they can get some stake in the ownership and perform some operations in the game [7]. In this

[2]https://markets.businessinsider.com/news/currencies/mana-decentraland-altcoin-metaverse-facebook-rebranding-land-ethereum-crypto-2021-11

metaverse, users or players create their own kingdoms and hunt for rare treasures. This metaverse also runs on an Ethereum-based blockchain where players compete to receive a native NFT called "Axies", which can cost anywhere from USD 4 to USD 100,000, depending on how rare the NFT is.

- Star Atlas: Unlike Decentraland and The Sandbox, Star Atlas [95] runs on the Solana blockchain [183]. Solana is known to be faster and more secure than Ethereum. The native token in the Star Atlas Metaverse is known as ATLAS [7]. In this metaverse, users can build their own spaceship, have their own crew members, create their own planet, and explore the whole universe. In addition, this digital economy has a native currency known as POLIS [7] that can be bought with ATLAS to customize and manage the users' gaming experience.

7.6. Centralized Digital Economy Vs. Decentralized Digital Economy

Digital economy implementations are widely discussed among big companies such as Apple, Microsoft, Meta, and NVIDIA [101]. The deployment and use case of the digital economy concerns security, interoperability, scalability, and performance. Broadly, the digital economy is categorized into two forms: centralized and decentralized. In a central digital economy, a central authority would control the applications, businesses, transactions, etc. The centralized metaverse would inherit all the problems from the internet technologies we know today. A significant concern of the central digital economy is data privacy. For instance, through Facebook, Instagram, Whatsapp, and many other applications, Meta collects a myriad of information that breaches the privacy of users [90]. Another aspect of concern is interoperability [83], where giant companies like Amazon, Google, Meta, and many others do not collaborate to provide a better experience to the users. Moreover, this also affects scalability since the performance of such applications will depend on the expandability of the company infrastructure. Conversely, the decentralized digital economy commits a deeper and wider bucket to meet the rising demands of metaverse users. Distributed digital economy provides a realistic and immersive virtual experience lowering the risk of security and raising the level of interoperability and scalability. Decentralized digital economy commits to

keeping the secrecy of data through blockchain technologies with consensus-based transactions [171]. Moreover, the interoperability and scalability of blockchain can help the digital economy grow rapidly. Hence, considering the overall benefits derivable from a decentralized metaverse compared to the centralized version discussed above, the blockchain can be seen as an integral component of the metaverse.

7.7. Threat Analysis of Digital Economy

As discussed in previous sections, the digital economy is composed of several major complex technologies. Not only do these complexities provide tons of exciting experiences, but these complexities from these different technologies also invite various security loopholes to the integrated digital economy as well. As in this chapter, we have broadly classified the security vulnerabilities and threats into three broad areas, i.e., Information Theft, Identity Theft, and Cryptocurrency Theft [163]. We discuss some of their real-world examples below and in Table 7.1 on how they can affect the security and safety of digital economy adversely.

7.7.1. Use of Deepfake for Avatar Theft

Users are represented by a cartoonish avatar in the digital economy [105]. They are the embodiment and the identity of the users in the metaverse that allows them to pursue different adventures in different experiences. However, with the massive use of deepfake technology, the concern about the security of digital economy is soaring high [62]. Deepfakes use Artificial intelligence and Deep Learning, where an image, video, or audio of a person, in which their face or body has been digitally changed so that they appear to be someone else [177]. This is generally used for malicious intent or to spread hoaxes or false information. For example, recently, in June 2022, it was reported that the mayors of several European capitals had been convinced to participate in a video conference with a deepfake of the Mayor of Kyiv, Vitali Klitschko[3] [24]. In metaverse, the users are represented with an avatar or, hopefully, with a realistic representation of the future. So, for instance, when two users, buyer and seller, are in a business conference in the metaverse and are closing on a multi-million-dollar

[3]https://www.theguardian.com/world/2022/jun/25/european-leaders-deepfake-video-calls-mayor-of-kyiv-vitali-klitschko

business deal, it is possible that the seller can be somebody else instead of the actual seller and receive the money after closing the deal just because of his/her appearance. On the other hand, the buyer does not know that it is not the actual seller but a deepfake representation of the actual seller on some malicious attacker. Hence, in a place like a metaverse, where nothing is in person and everything is based on virtual reality and avatars, deepfakes can be a substantial severe threat.

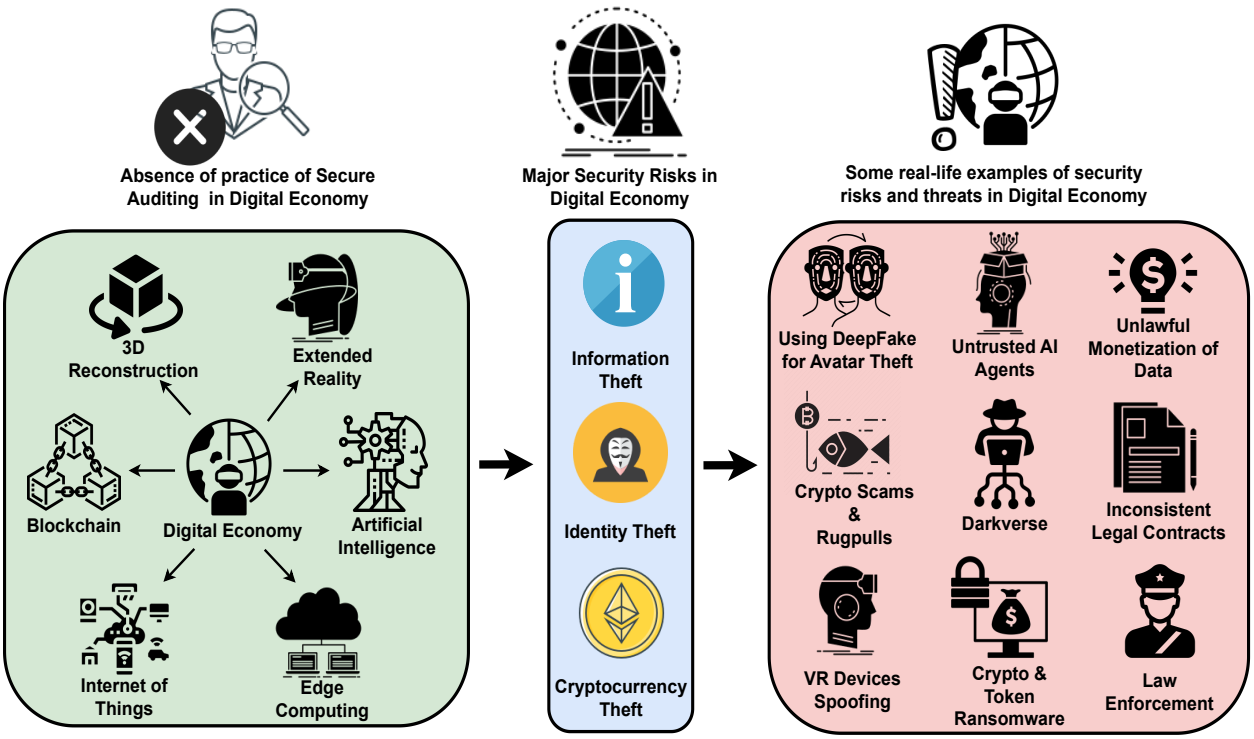

FIGURE 7.2. Metaverse as a digital economy is still in the germination phase, which composes of various major technologies that work together and provide users with an unprecedented experience. Nevertheless, the digital economy also has plenty of serious security threats and vulnerabilities that need immediate action. We discuss that with the use of the secure auditing technique incorporating multimodal Deep Learning approaches, it is possible to eliminate these threats, and safe and enriched experiences can be restored.

7.7.2. Crypto Scams and Rugpulls

Some of the well-known cryptocurrencies are Bitcoin and Ether. However, due to the hype and promising future, there are new cryptocurrencies being created almost every day as well. Some of these cryptocurrencies are legitimate, while others are just created to scam people. There are different types of crypto scams that already exist in the digital economy and Decentralized finance (DeFi) world. Some of the famous crypto scams are rugpulls, Ponzi schemes, romance scams, etc. For instance, A rugpull occurs when a developer attracts investors to a new cryptocurrency project, and the developers of the smart contract turn out to be fraudulent [80]. The fraudulent developers create a new crypto token, pump up the price of the token, then suddenly pull as much money out of the project as possible while the price is pumping before abandoning the project and the investors as the price of the so-called cryptocurrency/token project plummets to zero. If the rugpuller had never been given the majority of power, it would be difficult for him/her to run away with the investor's money. A famous example of this kind of attack is Squid Game Token[4] [120] when multiple investors could not sell their own tokens as the price of the token was pumped by twenty million percent. Then, the price of the token fell from USD 2,860 to nil as the rugpullers pulled out USD 3.3 million when the fraudulent developers abandoned the Squid Game Token project.

7.7.3. VR Devices Spoofing

When someone or something pretends to be something else in an attempt to gain a victim's confidence in order to get access to a system, then this is known as spoofing. In the metaverse, since VR devices are used by users all the time, VR device spoofing can happen when scammers fool the user's account by asking the victim to join the conference from somewhere it isn't. In addition, malware such as Trojan [5] can also be used to intercept and manipulate the communication between the metaverse network and the VR devices to commit fraud. The most common purpose of this attack would be to cause financial fraud by manipulating transactions that are controlled by VR devices, even when authentication

[4]https://nypost.com/2021/11/01/squid-game-cryptocurrency-plummets-nearly-100-in-scam-attack/

factors are in use, as the previously installed Trojan horse can be used to act between the VR devices and the metaverse security mechanism which allows sniffing the financial transactions as the user are browsing from their VR devices.

7.7.4. Untrusted AI Agents

Artificial intelligence (AI) facilitates intelligent decisions for various applications. AI models are designed with well-known algorithms proven to yield high accuracy with many modes of learning. One of the critical problems in recent development involves the trust of the data and model [16]. For example, data poisoning attacks create untrustworthy applications where input data, the machine learning model, and output data can be questioned [25]. If the data and model of the machine learning process are altered, then we can not trust the results. Similarly, we can not trust classification or prediction if it is not trained with immutable original data and model. Another perspective of trust is the fairness and explainability of the learning models [16]. As another example, the developers can bias machine learning models on particular features such as race, gender, and ethnicity, which can question the model itself. A trustable machine learning model, therefore, is required in modern applications that demand security, privacy, and immutability [16].

7.7.5. Darkverse

Just like Darknet, the digital economy can also harbor a darkverse which can act like an overlay network within the digital economy that can only be accessed with specific software, configurations, or authorization. Darkverse [82] can be a gathering place for cybercriminals that can make experiences in the metaverse a risky activity. Darknet can be a hotspot for several reasons, such as conducting organized crimes, sharing illegal files, selling restricted goods and tokens, leaking news and whistleblowing, etc.

7.7.6. Crypto Hacks and Token Ransomware

Another threat that probably will be very popular in digital economy is cryptocurrency and token ransomware. In this type of security threat, a malicious program sent by an

attacker encrypts the files stored on the victim's computer in order to extort the cryptocurrencies and tokens that the victim owns. The files or links can be sent to the victim by a malicious attacker via emails, messages, or other modes, and when they are downloaded on the victim's end, the malware will encrypt and lock the files until the victim sends the asked amount of extortion in the form of cryptos and tokens. Likewise, a famous DAO attack that caused a loss of USD 60 million in 2016 [117] that can also occur in the metaverse is known as a Reentrancy attack [149], where a malicious contract exploits the code in a vulnerable contract to drain its funds as shown in Fig. 7.3.

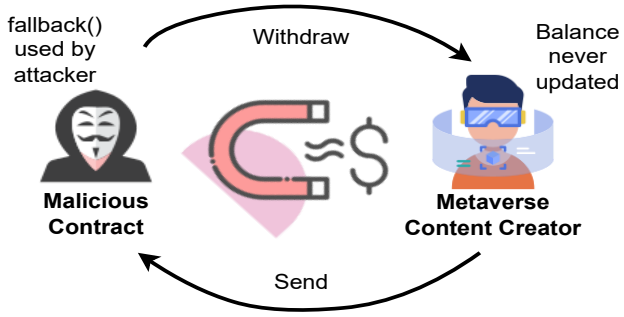

FIGURE 7.3. In the Reentrancy attack, the attacker can use a fallback function in the malicious contract and can continuously call the withdraw function to drain the Metaverse Content Creator's funds when their contract fails to update its state before sending funds.

7.7.7. Unlawful Monetization of Data

One main concern of the digital economy is data privacy. User's information in the metaverse can be stolen by hackers with the use of malicious contracts, malware, and spyware with the rise of phishing attacks. Similarly, AR and VR devices are also already proven to be vulnerable devices [2]. Hence, as metaverse will be a digital/virtual copy of real life, it will collect personal information from users, including brainwaves, biometric data, health information, preferences, etc., and the absence of proper and strict documentation of legal policies can allow the malicious users and companies to make illegal monetization of data.

7.7.8. Inconsistent Legal Contracts

As the digital economy is adopting blockchain and smart contract technologies, the legal contracts that comprise business logic and the policies that are converted into smart contracts should be clear and explicit [167]. Legal contracts are infamous for their legal jargon and vagueness. As the legal contracts employ vague words and phrases abundantly, this results in multiple interpretations for multiple readers [165]. As a result, while converting these legal contracts to smart contracts, the probability of inconsistency increases. The more there is an inconsistency between the actual legal contract and its smart version of the contract, the more vulnerable the metaverse can become, as it allows the attack surface to become bigger.

7.7.9. Law Enforcement

As social media have its own world, the digital economy will have its own world, too, probably even with the potential of being more significant than anything else. Since the digital economy also has the equal potential to be a perfect network to perform illegal activities such as phishing, ransom, fraud, money laundering, social engineering, propaganda, and fake news [148], it will be very difficult to trace, monitor and infiltrate by the law enforcement.

7.8. Importance of Secure Multimodal Auditing

As technologies such as Deepfake that are derived from Deep learning are responsible for the security threats in the metaverse for identity theft, but at the same time, deep learning models and algorithms can prevent tons of security threats and vulnerabilities, as it is one of the most powerful technology [104]. For this, all the modes such as text, image, audio, video, speech, and biomarker signals have to be provided great attention and have to be audited, including the core smart contracts, so that there is no room for any security loopholes as shown in Fig. 7.5. Likewise, one of the essential features of a decentralized digital economy is that it is incorruptible. Whatever goes inside the blockchain can never be changed due to the use of the impossible-to-break cryptography technique used.

TABLE 7.1. Digital Economy Services, Components, Security Risks, and Audit Remediation Technologies

Digital Economy Services	**Prominent Digital Economy Components**	**Examples of Security Risks in Digital Economy**	**Audit Remediation Technologies**
Social Network	AI (Deep learning), XR	Using Deepfake for Avatar Theft, Untrusted AI Agents	Time-stamped blocks for data provenance and Generative Adversarial Networks in Deep learning
Gaming	Deep learning, XR, 3D Reconstruction	VR Devices Spoofing	Cryptographic functions and signatures in blockchain
Trading	Blockchain and Smart contracts	Crypto Scams, Rugpulls, Ransomware, Unlawful monetization of data	Recurrent Neural Networks in Deep learning
Events	AI, 3D Reconstruction	Darkverse, Improper Law Enforcement	Blockchain and Pattern Recognition in Deep learning
Remote Working	Internet of Things, Edge Computing	VR Devices Spoofing, Inconsistent legal contracts	Cryptographic functions and signatures in blockchain
Entertainment	AI, XR	VR Devices Spoofing	Behavioural analytics and Pattern recognition in Deep learning

Hence, this is the reason why blockchain is known to be exceptionally secure. Nevertheless, one disadvantage of blockchain is that it is tremendously expensive to operate due

to its distributed and decentralized nature. Therefore, any facility we use from blockchain technology is also costly. As we know, a smart contract, that is, a computer program, is one integral component of blockchain and resides in the decentralized digital economy.

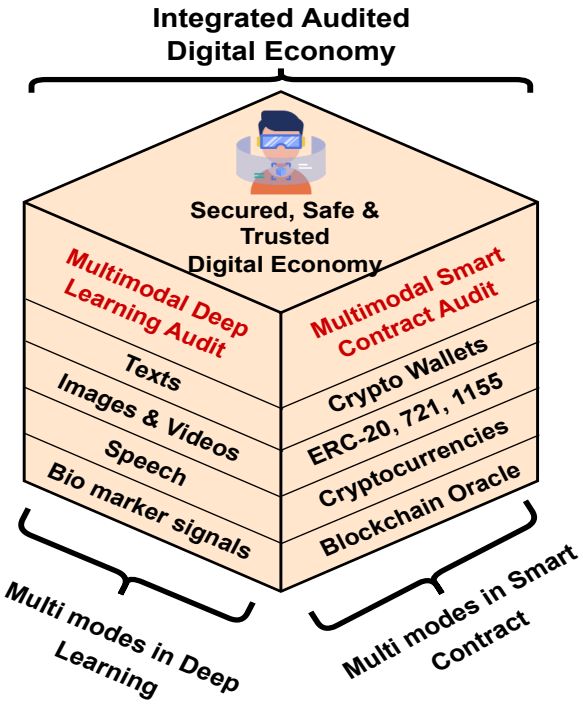

FIGURE 7.4. The cube represents that there are various modes in both Deep learning and Smart contract which needs to be audited multi-dimensionally in order to achieve a safe and trusted digital economy.

Similarly, it is also known that anything that is in the blockchain is immutable and irreversible. Hence, it is significant to understand that, like any other programming language, it is not possible to keep making changes to the smart contract, as once the smart contract is deployed into the blockchain, it stays immutable. However, even an experienced and smart developer can make some mistakes while writing a smart contract, as its inherent to all human beings, and nobody is perfect.

Therefore, it is crucial that before the smart contract is ready to be deployed into the digital economy, it is checked over and over again and, if possible, audited by a third

party. This way, the developers can save millions of dollars if there are any potential bugs or critical security risks hidden in the smart contract for malicious hackers to exploit. One famous example of why secure auditing is so vital was the DAO hack [117] on the Ethereum blockchain when approximately USD 60 million was stolen in 2016. In addition, there is a possibility that the smart contracts sent by the developers to the auditors for secured auditing can be optimized in terms of gas fees and transaction fees so that even if the contract is working properly without any security risk, the gas fees can still be lowered drastically.

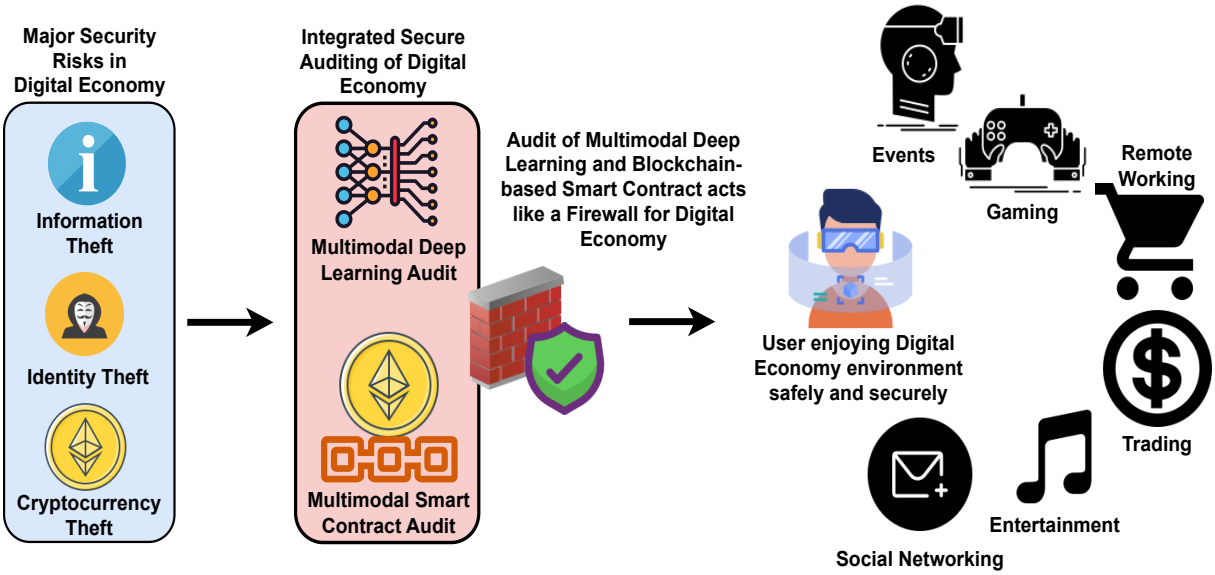

FIGURE 7.5. As there are three major security risks in the digital economy, which are Information, Identity, and Cryptocurrency Theft, in our model, we make integrated secured auditing the focal point that takes all the aspects and layers of decentralized digital economy via Multimodal Deep learning and Smart contract audit into consideration to ensure all entities are safe and secure and can achieve enriched experiences in the digital economy.

7.9. Phases in Secure Auditing

In secure auditing [75], a given project codebase is examined, and then mitigation strategies are recommended by the security auditors to the project developers so that there

is no room for security loopholes and vulnerabilities. Although, generally, smart contracts are written in Solidity, but can be in any other language as well as presented in Table 2.1.

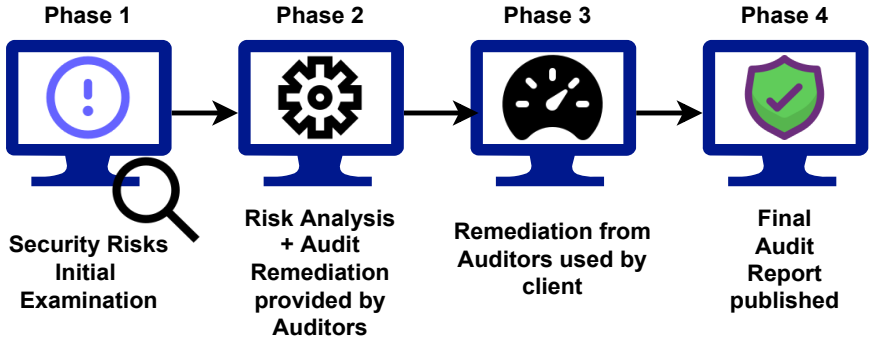

FIGURE 7.6. Division of the process of Secured Auditing of digital economy in four different phases. The secure audit time depends on the size and complexity of the project.

Typically, secured auditing has the following four major stages:

(1) Phase 1: The project codebase is provided by the developers to the security auditors for initial examination and analysis.

(2) Phase 2: The security audit team examines the project codebase provided by the project developers using the static analysis method and/or manual auditing method and then presents all the critical, medium, and minor level findings that are related to security vulnerabilities, inconsistent business logic with respect to the code and optimization issues in a preliminary audit report. Likewise, all the remedies and mitigation strategies to deal with all the presented findings are provided to the developers so that they can act upon them.

(3) Phase 3: The project developers then make changes to their codebase according to the preliminary report provided by the security audit team.

(4) Phase 4: After the project developers make changes to their codebase and there are no new changes to be made or no errors and security vulnerabilities, the security audit team finally publishes their final audit report and provides it to the project developers.

7.10. Types of Secure Auditing

Secure auditing is classified into two main types. The first one is known as static analysis, and the second is manual auditing which we further discuss below.

(1) Static analysis auditing: Static analysis auditing is a process where the debugging and examination of the codebase are performed by the security auditors without executing the program. In this kind of auditing, the audit tools scan and identify the security risks presented in the codebase. Static analysis is typically easier as well as faster to perform. Some of the popular static analysis tools are Slither, Mythril, Securify, SmartCheck, Echidna, etc. [75]. These tools help in performing tasks such as automated vulnerability detection, automated optimization detection, code review, and understanding.

(2) Manual auditing: Manual auditing takes more time and effort compared to static analysis, as security auditors have to examine each and every line of code carefully. Sometimes, there are tens of thousands of lines of code that auditors have to examine and other times less. There is no hard and fast rule on how security auditors audit a given project as different auditors have different ways to perform their examination as its a constant and to-and-fro process. Nevertheless, the following are some critical discussions that a security auditor has to consider while performing manual auditing. We also discuss why static analysis by itself is not enough and why manual auditing is significant. We again divide manual auditing into two categories based on deep learning and smart contract and briefly discuss how they can assist as audit remediation strategies.

Examples of Vulnerabilities in Smart contract	**Cause of the Vulnerability**	**Audit Remediation Strategies**

Reentrancy Attack	A recursive call is made by the untrusted contract back to the function from the original contract to steal the funds from that contract	Check-effect-interactions pattern, Reentrancy Guard from OpenZeppelin [170]
Centralization Risks	Overpowered owner, Initial token distribution, Rugpulls, Private key leaks	Safe management of private key, Usage of Multi-Signature wallets, voting modules for transparency and consensus
Integer Overflow and/or Underflow	The value circles back to zero due to an increment by 1 bit and can be used repeatedly to keep increasing the value. On the contrary, instead of going beyond the range in the highest order, an underflow error occurs when it goes below the range	Using the SafeMath library from OpenZeppelin or by manually checking for integer overflows and underflows. This vulnerability has been solved since Solidity upgraded its version to 0.8.0. [170]
Authorized Proxies	If a smart contract is capable of behaving like a proxy by being able to call other smart contracts with the data provided by the user, then that user also can figure out the identity of the proxy contract	Build a system in the smart contract where the proxy does not have any permissions, including for itself

Absence of Address Validation	Without sanity checks and missing validations for the correct address, one might send the funds to incorrect addresses or bring the privileges to a certain malicious entity.	Sanity checks and validation should be performed whenever possible to ensure that the provided addresses are accurate and also to verify they are not zero addresses.
Gas limits and costly loops	Loops that do not have any explicit limit and are dependent on storage values can drain the funds due to their expensiveness	Loops should be used very carefully and in necessary cases in smart contracts. Irrelevant functionality and libraries should be removed.
Arithmetic precision	Division performed before the multiplication can cause issues related to rounding as division before multiplication truncates the lower bits, which will lead to the loss of precision in the calculation	Multiplication should always be used before division instead in a mathematical expression and equation to achieve arithmetic precision

Table 7.2: Common security vulnerabilities, cause, and audit remediation strategies in Smart contract

7.11. Integrated Secure Multimodal Audit as a Remediation Strategy

7.11.1. Multimodal Deep Learning Audit

- Prevent deepfake avatar theft: Deep learning has been both a source of the problem and a remediation strategy for avatar theft via Deepfake technology. As deep learning is making it easier to steal avatar's identity, it is also helping in the detec-

tion of the stolen avatars and the potential threat. Researchers have been able to distinguish between fake and authentic media by looking for inconsistencies among the deepfake from frame to frame [113]. In another deep learning model, also known as the biological feature deepfake detection model, the researchers were able to detect the deepfake by identifying biological traits only an actual human can possess, which includes anatomical actions, such as heartbeat, breathing, and blood flow [9]. These signals were easily detected by deep learning algorithms, as even the subtle change of light or reflection on the face can be a key point. Deep learning merged with blockchain can even make it easier when it comes to detecting and preventing deepfakes in the metaverse. In case an Avatar is stolen by an attacker with malicious intent, it is possible to find the source of the original and stolen Avatar using blockchain, as everything in the blockchain is time-stamped with cryptographic functions, as this process can be augmented by deep learning algorithms.

- Predict crypto scams, rugpulls, and ransomware: Crypto scams, especially rugpulls, and pump-and-dump, are caused when fake recommendations and speculations boost the price of the potential cryptocurrency, and when the prices are soaring high, the scammers dump shares of cryptos by selling their own shares at inflated prices. Deep learning models and algorithms can be used to predict if it actually is some kind of crypto scam, a rugpull, or ransomware [34]. Generally, these scams often have multiple phases that occur over a different periods of time. With the collection of standard anomaly detection datasets, researchers have been able to identify and detect longer-term anomalies as well as shorter-term anomalies in crypto scams. Likewise, it is also possible to detect crypto ransomware before the encryption to prevent the files from being irreversibly encrypted. An effectively and efficiently designed deep learning algorithm can analyze billions of relevant data and detect threats and suspicious activities or keywords in milliseconds without any human intervention. Financial fraud, malware, and ransomware detection with the use of the machine and deep learning have been considered a popular mechanism,

and there is no doubt this can be a powerful tool in the realm of the metaverse as well.

- Reconcile legal contracts and program code: As reading and understanding legal contracts are still one of the most arduous tasks to perform, with the advancement in deep learning algorithms, this process can be done quickly without the supervision of humans. In addition to contract analysis, deep learning can also facilitate understanding the semantics and syntax of the program code. These days, there are already a few tools that analyze the business logic or the legal contracts first and then convert them to pseudo-code and program code with the use of Natural language processing and Deep learning algorithms [164]. Deep learning has provided unprecedented ways to identify and extract critical variables, such as clauses, dates, names, entities, and IF/ELSE statements.
- Strengthen trust in AI agents: Deep learning needs less human intervention compared to traditional machine learning. Deep learning helps to establish a strong level of trust as it has the capacity to execute feature engineering on its own. Also, deep learning performs very well on multimodal such as image, audio, text, etc., and the models that are used in deep learning are more reliable and accurate as the amount of data used to train these models is huge compared to traditional machine learning models. AI agents are fed information from the raw data, which has a statistical correlation to identify any labels or continuous targets. Deep learning algorithms can train data with higher dimensions and samples, which is essential to study a more convincing pattern for making correct decisions [16]. Apart from that, the mathematical computations involved in deep learning algorithms study the data with more granularity and preciseness. Higher accuracy of learning and predictions can make AI models more meticulous, which makes deep learning inevitable for a trusted digital economy.
- Prevent illegal monetization of user data and criminal activities in Darkverse: Deep learning can analyze billions of data and then identify a pattern on who is stealing

the users' data for illicit monetization or for any other specific criminal activities in darknet or darkverse [33]. As deep learning algorithms are always connected to the database, pattern recognition algorithms can be used to scan through the records and transaction details that are stored over the years. Deep learning can immediately spot any suspicious patterns with the use of behavioral analytics, which helps to understand and predict an individual's behavior for different transactions. The use of deep learning in data theft detection is a step forward in multimodal auditing to ensure the safety and security of the digital economy.

7.11.2. Multimodal Smart Contract Audit

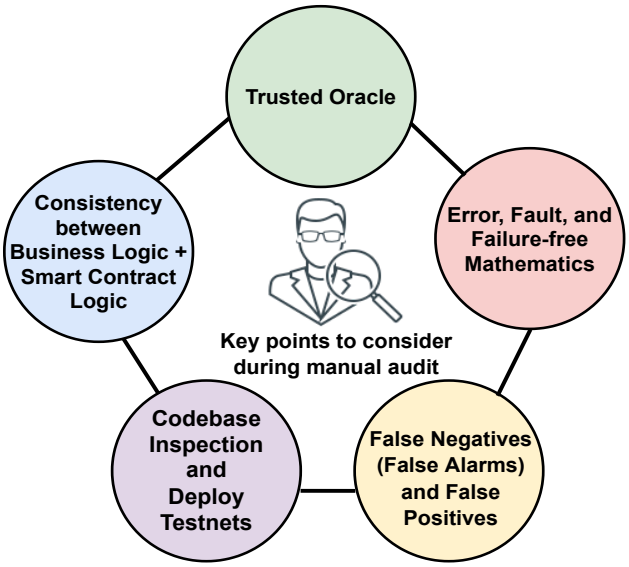

FIGURE 7.7. In addition to all the security risks mentioned in Table 7.2, the auditor needs to ascertain the business logic is consistent with the smart contract logic. Other crucial points in manual audit to consider that cannot be performed with static analysis are trusted oracle, mathematical equations, false negatives and positives, and final testing by deploying in testnet.

- Consistency between business vs. smart contract logic: In many cases, the intention and the implementation of smart contracts happen to be different. While the business logic delineates the purpose and rules of the smart contract in one way, the programming logic of the smart contract can go another way. This can open

the door to a lot of vulnerabilities when the blueprint of the project does not match the actual product [6]. These types of issues are simply out of an automated audit tool's reach. Hence, manual auditing is crucial in cases like these to understand the intention and rules of the given project and how consistent the codebase is with the provided business logic in the documentation files. There should always be consistency between the business logic of the project with its actual programming logic.

- Trusted oracle: An external entity that provides the external data as input to the blockchain or performs some external computation and delivers the result to the smart contract in the blockchain is known as an oracle [27]. An essential question any security auditor needs to ask themselves while performing audits manually is if the oracle that the given smart contract is using can be trusted. Hence, while auditing the project codebase provided by the developers/clients, an auditor is always supposed to check if the external inputs from an oracle can be trusted. Lack of validation check for inputs from oracle to the smart contracts can leave room for severe and critical bugs and security vulnerabilities.
- Error, fault, and failure-free mathematical equations: There are plenty of mathematical equations used in the smart contracts ranging from simple to complex nature, especially when the smart contract is related to DeFi and the smart contact is responsible for the calculation of stakes, tokens, fee, etc. Solidity as a programming language for the smart contract already has an issue with the floating point numbers. In addition, the security auditors need to be double-sure that the mathematical equations and expressions in the project codebase are not just the correct equations and expressions but also don't have any calculation pitfalls. An automated audit tool with static analysis might be able to find issues related to the arithmetic signs and expressions; however, to understand the working mechanism of these mathematical equations, an auditor must investigate them. For example, various ranges of numbers as input can be provided to all the parameters in equations,

in both numerator and denominator, to investigate if any pitfalls, errors, faults, or failures are hidden.

- False negatives and false positives: False negative is a situation when an attack has taken place, but no alarms have been raised. A security auditor should definitely not rely only on static analysis tools as these tools cannot always identify, sometimes even, critical bugs and security risks. For this reason, an auditor may use the static analysis but, in addition, must also manually audit the codebase line by line to scrutinize if static analysis tools missed something. Almost always, with a thorough manual audit, auditors are able to find many hidden vulnerabilities in the codebase. On the other hand, a false positive is a situation when false alarms are raised. For example, there are also plenty of cases when the static analysis tools identify an activity or event as an attack due to their high sensitivity, but that activity or attack is benign and acceptable in a practical scenario. So, auditors need to be careful in first confirming if it's actually a false positive case and removing these false positive cases from the findings.
- Codebase testing in test networks: A significant step of manual audit also requires the auditor to identify the behavior of the smart contract when it is deployed onto the test networks. After deploying the smart contract onto the testnet, each parameter and function should be checked by passing several rounds of inputs to see if they are behaving properly. This way, the auditor can also find out more about the smart contract's gas cost efficiency [37] and how they can be optimized in the best way possible.

7.12. Conclusion

Learning about systematic threats and security analysis is crucial to metaverse, a kind of digital and virtual economy as it is still in the germination phase. In this chapter, we discussed the unprecedented security vulnerabilities as significant challenges that digital economy will be facing in the near future. We also argued why the digital economy needs to be decentralized. Then, we highlighted how secure auditing could assist and be of service

to the digital economy. Furthermore, we discussed why and each mode of deep learning has to be audited with respect to the smart contract for its proper, safe, and secure functioning. For each security vulnerability example, we also provided the respective remediation, mitigation, and countermeasure strategies from an integrated multimodal deep learning and smart contract audit perspective.

CHAPTER 8

SUMMARY AND CONCLUSION

8.1. Summary

A blockchain-based smart contract is in the germination phase, but it is also one of the hottest topics and emerging technologies. It is spread not just in the scope of computer science but also towards computational law and computational linguistics. We have discussed why a smart contract is advantageous and beneficial to bring into practice now. Due to its unique features, such as being explicit, non-vague, self-executable, distributive, decentralized, immutable, cryptographically secured, and having faster settlement ability, it is undoubtedly true that smart contracts are better legal contracts compared to the traditional versions of legal contracts. In addition, it is also better than conventional contracts because it entirely eliminates the issues of vagueness in the contract as it is written in the code. As a result, the involved parties in the contract would not have to suffer from confusion, multiple interpretations, and misunderstandings. Therefore, the smart contract is the answer for an effective and efficient tool that can accommodate increasing the clarity and accuracy and reducing the complexity of the dispute settlement process.

The focus of the studies performed in this entire dissertation is to propose an alternative solution where traditional legal contracts, which are inherently vague, fuzzy, and ambiguous by nature, are considered and translated to blockchain-based smart contracts that are explicit, vague-free, and self-executable. The legal contracts and policies written by the lawyers always have been vague and extremely unclear to consumers who belong to various backgrounds with different real-world knowledge. When the consumers have not been satisfied with the services from a service-providing organization, they need to take the reference from the legal contract or service-level agreement they are bound with. However, the way these legal contracts and service-level agreements are designed and written in such ways that consumers from other backgrounds cannot completely understand. The usage of "legal language", although it is necessary to use, simultaneously invites a plethora of issues,

especially for consumers who do not have much knowledge and understanding of legal jargon. Even if they do, it still is a massive hassle for the consumers to file complaints against the service providers for compensation, as the traditional legal contracts do not provide the facility of self-execution when the predetermined "vague" conditions are triggered. Therefore, there is a need for an exhaustive and deeper understanding of the challenges that vagueness brings.

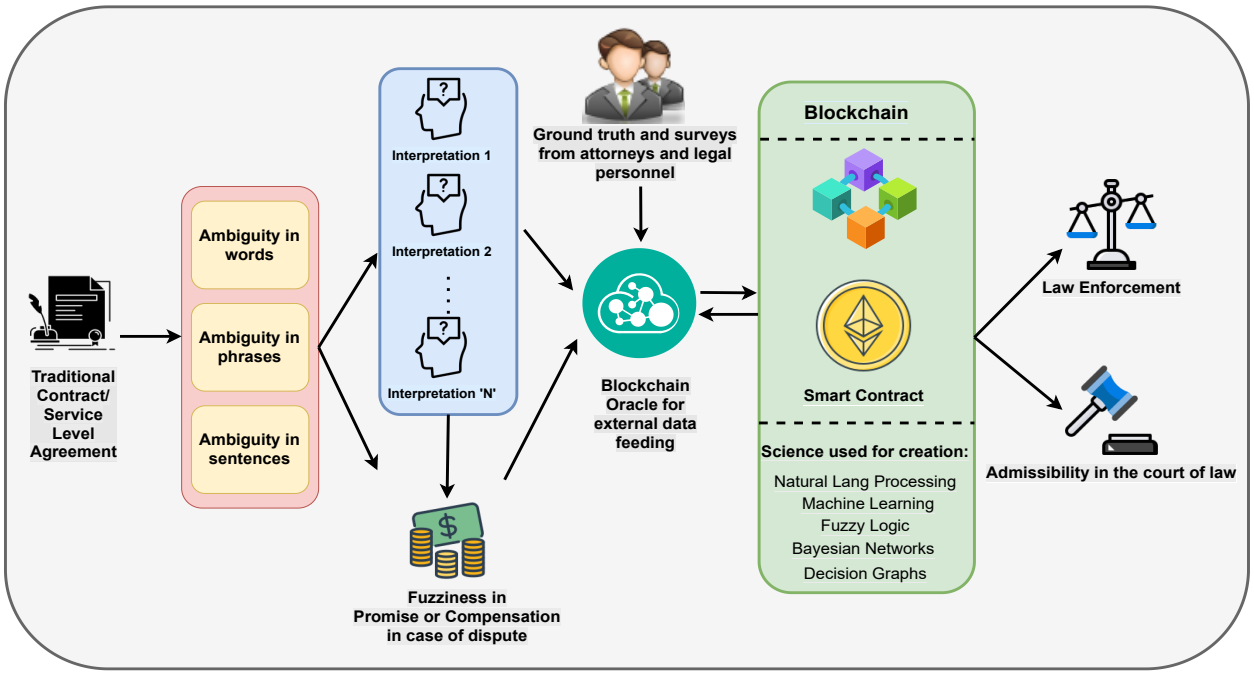

FIGURE 8.1. Conversion of the traditional paper contract to a legally enforceable blockchain-based smart contract. Here, a paper contract has plenty of possibilities for vagueness. Hence, the vagueness is explored on the word, phrase, and sentence level, and their corresponding interpretations are created and fed onto the blockchain with the help of blockchain oracle, including the fuzziness of the agreements and ground truth from the lawyers. Once the smart contract is created, it is made enforceable and admissible in the courts of law [167], [166].

Hence, this entire dissertation focuses on the novel proof-of-concept methods that explain how existing traditional legal contracts that are vague and fuzzy can be transformed

into blockchain-based smart contracts. This dissertation also delineates and measures the behavior of legal contracts after being transformed into smart contracts inside the blockchain network.

A novel study was performed in Chapter 3, where a real and existing crowdfunding legal contract that had 12 clauses or sections in total was translated into the blockchain-based smart contract. This contract had a lot of vague words and phrases in every clause. However, only four clauses which are "Agreement", "The Project", "Rewards", and "Contribution and Payment" were considered for translating from the vague legal contract into the smart contract due to the fact these four clauses revolve mainly around the crowdfunding and transactional mechanism. The vague words and phrases were manually hand-picked, and all multiple possible meanings with degrees of truth were generated for the analysis of each interpretation. All generated interpretations were portrayed in the control flow graphs, where they were quantified in order to measure their complexities. In addition, these generated interpretations were translated and transformed into smart contracts that are based on the control flow graphs. Finally, all these different possible interpretations that were translated into the smart contracts were deployed onto the blockchain for the analysis and evaluation of their behavior in the blockchain network. In this chapter, we found that the more vague and fuzzy the crowdfunding legal contract was, the more deployment and transaction fees it was also incurring in the blockchain network. As the whole crowdfunding could not be translated into a single, smart contract due to the presence of the clauses in it, such as activities, events, etc., that included physical and non-transactional activities, only a subset of the whole legal contract was taken for the conversion.

In Chapter 4, another novel proof-of-concept methodology was performed in order to analyze the degree of uncertainty and vagueness in real-life popular internet service providers' SLAs. SLAs from popular vendors, such as AT&T, T-Mobile, Spectrum, CenturyLink, etc., were chosen for this study. Instead of manually hand-picking the vague words and phrases from these SLAs, machine learning was used to binary classify the vague words from non-vague words from two test SLAs which were Ziply Fiber and CenturyLink. We used Shan-

non's entropy to measure the uncertainty index of various interpretations that was derived from the root SLA of both Ziply Fiber and CenturyLink. Similarly, we used McCabe's cyclomatic complexity to measure the vagueness index in both of these SLAs. When all the generated interpretations were translated into each of the smart contracts and deployed onto the blockchain for the evaluation of their behavior, we found out that Ziply Fiber's SLA consumed more deployment and transaction cost on average compared to CenturyLink's SLA, which also corroborated the results that were generated from a measurement of uncertainty and vagueness index of both of these SLAs.

As fuzzy logic is extremely good at modeling logical reasoning with vague and imprecise sentences, we used a weighted fuzzy reasoning technique as a mathematical model in Chapter 5, which can easily handle the problem of vagueness in a service-level agreement that causes communication gap due to the multiple interpretations between a company and its customer. With the help of fuzzy quantifiers, certainty levels, and their corresponding numerical intervals, the set of complaints filed by customers based on different information available in an SLA, such as performance, operation, availability, latency, jitter, and maintenance were efficiently quantified so that they can be used inside the production rules. We also introduced the similarity measurement function along with the weighted vector, which finds out the degree of similarity between two fuzzy sets, which are a set of metrics in an SLA and a set of customer's manifestations. Weighted vectors allowed us to allocate the degrees of importance in the antecedent part of the production rule, whereas the certainty factor allowed us to measure the confidence of the consequent part. Finally, we were able to demonstrate an example of how an analysis can be performed that can measure and tell us the confidence level of customers regarding their claim for compensation when their service is not proper. This technique of weighted fuzzy reasoning acted like a consumer's claim and compensation interpretation diagnosis system.

In Chapter 6, our methodology consisted of the architecture where we designed a smart contract that incorporated the fuzzy logic mechanism inside. As it always has been a big issue for customers to claim compensation when the service they have been getting

is not proper, understanding the contents of legal contracts and service-level agreements first is a necessary step that most of the customers can't seem to pass due to the contract being full of vague words and legal jargons. Therefore, we introduced a very novel method where the whole smart contract is designed with the core idea of fuzzy logic, which has four main components. These four components of fuzzy logic-based smart contracts are fuzzifier, rule-based system, inference engine, and defuzzifier. We created three smart contracts with the same architectural designs except for the linguistic descriptors for experimental and further evaluation purposes. The first smart contract had the lowest linguistic descriptors, the second had higher, and the third had the highest. From our evaluation, we concluded that the smart contract, which had the highest degree of truth or linguistic descriptors, was more accurate compared to the other two. However, at the same time, the most accurate smart contract was also incurring more deployment and transaction costs compared to the other two. Similarly, out of the four major functions inside the smart contract, the function for rules' strength evaluation was the most expensive due to its complex nature. Finally, by incorporating the idea of fuzzy logic inside the smart contract, the smart contract was not only smart but also intelligent as now the smart contract had the ability to use vague linguistic descriptors and defuzzify them into the crisp results that a layperson can understand.

At last, in Chapter 7, we discussed the security threats and vulnerabilities that a digital or virtual economy, such as the metaverse, can harbor if smart contracts are used inside a digital economy for various purposes. In the digital/virtual economy, as there will be three major security risks, which are information theft, identity theft, and cryptocurrency theft, we presented a security model where we made "integrated secured auditing" the focal point. This security model for the digital economy, such as the metaverse, emphasizes multimodal auditing in two main sectors: multimodal deep learning audit and multimodal smart contract audit. The multimodal deep learning audit focuses on the auditing of various modes such as texts, images, audios/videos, speeches, and biomarker signals, whereas the multimodal smart contract audit focuses on the auditing of tokens, smart contracts, cryptocurrencies, crypto wallets, and blockchain oracles. In this chapter, we have presented the

relevant threats and vulnerabilities that a digital economy can invite due to the usage of smart contracts and also suggested preventative remedies for the defense mechanisms.

The contract originated as a paper contract and is still widespread worldwide from tiny to large tasks, but the development kept on advancing. Eventually, the paper contract evolved into electronic or digital contracts and from digital contracts to Artificial intelligence-based contracts, and finally to blockchain-based smart contracts. Looking back at the developmental trends in computer science and information technology sector, it is evident that the paradigm shift of the contract from the traditional paper contract is not going to stop at blockchain-based smart contracts. There will be other advanced technologies in the future that will adopt the contract and add more features and make the contract even more versatile and efficient. Regardless of how advanced and smart the contract is, the usability of such smart contracts should always be encouraged, and the legal enforceability of such smart contracts should always be maintained. We hope that this dissertation will facilitate the researchers currently working on legal contracts and smart contracts to find new paths and open problems to tackle in the coming years.

8.2. Challenges and Directions for Future Work

Understanding the semantic legal terms, which makes a contract vague, will always be a challenge as these legal words are understood clearly only by certain people whose profession lies in the legal sector. The idea of a Smart Legal Contract is itself a novel idea. Conversion of the legal terms into a smart contract correctly without being vague is quite difficult. Also, the whole contract might not be converted into a smart contract since the activities and events in a legal contract might include physical and non-transactional activities. One of the main challenges we faced was to label all the tokens manually as vague or non-vague in the training dataset correctly. Labeling the dataset involved meticulous planning while preparing the dataset at the beginning of this project because many words are vague literally for a layperson but may not be considered as vague by the lawyers who draft the SLAs. Another challenge was to increase the accuracy of the existing model that we used to classify and detect the vague words which were used to generate various interpretations

of SLA that can later be translated into their respective smart contracts. Nevertheless, we are persistently working to gather more SLAs from vendors to increase our training dataset, which will help increase the accuracy of the model.

There is plenty of research possibility in this scope, especially with the exponential rise in the advancement of the digital economy, artificial intelligence, and blockchain technology in recent years. Some of the possible research topics and directions for our future work are as follows:

- Use and development of natural language processing systems and artificial intelligence for legal contract analysis by extracting the texts of a given contract automatically and generating all possible interpretations to find vagueness can be a fascinating area of research.
- In our existing work, we have used a simple binary classification model to classify vague from non-vague words and phrases with decent accuracy considering our dataset's size. However, increasing our dataset exponentially and then using Generative Pre-Trained Transformer 3 (GPT-3) [46] and Bidirectional Encoder Representations from Transformers (BERT) [97] to get much better and more accurate predictions of vague words would also be a part of the future work. In addition, these technologies also excel at capturing contextual information and can aid in comprehending how specific words and phrases in a clause should be interpreted within a given context.
- Use of other defuzzification techniques such as the mean-max method, center of sum (COS) method, and weighted average method in the smart contract and comparing the accuracy of the defuzzification with the existing center of gravity (COG) method is a part of our future work.
- Regarding fuzzy logic-based smart contracts, future work also includes comparing the ground truth of smart contracts that implement various defuzzification methods with the legal department of the vendors that calculates and decides the compensation in their SLA.

- A future work would also be developing a smart contract comprising the methodology of type-2 fuzzy logic [94], which would be an extension of our existing word of fuzzy logic and would be characterized by a three-dimensional membership function. Type-2 fuzzy logic systems would provide a higher level of uncertainty management compared to type-1 fuzzy logic.
- Comparing the ground truth of the translated smart contracts with the lawyers and studying the contrast integrity of our vagueness index with the lawyers' measurement standard is a major part of our future work, which would help us better tune the parameters of our study to increase the accuracy.
- The security of the translated smart contracts needs to be provided with more attention simultaneously. Hence, the security model of integrated multimodal auditing that we discussed in chapter 7 of this dissertation will be studied further and be implemented in the future as a remediation strategy for the smart contracts used inside digital economy.

REFERENCES

[1] Bruce A Ackerman, *Law, economics, and the problem of legal culture*, Duke LJ (1986), 929.

[2] Devon Adams, Alseny Bah, Catherine Barwulor, Nureli Musaby, Kadeem Pitkin, and Elissa M Redmiles, *Ethics emerging: the story of privacy and security perceptions in virtual reality*, SOUPS@ USENIX Security Symposium, 2018, pp. 427–442.

[3] KA Adams, *Sources of uncertain meaning in contracts*, Michigan Bar Journal (2016), 40–45.

[4] Klaus-Peter Adlassnig, *Fuzzy set theory in medical diagnosis*, IEEE Transactions on Systems, Man, and Cybernetics 16 (1986), no. 2, 260–265.

[5] Monika Agrawal, Heena Singh, Nidhi Gour, and Mr Ajay Kumar, *Evaluation on malware analysis*, International Journal of Computer Science and Information Technologies 5 (2014), no. 3, 3381–3383.

[6] Wolfgang Ahrendt, Richard Bubel, Joshua Ellul, Gordon J Pace, Raúl Pardo, Vincent Rebiscoul, and Gerardo Schneider, *Verification of smart contract business logic: exploiting a java source code verifier*, Fundamentals of Software Engineering: 8th International Conference, FSEN 2019, Tehran, Iran, May 1-3, 2019, Revised Selected Papers 8, Springer, 2019, pp. 228–243.

[7] Hilmi Tunahan Akkus, Samet Gursoy, Mesut Dogan, and Ahmet Burak Demir, *Metaverse and metaverse cryptocurrencies (meta coins): Bubbles or future?*, Journal of Economics Finance and Accounting 9 (2022), no. 1, 22–29.

[8] Omar Adil M Ali, Aous Y Ali, and Balasem Salem Sumait, *Comparison between the effects of different types of membership functions on fuzzy logic controller performance*, International Journal 76 (2015), 76–83.

[9] Abdulqader M Almars, *Deepfakes detection techniques using deep learning: a survey*, Journal of Computer and Communications 9 (2021), no. 5, 20–35.

[10] Ali Alzubaidi, Karan Mitra, Pankesh Patel, and Ellis Solaiman, *A blockchain-based approach for assessing compliance with sla-guaranteed iot services*, 2020 IEEE International Conference on Smart Internet of Things (SmartIoT), IEEE, 2020, pp. 213–220.

[11] Oxford Analytica, *Coinbase listing will pave way for other exchanges*, Emerald Expert Briefings (2021), no. oxan-es.

[12] Andrew Antos and Nischal Nadhamuni, *Practical guide to artificial intelligence and contract review*, Research Handbook on Big Data Law, Edward Elgar Publishing, 2021, pp. 467–481.

[13] Robert J Aumann and Aviad Heifetz, *Incomplete information*, Handbook of game theory with economic applications 3 (2002), 1665–1686.

[14] Mohammad Fazle Azeem, *Fuzzy inference system: theory and applications*, BoD–Books on Demand, 2012.

[15] Syed Badruddoja, Ram Dantu, Yanyan He, Mark Thompson, Abiola Salau, and Kritagya Upadhyay, *Trusted ai with blockchain to empower metaverse*, 2022 Fourth International Conference on Blockchain Computing and Applications (BCCA), IEEE, 2022, pp. 237–244.

[16] Syed Badruddoja, Ram Dantu, Yanyan He, Kritagya Upadhayay, and Mark Thompson, *Making smart contracts smarter*, 2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), IEEE, 2021, pp. 1–3.

[17] Syed Badruddoja, Ram Dantu, Logan Widick, Zachary Zaccagni, and Kritagya Upadhyay, *Integrating dots with blockchain can secure massive iot sensors*, 2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), IEEE, 2020, pp. 937–946.

[18] Yavuz Selim Balcıoğlu, Melike Artar, and Oya Erdil, *The use of ai in metaverse*, Global Economic Challenges: 6th International Conference on Banking and Finance Perspectives, Cuenca, Spain, Springer, 2023, pp. 21–30.

[19] Natalie M Banta, *Property interests in digital assets: The rise of digital feudalism*, Cardozo L. Rev. 38 (2016), 1099.

[20] B Douglas Bernheim and Michael D Whinston, *Incomplete contracts and strategic ambiguity*, American Economic Review (1998), 902–932.

[21] Thomas Bittner, *Ontology, vagueness, and indeterminacy*, (2000).

[22] John B Bowles and Colon E Pelaez, *Application of fuzzy logic to reliability engineering*, Proceedings of the IEEE 83 (1995), no. 3, 435–449.

[23] Harris Brakmić and Harris Brakmić, *Bitcoin script*, Bitcoin and Lightning Network on Raspberry Pi: Running Nodes on Pi3, Pi4 and Pi Zero (2019), 201–224.

[24] Anna Broinowski, *The future is hackable: Apocalypse and euphoria in a deepfake world*, Griffith REVIEW (2023), no. 79, 9–19.

[25] Miles Brundage, Shahar Avin, Jasmine Wang, Haydn Belfield, Gretchen Krueger, Gillian Hadfield, Heidy Khlaaf, Jingying Yang, Helen Toner, Ruth Fong, et al., *Toward trustworthy ai development: mechanisms for supporting verifiable claims*, arXiv preprint arXiv:2004.07213 (2020).

[26] Vitalik Buterin et al., *A next-generation smart contract and decentralized application platform*, white paper 3 (2014), no. 37, 2–1.

[27] Giulio Caldarelli, *Understanding the blockchain oracle problem: A call for action*, Information 11 (2020), no. 11, 509.

[28] Bo Carlsson, *The digital economy: what is new and what is not?*, Structural change and economic dynamics 15 (2004), no. 3, 245–264.

[29] Anthony J Casey and Anthony Niblett, *Self-driving contracts*, J. Corp. L. 43 (2017), 1.

[30] John "Jack" Castonguay and Sean Stein Smith, *Digital assets and blockchain: Hackable, fraudulent, or just misunderstood?*, Accounting Perspectives 19 (2020), no. 4, 363–387.

[31] Irina Ceaparu, Jonathan Lazar, Katie Bessiere, John Robinson, and Ben Shneiderman, *Determining causes and severity of end-user frustration*, International journal of human-computer interaction 17 (2004), no. 3, 333–356.

[32] Federico Cernera, Massimo La Morgia, Alessandro Mei, and Francesco Sassi, *Token*

spammers, rug pulls, and sniperbots: An analysis of the ecosystem of tokens in ethereum and the binance smart chain (bnb), arXiv preprint arXiv:2206.08202 (2022).

[33] Sharmila Chackravarthy, Steven Schmitt, and Li Yang, *Intelligent crime anomaly detection in smart cities using deep learning*, 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC), IEEE, 2018, pp. 399–404.

[34] Viswanath Chadalapaka, Kyle Chang, Gireesh Mahajan, and Anuj Vasil, *Crypto pump and dump via deep learning techniques*, arXiv preprint arXiv:2205.04646 (2022).

[35] Snehashish Chakraverty, Deepti Moyi Sahoo, Nisha Rani Mahato, Snehashish Chakraverty, Deepti Moyi Sahoo, and Nisha Rani Mahato, *Defuzzification*, Concepts of Soft Computing: Fuzzy and ANN with Programming (2019), 117–127.

[36] Shyi-Ming Chen, *A weighted fuzzy reasoning algorithm for medical diagnosis*, Decision support systems 11 (1994), no. 1, 37–43.

[37] Ting Chen, Xiaoqi Li, Xiapu Luo, and Xiaosong Zhang, *Under-optimized smart contracts devour your money*, 2017 IEEE 24th international conference on software analysis, evolution and reengineering (SANER), IEEE, 2017, pp. 442–446.

[38] Xusen Cheng, Shuang Zhang, Shixuan Fu, Wanxin Liu, Chong Guan, Jian Mou, Qiongwei Ye, and Caiming Huang, *Exploring the metaverse in the digital economy: an overview and research framework*, Journal of Electronic Business & Digital Economics (2022), no. ahead-of-print.

[39] Albert Choi and George Triantis, *Strategic vagueness in contract design: The case of corporate acquisitions*, The Yale Law Journal (2010), 848–924.

[40] Wujin Chu, Eitan Gerstner, and James D Hess, *Managing dissatisfaction: How to decrease customer opportunism by partial refunds*, Journal of Service Research 1 (1998), no. 2, 140–155.

[41] Stephanie Hui-Wen Chuah, *Why and who will adopt extended reality technology? literature review, synthesis, and future research agenda*, Literature Review, Synthesis, and Future Research Agenda (December 13, 2018) (2018).

[42] Christopher D Clack, *Smart contract templates: legal semantics and code validation*,

Journal of Digital Banking 2 (2018), no. 4, 338–352.

[43] Morris R Cohen, *The basis of contract*, Harv. L. Rev. 46 (1932), 553.

[44] Corinna Cortes and Vladimir Vapnik, *Support-vector networks*, Machine learning 20 (1995), 273–297.

[45] James A Craft, *Unions, bureaucracy, and change: Old dogs learn new tricks very slowly.*, Journal of Labor Research 12 (1991), no. 4, 393–405.

[46] Robert Dale, *Gpt-3: What's it good for?*, Natural Language Engineering 27 (2021), no. 1, 113–118.

[47] Chris Dannen, *Introducing ethereum and solidity*, vol. 1, Springer, 2017.

[48] Aspassia Daskalopulu and Tom Maibaum, *Towards electronic contract performance*, 12th International Workshop on Database and Expert Systems Applications, IEEE, 2001, pp. 771–777.

[49] Amanda Davenport, Sachin Shetty, and Xueping Liang, *Attack surface analysis of permissioned blockchain platforms for smart cities*, 2018 IEEE International Smart Cities Conference (ISC2), IEEE, 2018, pp. 1–6.

[50] Alex De Vries, *Cryptocurrencies on the road to sustainability: Ethereum paving the way for bitcoin*, Patterns 4 (2023), no. 1.

[51] Phyllis M Deane, *The first industrial revolution*, Cambridge University Press, 1979.

[52] Amelia J Delic and Paul H Delfabbro, *Profiling the potential risks and benefits of emerging "play to earn" games: a qualitative analysis of players' experiences with axie infinity*, International Journal of Mental Health and Addiction (2022), 1–14.

[53] Sahraoui Dhelim, Tahar Kechadi, Liming Chen, Nyothiri Aung, Huansheng Ning, and Luigi Atzori, *Edge-enabled metaverse: The convergence of metaverse and mobile edge computing*, arXiv preprint arXiv:2205.02764 (2022).

[54] John David N Dionisio, William G Burns III, and Richard Gilbert, *3d virtual worlds and the metaverse: Current status and future possibilities*, ACM Computing Surveys (CSUR) 45 (2013), no. 3, 1–38.

[55] Haihan Duan, Jiaye Li, Sizheng Fan, Zhonghao Lin, Xiao Wu, and Wei Cai, *Meta-*

verse for social good: A university campus prototype, Proceedings of the 29th ACM international conference on multimedia, 2021, pp. 153–161.

[56] Gregory M Duhl, *The ethics of contract drafting*, Lewis & Clark L. Rev. 14 (2010), 989.

[57] Antonı n Dvořák and Vilém Novák, *Formal theories and linguistic descriptions*, Fuzzy Sets and Systems 143 (2004), no. 1, 169–188.

[58] Elad Elrom and Elad Elrom, *Neo blockchain and smart contracts*, The Blockchain Developer: A Practical Guide for Designing, Implementing, Publishing, Testing, and Securing Distributed Blockchain-based Projects (2019), 257–298.

[59] Timothy Endicott, *The value of vagueness*, Vagueness in normative texts 23 (2005), 27–48.

[60] Sunil Erevelles, Shuba Srinivasan, and Steven Rangel, *Consumer satisfaction for internet service providers: an analysis of underlying processes*, Information Technology and Management 4 (2003), no. 1, 69–89.

[61] Ahmad Firdaus, Mohd Faizal Ab Razak, Ali Feizollah, Ibrahim Abaker Targio Hashem, Mohamad Hazim, and Nor Badrul Anuar, *The rise of "blockchain": bibliometric analysis of blockchain study*, Scientometrics 120 (2019), 1289–1331.

[62] Darren Paul Fisher and James R Birt, *The metaverse has been heavily hyped–but it could enable entirely new ways of screen production*, The Conversation (2022).

[63] Peter Fisher, *Boolean and fuzzy regions*, Geographic objects with indeterminate boundaries, CRC Press, 2020, pp. 87–94.

[64] Mark D Flood and Oliver R Goodenough, *Contract as automaton: The computational representation of financial agreements*, Office of Financial Research Working Paper (2017), no. 15-04.

[65] Philippe Fortemps and Marc Roubens, *Ranking and defuzzification methods based on area compensation*, Fuzzy sets and systems 82 (1996), no. 3, 319–330.

[66] Philipp Frauenthaler, Marten Sigwart, Christof Spanring, Michael Sober, and Stefan Schulte, *Eth relay: A cost-efficient relay for ethereum-based blockchains*, 2020 IEEE

International Conference on Blockchain (Blockchain), IEEE, 2020, pp. 204–213.

[67] Cloudset Solutions from Coherence Design [Online], *Crowdfunding contract by cloudset solutions from coherence design [accessed: 10-nov-2019]*, `https://support.cloudset.net/hc/en-us/articles/201367433`, November 2019.

[68] Thippa Reddy Gadekallu, Thien Huynh-The, Weizheng Wang, Gokul Yenduri, Pasika Ranaweera, Quoc-Viet Pham, Daniel Benevides da Costa, and Madhusanka Liyanage, *Blockchain for the metaverse: A review*, arXiv preprint arXiv:2203.09738 (2022).

[69] Anne von der Lieth Gardner, *An artificial intelligence approach to legal reasoning*, MIT press, 1987.

[70] Jon M Garon, *Legal implications of a ubiquitous metaverse and a web3 future*, Marq. L. Rev. 106 (2022), 163.

[71] Nicola Gennaioli and Giacomo AM Ponzetto, *Optimally vague contracts and the law*, (2017).

[72] Mark Giancaspro, *Is a 'smart contract'really a smart idea? insights from a legal perspective*, Computer law & security review 33 (2017), no. 6, 825–835.

[73] John Yukio Gotanda, *Awarding costs and attorneys' fees in international commercial arbitrations*, Mich. J. Int'l L. 21 (1999), 1.

[74] James Grimmelmann, *All smart contracts are ambiguous*, JL & Innovation 2 (2019), 1.

[75] Daojing He, Zhi Deng, Yuxing Zhang, Sammy Chan, Yao Cheng, and Nadra Guizani, *Smart contract vulnerability analysis and security audit*, IEEE Network 34 (2020), no. 5, 276–282.

[76] Sherry He, Brett Hollenbeck, and Davide Proserpio, *The market for fake reviews*, Marketing Science 41 (2022), no. 5, 896–921.

[77] Benjamin E Hermalin, *Vague terms: Contracting when precision in terms is infeasible*, Journal of Institutional and Theoretical Economics (JITE)/Zeitschrift für die gesamte Staatswissenschaft (2008), 76–94.

[78] Everett Hildenbrandt, Manasvi Saxena, Nishant Rodrigues, Xiaoran Zhu, Philip Daian,

Dwight Guth, Brandon Moore, Daejun Park, Yi Zhang, Andrei Stefanescu, et al., *Kevm: A complete formal semantics of the ethereum virtual machine*, 2018 IEEE 31st Computer Security Foundations Symposium (CSF), IEEE, 2018, pp. 204–217.

[79] Andrew N Hiles, *Service level agreements: panacea or pain?*, The TQM Magazine (1994).

[80] Henry Hon, Kevin Wang, and Michael Bolger, *Research and insights*.

[81] Anthony Hunter and Sébastien Konieczny, *Approaches to measuring inconsistent information*, Inconsistency tolerance (2005), 191–236.

[82] Numaan Huq, Roel Reyes, Philippe Lin, and Morton Swimmer, *Cybersecurity threats against the internet of experiences*, Trend Micro Res., TX, USA (2022).

[83] Wook Hyun, *Study on standardization for interoperable metaverse*, 2023 25th International Conference on Advanced Communication Technology (ICACT), IEEE, 2023, pp. 319–322.

[84] Florian Idelberger, Guido Governatori, Régis Riveret, and Giovanni Sartor, *Evaluation of logic-based smart contracts for blockchain systems*, Rule Technologies. Research, Tools, and Applications: 10th International Symposium, RuleML 2016, Stony Brook, NY, USA, July 6-9, 2016. Proceedings 10, Springer, 2016, pp. 167–183.

[85] Shashank Mohan Jain, *Introduction to remix ide*, A Brief Introduction to Web3: Decentralized Web Fundamentals for App Development, Springer, 2022, pp. 89–126.

[86] J Betty Jane and EN Ganesh, *A review on big data with machine learning and fuzzy logic for better decision making*, International Journal of Scientific & Technology Research 8 (2019), no. 10, 1121–1125.

[87] Martin Janicke and Klaus Jacob, *A third industrial revolution*, Long-term governance for social-ecological change (2013), 47–71.

[88] Heejeong Jeong, Youkyoung Yi, and Dongsoo Kim, *An innovative e-commerce platform incorporating metaverse to live commerce*, International Journal of Innovative Computing, Information and Control 18 (2022), no. 1, 221–229.

[89] Sandra Johnson, Peter Robinson, and John Brainard, *Sidechains and interoperability*,

arXiv preprint arXiv:1903.04077 (2019).

[90] Harvey Jones and José Hiram Soltren, *Facebook: Threats to privacy*, Project MAC: MIT project on mathematics and computing 1 (2005), no. 01, 2005.

[91] Jamal MM Joudeh and Ala'O Dandis, *Service quality, customer satisfaction and loyalty in an internet service providers*, International Journal of Business and Management 13 (2018), no. 8, 108–120.

[92] Elie Kapengut and Bruce Mizrach, *An event study of the ethereum transition to proof-of-stake*, Commodities 2 (2023), no. 2, 96–110.

[93] Ioannis Karamitsos, Maria Papadaki, and Nedaa Baker Al Barghuthi, *Design of the blockchain smart contract: A use case for real estate*, Journal of Information Security 9 (2018), no. 3, 177–190.

[94] Nilesh Naval Karnik, Jerry M Mendel, and Qilian Liang, *Type-2 fuzzy logic systems*, IEEE transactions on Fuzzy Systems 7 (1999), no. 6, 643–658.

[95] M Kaur and B Gupta, *Metaverse technology and the current market*, National Institute of Technology Kurukshetra (2021).

[96] Rochelle Klempner, *The case for court-based document assembly programs: A review of the new york state court system's diy forms*, Fordham Urb. LJ 41 (2013), 1189.

[97] MV Koroteev, *Bert: a review of applications in natural language processing and understanding*, arXiv preprint arXiv:2103.11943 (2021).

[98] Sotiris B Kotsiantis, *Decision trees: a recent overview*, Artificial Intelligence Review 39 (2013), 261–283.

[99] P Krishna, Kamalakar Karlapalem, and A Dani, *From contracts to e-contracts: Modeling and enactment.*, Information Technology & Management 6 (2005), no. 4.

[100] Shlomit Labin and Uri Segal, *Ai-driven contract review: A product development journey*, Research Handbook on Big Data Law, Edward Elgar Publishing, 2021, pp. 454–466.

[101] Kashif Laeeq, *Metaverse: why, how and what*, How and What (2022).

[102] Pablo Lamela Seijas, Alexander Nemish, David Smith, and Simon Thompson, *Marlowe:*

implementing and analysing financial contracts on blockchain, Financial Cryptography and Data Security: FC 2020 International Workshops, AsiaUSEC, CoDeFi, VOTING, and WTSC, Kota Kinabalu, Malaysia, February 14, 2020, Revised Selected Papers 24, Springer, 2020, pp. 496–511.

[103] Stephen Martin Leake, *The elements of the law of contracts*, Stevens & Sons, 1867.

[104] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, *Deep learning*, nature 521 (2015), no. 7553, 436–444.

[105] Lik-Hang Lee, Tristan Braud, Pengyuan Zhou, Lin Wang, Dianlei Xu, Zijun Lin, Abhishek Kumar, Carlos Bermejo, and Pan Hui, *All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda*, arXiv preprint arXiv:2110.05352 (2021).

[106] Wei-Meng Lee and Wei-Meng Lee, *Testing smart contracts using ganache*, Beginning Ethereum Smart Contracts Programming: With Examples in Python, Solidity, and JavaScript (2019), 147–167.

[107] Wei-Meng Lee and W.M. Lee, *Using the metamask chrome extension*, Beginning Ethereum Smart Contracts Programming: With Examples in Python, Solidity, and JavaScript (2019), 93–126.

[108] Arthur Allen Leff, *Contract as thing*, Am. UL Rev. 19 (1970), 131.

[109] Kai Li, Yingping Cui, Weicai Li, Tiejun Lv, Xin Yuan, Shenghong Li, Wei Ni, Meryem Simsek, and Falko Dressler, *When internet of things meets metaverse: Convergence of physical and cyber worlds*, arXiv preprint arXiv:2208.13501 (2022).

[110] Shuangling Li, *A corpus-based study of vague language in legislative texts: Strategic use of vague terms*, English for Specific Purposes 45 (2017), 98–109.

[111] John Linarelli, *Advanced artificial intelligence and contract*, Forthcoming, Uniform Law Review (Special Issues on Transnational Commercial Law and the Technology/Digital Economy (2019).

[112] Richard TB Ma, Dah-ming Chiu, John CS Lui, Vishal Misra, and Dan Rubenstein, *On cooperative settlement between content, transit and eyeball internet service providers*,

Proceedings of the 2008 ACM CoNEXT Conference, 2008, pp. 1–12.

[113] Jacob Mallet, Rushit Dave, Naeem Seliya, and Mounika Vanamala, *Using deep learning to detecting deepfakes*, arXiv preprint arXiv:2207.13644 (2022).

[114] Ebrahim H Mamdani, *Application of fuzzy algorithms for control of simple dynamic plant*, Proceedings of the institution of electrical engineers, vol. 121, IET, 1974, pp. 1585–1588.

[115] Alex B McBratney and Inakwu OA Odeh, *Application of fuzzy sets in soil science: fuzzy logic, fuzzy measurements and fuzzy decisions*, Geoderma 77 (1997), no. 2-4, 85–113.

[116] Thomas J McCabe, *A complexity measure*, IEEE Transactions on software Engineering (1976), no. 4, 308–320.

[117] Muhammad Izhar Mehar, Charles Louis Shier, Alana Giambattista, Elgar Gong, Gabrielle Fletcher, Ryan Sanayhie, Henry M Kim, and Marek Laskowski, *Understanding a revolutionary and flawed grand experiment in blockchain: the dao attack*, Journal of Cases on Information Technology (JCIT) 21 (2019), no. 1, 19–32.

[118] Shahab Mohaghegh, *Virtual-intelligence applications in petroleum engineering: part 3—fuzzy logic*, Journal of petroleum technology 52 (2000), no. 11, 82–87.

[119] Bhabendu Kumar Mohanta, Soumyashree S Panda, and Debasish Jena, *An overview of smart contract and use cases in blockchain technology*, 2018 9th international conference on computing, communication and networking technologies (ICCCNT), IEEE, 2018, pp. 1–4.

[120] Hossein Mohit and Vess L Johnson, *Coin or token: Different motivations for investing in cryptocurrency*, (2022).

[121] Joel Mokyr and Robert H Strotz, *The second industrial revolution, 1870-1914*, Storia dell'economia Mondiale 21945 (1998), no. 1.

[122] Emiliano Monteiro, Rodrigo Righi, Rafael Kunst, Cristiano da Costa, and Dhananjay Singh, *Combining natural language processing and blockchain for smart contract generation in the accounting and legal field*, Intelligent Human Computer Interaction: 12th

International Conference, IHCI 2020, Daegu, South Korea, November 24–26, 2020, Proceedings, Part I 12, Springer, 2021, pp. 307–321.

[123] Satoshi Nakamoto, *Bitcoin whitepaper*, URL: https://bitcoin. org/bitcoin. pdf-(: 17.07. 2019) (2008).

[124] Aslihan Nasir, *E-consumer complaints about on-line stores*, The Journal of Consumer Satisfaction, Dissatisfaction and Complaining Behavior 17 (2004), 68–87.

[125] Trung-Viet Nguyen, Lam-Son Lê, Bo Dao, and Khuong Nguyen-An, *Leveraging blockchain in monitoring sla-oriented tourism service provisioning*, 2019 International Conference on Advanced Computing and Applications (ACOMP), IEEE, 2019, pp. 42–50.

[126] Y-son Nguyên, *Artificial intelligence contract: How algorithms and machines have disrupted the way law is practices*, PM World Journal 8 (2019), no. 9.

[127] Michael Nofer, Peter Gomber, Oliver Hinz, and Dirk Schiereck, *Blockchain*, Business & Information Systems Engineering 59 (2017), 183–187.

[128] Silas Nzuva, *Smart contracts implementation, applications, benefits, and limitations*, Journal of Information Engineering and Applications 9 (2019), no. 5, 63–75.

[129] TP Oliphant and P Jones, *Scipy-skfuzzy*, URL: https://pythonhosted. org/scikitfuzzy (2020).

[130] AT&T Service-Level Agreement [Online], *At&t switched ethernet service guide-service-level agreement [accessed: 30-oct-2020]*, `https://cpr.att.com/pdf/se/0001-0003.pdf`, October 2020.

[131] CenturyLink Service-Level Agreement [Online], *Centurylink fiber-service-level agreement [accessed: 30-oct-2020]*, `http://internethelp.centurylink.com/legal/docs/CenturyLink-Ethernet-SLA.pdf`, October 2020.

[132] Employment Contract [Online], *Employment contract [accessed: 21-nov-2019]*, `https://www.template.net/business/effective-employment-agreements/`, November 2019.

[133] Spectrum Service-Level Agreement [Online], *Spectrum-service-level agreement [ac-*

cessed: 30-oct-2020], https://spectruminternet.com/wp-content/uploads/2017/02/Spectrum-SLA-Business-Broadband.pdf, October 2020.

[134] T-Mobile Service-Level Agreement [Online], *T-mobile-service-level agreement [accessed: 30-oct-2020]*, https://www.t-mobile.com/responsibility/legal/terms-and-conditions, October 2020.

[135] Verizon Service-Level Agreement [Online], *Verizon-service-level agreement [accessed: 30-oct-2020]*, https://enterprise.verizon.com/serviceguide/reg/cpmwansla.pdf, October 2020.

[136] Ziply Fiber Service-Level Agreement [Online], *Ziply fiber-service-level agreement [accessed: 30-oct-2020]*, https://ziplyfiber.com/media/corporate/terms/sla.ashx?la=en&la=en, October 2020.

[137] Esteban Ordano, Ariel Meilich, Yemel Jardi, and Manuel Araoz, *Decentraland: A blockchain-based virtual world*, Decentraland, White Paper (2017).

[138] Terence Parsons, *Modifiers and quantifiers in natural language*, Canadian Journal of Philosophy Supplementary Volume 6 (1980), 29–60.

[139] Witold Pedrycz, *Why triangular membership functions?*, Fuzzy sets and Systems 64 (1994), no. 1, 21–30.

[140] Thu Nguyen Quach, Charles Jebarajakirthy, and Park Thaichon, *The effects of service quality on internet service provider customers' behaviour: A mixed methods study*, Asia Pacific Journal of Marketing and Logistics (2016).

[141] Ofer Raban, *The fallacy of legal certainty: Why vague legal standards may be better for capitalism and liberalism*, BU Pub. Int. LJ 19 (2009), 175.

[142] David N Rapp, *The consequences of reading inaccurate information*, Current Directions in Psychological Science 25 (2016), no. 4, 281–285.

[143] Digvijaysinh Rathod, *Web browser forensics: google chrome*, International Journal of Advanced Research in Computer Science 8 (2017), no. 7, 896–899.

[144] Danda B Rawat, Vijay Chaudhary, and Ronald Doku, *Blockchain: Emerging applications and use cases*, arXiv preprint arXiv:1904.12247 (2019).

[145] Teófilo Redondo, *The digital economy: Social interaction technologies–an overview*, (2015).

[146] Damien Rolon-Mérette, Matt Ross, Thaddé Rolon-Mérette, and Kinsey Church, *Introduction to anaconda and python: Installation and setup*, Quant. Methods Psychol 16 (2016), no. 5, S3–S11.

[147] Edward L Rubin, *Types of contracts, interventions of law*, Wayne L. Rev. 45 (1999), 1903.

[148] Abiola Salau, Ram Dantu, and Kritagya Upadhyay, *Data cooperatives for neighborhood watch*, 2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), IEEE, 2021, pp. 1–9.

[149] Sarwar Sayeed, Hector Marco-Gisbert, and Tom Caira, *Smart contract: Attacks and protections*, IEEE Access 8 (2020), 24416–24427.

[150] Eder J Scheid, Bruno B Rodrigues, Lisandro Z Granville, and Burkhard Stiller, *Enabling dynamic sla compensation using blockchain-based smart contracts*, 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), IEEE, 2019, pp. 53–61.

[151] Dudi Schwartz, *Interpretation and disclosure in insurance contracts*, Loy. Consumer L. Rev. 21 (2008), 105.

[152] Claude Elwood Shannon, *A mathematical theory of communication*, ACM SIGMOBILE mobile computing and communications review 5 (2001), no. 1, 3–55.

[153] Edward Shortliffe, *Computer-based medical consultations: Mycin*, vol. 2, Elsevier, 2012.

[154] James Skene, D Davide Lamanna, and Wolfgang Emmerich, *Precise service level agreements*, Proceedings. 26th International Conference on Software Engineering, IEEE, 2004, pp. 179–188.

[155] Puli Sreehari, M Nandakishore, Goutham Krishna, Joshin Jacob, and VS Shibu, *Smart will converting the legal testament into a smart contract*, 2017 International Conference on Networks & Advances in Computational Technologies (NetACT), IEEE, 2017, pp. 203–207.

[156] Neal Stephenson, *Snow crash: A novel*, Spectra, 2003.

[157] Michio Sugeno and Takahiro Yasukawa, *A fuzzy-logic-based approach to qualitative modeling*, IEEE Transactions on fuzzy systems 1 (1993), no. 1, 7.

[158] Harry Surden, *Computable contracts*, UCDL Rev. 46 (2012), 629.

[159] Nick Szabo, *Formalizing and securing relationships on public networks*, First monday (1997).

[160] Jane Thomason, *Metahealth-how will the metaverse change health care?*, Journal of Metaverse 1 (2021), no. 1, 13–16.

[161] Stefan Tilkov and Steve Vinoski, *Node. js: Using javascript to build high-performance network programs*, IEEE Internet Computing 14 (2010), no. 6, 80–83.

[162] Jack Tsai, Minda Huang, John R Blosnich, and Eric B Elbogen, *Evictions and tenant-landlord relationships during the 2020–2021 eviction moratorium in the us*, American journal of community psychology 70 (2022), no. 1-2, 117–126.

[163] © Kritagya Upadhyay, Ram Dantu, Yanyan He, Syed Badruddoja, and A. Salau, *Auditing metaverse requires multimodal deep learning*, 2022 IEEE 4th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications (TPS-ISA), IEEE, 2022, pp. 39–46.

[164] © Kritagya Upadhyay, Ram Dantu, Yanyan He, Syed Badruddoja, and Abiola Salau, *Can't understand slas? use the smart contract*, 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), IEEE, 2021, pp. 129–136.

[165] © Kritagya Upadhyay, Ram Dantu, Yanyan He, Abiola Salau, and S. Badruddoja, *Paradigm shift from paper contracts to smart contracts*, 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), IEEE, 2021, pp. 261–268.

[166] © Kritagya Upadhyay, Ram Dantu, Yanyan He, Abiola Salau, and Syed Badruddoja, *Make consumers happy by defuzzifying the service level agreements*, 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and

Applications (TPS-ISA), IEEE, 2021, pp. 98–105.

[167] © Kritagya Upadhyay, Ram Dantu, Zachary Zaccagni, and Syed Badruddoja, *Is your legal contract ambiguous? convert to a smart legal contract*, 2020 IEEE International Conference on Blockchain (Blockchain), IEEE, 2020, pp. 273–280.

[168] Rafael Brundo Uriarte, Rocco De Nicola, and Kyriakos Kritikos, *Towards distributed sla management with smart contracts and blockchain*, 2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), IEEE, 2018, pp. 266–271.

[169] Rafael Brundo Uriarte, Huan Zhou, Kyriakos Kritikos, Zeshun Shi, Zhiming Zhao, and Rocco De Nicola, *Distributed service-level agreement management with smart contracts and blockchain*, Concurrency and Computation: Practice and Experience 33 (2021), no. 14, e5800.

[170] Turgay Arda Usman, Ali Aydın Selçuk, and Süleyman Özarslan, *An analysis of ethereum smart contract vulnerabilities*, 2021 International Conference on Information Security and Cryptology (ISCTURKEY), IEEE, 2021, pp. 99–104.

[171] Katarina Valaskova, Veronika Machova, and Elizabeth Lewis, *Virtual marketplace dynamics data, spatial analytics, and customer engagement tools in a real-time interoperable decentralized metaverse*, Linguistic and Philosophical Investigations 21 (2022), 105–120.

[172] Ariën J van der Wal, *Application of fuzzy logic control in industry*, Fuzzy Sets and Systems 74 (1995), no. 1, 33–41.

[173] Rajat Verma, Namrata Dhanda, and Vishal Nagar, *Application of truffle suite in a blockchain environment*, Proceedings of Third International Conference on Computing, Communications, and Cyber-Security: IC4S 2021, Springer, 2022, pp. 693–702.

[174] Ivaylo Vladimirov, Maria Nenova, Desislava Nikolova, and Zornitsa Terneva, *Security and privacy protection obstacles with 3d reconstructed models of people in applications and the metaverse: A survey*, 2022 57th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST), IEEE, 2022, pp. 1–4.

[175] Qin Wang, Rujia Li, Qi Wang, Shiping Chen, Mark Ryan, and Thomas Hardjono, *Exploring web3 from the view of blockchain*, arXiv preprint arXiv:2206.08821 (2022).

[176] Shuai Wang, Liwei Ouyang, Yong Yuan, Xiaochun Ni, Xuan Han, and Fei-Yue Wang, *Blockchain-enabled smart contracts: architecture, applications, and future trends*, IEEE Transactions on Systems, Man, and Cybernetics: Systems 49 (2019), no. 11, 2266–2277.

[177] Mika Westerlund, *The emergence of deepfake technology: A review*, Technology innovation management review 9 (2019), no. 11.

[178] Timothy Williamson, *Vagueness*, Routledge, 2002.

[179] Linlin Wu and Rajkumar Buyya, *Service level agreement (sla) in utility computing systems*, Performance and dependability in service computing: Concepts, techniques and research directions, IGI Global, 2012, pp. 1–25.

[180] Nannan Xi, Juan Chen, Filipe Gama, Marc Riar, and Juho Hamari, *The challenges of entering the metaverse: An experiment on the effect of extended reality on workload*, Information Systems Frontiers 25 (2023), no. 2, 659–680.

[181] Min Xu, Jeanne M David, Suk Hi Kim, et al., *The fourth industrial revolution: Opportunities and challenges*, International journal of financial research 9 (2018), no. 2, 90–95.

[182] Yingjie Xu, Gengran Hu, Lin You, and Chengtang Cao, *A novel machine learning-based analysis model for smart contract vulnerability*, Security and Communication Networks 2021 (2021), 1–12.

[183] Anatoly Yakovenko, *Solana: A new architecture for a high performance blockchain v0. 8.13*, Whitepaper (2018).

[184] Xue-hai Yuan, Zeng-liang Liu, and E Stanley Lee, *Center-of-gravity fuzzy systems based on normal fuzzy implications*, Computers & Mathematics with Applications 61 (2011), no. 9, 2879–2898.

[185] Zachary Zaccagni and Ram Dantu, *Proof of review (por): A new consensus protocol for deriving trustworthiness of reputation through reviews*, Cryptology ePrint Archive (2020).

[186] Lotfi Zadeh, *Fuzzy logic*, Granular, Fuzzy, and Soft Computing, Springer, 2023, pp. 19–49.

[187] Lotfi A Zadeh, *Knowledge representation in fuzzy logic*, An introduction to fuzzy logic applications in intelligent systems (1992), 1–25.

[188] Lotfi Asker Zadeh, *The concept of a linguistic variable and its application to approximate reasoning—i*, Information sciences 8 (1975), no. 3, 199–249.

[189] Lotfi Asker Zadeh, George J Klir, and Bo Yuan, *Fuzzy sets, fuzzy logic, and fuzzy systems: selected papers*, vol. 6, World scientific, 1996.

[190] Lin Zhang, Brian Lee, Yuhang Ye, and Yuansong Qiao, *Ethereum transaction performance evaluation using test-nets*, Euro-Par 2019: Parallel Processing Workshops: Euro-Par 2019 International Workshops, Göttingen, Germany, August 26–30, 2019, Revised Selected Papers 25, Springer, 2020, pp. 179–190.

[191] Shijie Zhang and Jong-Hyouk Lee, *Analysis of the main consensus protocols of blockchain*, ICT express 6 (2020), no. 2, 93–97.

[192] Zibin Zheng, Shaoan Xie, Hongning Dai, Xiangping Chen, and Huaimin Wang, *An overview of blockchain technology: Architecture, consensus, and future trends*, 2017 IEEE international congress on big data (BigData congress), Ieee, 2017, pp. 557–564.