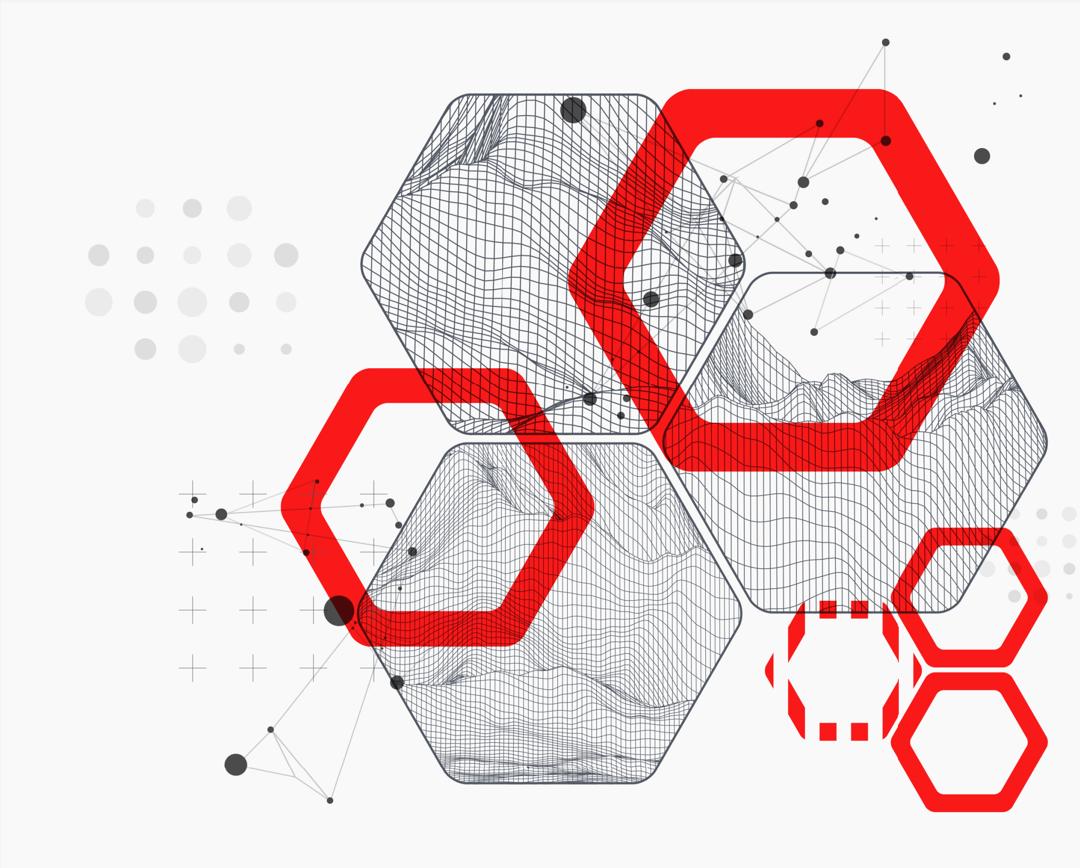
ADCAIJ

Advances in Distributed Computing and Artificial Intelligence Journal



Vol.11 N.4

ADCAIJ.USAL.ES 2022

REGULAR ISSUE

Ediciones Universidad Salamanca

ADCAIJ
Advances in Distributed Computing and Artificial Intelligence Journal

REGULAR ISSUE
Vol. 11 N. 4
2022

Ediciones Universidad Salamanca

eISBN: 2255-2863
Volume 11, number 4
BISITE Research Group
University of Salamanca, 2022

ADVANCES IN DISTRIBUTED COMPUTING AND ARTIFICIAL INTELLIGENCE
https://adcaij.usal.es

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
eISSN: 2255-2863 - DOI: https://doi.org/10.14201/adcaij2022114 - CDU: 004 -
IBIC: Computación e informática (U) - BIC: Computing & Information Technology (U) - BISAC: Computers / General (COM000000)

Regular Issue, Vol. 11, N. 4 (2022)

SCOPE

The Advances in Distributed Computing and Artificial Intelligence Journal (ADCAIJ) is an open access journal that publishes articles which contribute new results associated with distributed computing and artificial intelligence, and their application in different areas.

The artificial intelligence is changing our society. Its application in distributed environments, such as the Internet of Thing (IoT), electronic commerce, mobile communications, wireless devices, distributed computing, Big Data and so on, is increasing and becoming and element of high added value and economic potential in industry and research. These technologies are changing constantly as a result of the large research and technical effort being undertaken in both universities and businesses. The exchange of ideas between scientists and technicians from both academic and business areas is essential to facilitate the development of systems that meet the demands of today's society.

This issue will be focused on the importance of knowledge in advanced digital technologies and their involvement in the different activities of the public-private sector related to specialization in digital technologies and blockchain. The issue also includes articles focusing on research into new technologies and an in-depth look at advanced digital tools from a practical point of view in order to be able to implement them in organisations in peripheral and border areas.

We would like to thank all the contributing authors f or their hard and highly valuable work and members of 0631_DIGITEC_3_E project (Smart growth through the specialization of the cross-border business fabric in advanced digital technologies and blockchain) supported by the European Regional Development Fund (ERDF) through the Interreg Spain-Portugal V-A Program (POCTEP) . Their work has helped to contribute to the success of this issue. Finally, the Editors wish to thank Scientific Committee of Advances in Distributed Computing and Artificial Intelligence Journal for the collaboration of this issue, that notably contributes to improve the quality of the journal.

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
eISSN: 2255-2863 - DOI: https://doi.org/10.14201/adcaij2022114 - CDU: 004 -
IBIC: Computación e informática (U) - BIC: Computing & Information Technology (U) - BISAC: Computers / General (COM000000)

Regular Issue, Vol. 11, N. 4 (2022)

INDEX

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 11 N. 4 (2022), 395-409
eISSN: 2255-2863
DOI: https://doi.org/10.14201/adcaij.28351

Containerization and its Architectures: A Study

Satya Bhushan Verma[a], Brijesh Pandey[b], and Bineet Kumar Gupta[C]

[a,c] Shri Ramswaroop Memorial University, Barabanki, India
[b] Goel Institute of Technology & Management, Lucknow, India
[a] satyabverma1@gmail.com, [b] brijesh84academics@gmail.com, [c] bkguptacs@gmail.com

KEYWORDS

Cloud Computing; storage; containerization; Docker; LXC

ABSTRACT

Containerization is a technique for the lightweight virtualization of programmes in cloud computing, which leads to the widespread use of cloud computing. It has a positive impact on both the development and deployment of software. Containers can be divided into two groups based on their setup. The Application Container and the System Container are two types of containers. A container is a user-space that is contained within another container, while a system container is a user-space that is contained within another container. This study compares and contrasts several container architectures and their organisation in micro-hosting environments for containers.

1. Introduction

Containerization is a method for the lightweight virtualization of programs, which contributes to the widespread adoption of cloud computing. As described in C. Pahl et al., orchestrating and deploying containers separately and in groups has been a significant issue (C. Pahl 2015). Numerous studies on container knowledge in the cloud have been conducted to detect trends, knowledge gaps, and future directions. The studies provide a comparative picture of the status of research by identifying, classifying, and synthesizing the available data. The systematic mapping study (SMS) was used in this research to help define and structure new topics of inquiry.

Both the development and deployment processes benefit from the use of containers. For example, cloud architecture is evolving toward DevOps approaches, allowing for a continuous advancement and distribution pipeline based on containers and orchestration that incorporates cloud-native architectural results (Brunnert et al., 2015).

According to the findings, containers may offer continuous progress in the cloud when employed with cloud-native platform progress and distribution facilities, however, they require sophisticated orchestration provision. Thus, orchestration methods based on containers emerge as a tool for orchestrating compute in cloud-based, grouped systems. The findings of this research indicate that such methods are realized to strike an equilibrium between the necessity for reasonable quality control, e.g., optimal source usage and performance, which is a price issue in the cloud (due to its utility assessing code).

This research article aims to assist researchers working in software engineering, dispersed arrangements, and cloud computing. A systematic research presentation establishes a knowledge foundation on which to build theories and explanations, evaluate study consequences, and define future scopes. Additionally, it helps practitioners who engage in this activity by using existing tools and technology.

As virtualization enables the virtualization of computer, storage, and networking components, it is the foundation of Cloud computing. Through the use of a specialized software layer known as a hypervisor, virtualization enables the creation of several separate execution environments known as Virtual Machines (VM) that can be easily moved from one Cloud to another. KVM, Xen, WM-Ware, Virtual Box, and Virtual PC are some hypervisor virtualization software solutions. Linux Container Virtualization has recently emerged as a lightweight alternative to HVV (LCV). This technique divides the actual resources of a computer into several separate user-space instances. Docker, Kubernetes, OpenVZ, and Virtuozzo are just a few examples of LCV software solutions.

The primary distinction between HVV and LCV is that HVV abstracts the visitor operating systems, while LCV bonds a single OS kernel across containers. In a nutshell, HVV establishes a computer hardware concept layer, while LCV utilizes system calls. Containers and virtual machines are equivalent from the user's perspective. Additionally, since LCV consumes fewer hardware resources than HVV, it enables more containers on a single physical host than VMs.

1.1. Container Configurations

Containers are available in two distinct configurations.

1.1.1. Application Container: An individual container is purpose-built to execute a single kind of application;

1.1.2. System Container: A user-space is contained inside another container. LCV enables greater adaptability in terms of service design, optimization, and administration.

The following section compares HVV with LCV.

- Start-Up-Time. For the start-up phase, LCV is quicker than HVV.
- Dynamic-Runtime Control. LCV enables the host OS to start or stop a containerized program, while HVV usually needs the host OS to establish a virtual network/serial connection.
- Speed. Only experiments conducted under particular conditions are capable of determining the latency and throughput overhead.
- Isolation. In comparison to LCV, HVV has a high level of isolation.
- Flash Memory Consumption. LCV enables the OS core and other portions of the user's environment, while HVV does not allow sharing due to separated VM images.
- Assigning or separating resources dynamically. Both solutions are capable of implementing this functionality.
- Direct computer hardware access. In contrast to HVV, LCV needs specialized drivers in the host OS core to entree the device's computer peripheral. We conclude that, on the basis of this study, LCV triumphs over HVV.

1.2. Responsibilities for Resource Supervision on the Internet of Things-Cloud

IoT Cloud service providers can employ both HVV and LCV models to deliver their services. The Internet of Things devices, which run custom software, are connected to a Cloud stage that can manage heterogeneous identifying data from several sensors. Due of its limited computational capability, an IoT device typically only runs primitive programmes for detecting and actuating. A Cloud data centre, on the other hand, conducts heavy computational tasks such as storing and processing sensor data. Several projects have looked into LCV-based applications for IoT devices to take advantage of both HVV and LCV benefits. Figure 1 depicts an IoT Cloud that makes use of both HVV and LCV knowledge. The structure of the Internet of Things is in charge of the following:

- Instantiation;
- Guaranteeing consistency, QoS, safety, and scalability;
- Supervision and optimization of IoTaaS.

Additionally, since IoT devices include LCV capabilities, IoTaaS may be deployed in many dispersed containers. The information collected from IoT gadgets is standardized and saved on the Cloud stage, activating actuators linked to IoT gadgets. HVV and LCV create a film of abstraction that conceals all corporal proficiencies. Typically, IoT cloud workers provide both patterns, but only LCV is utilized in IoT devices. To dynamically offer services to their customers, the operators of IoT clouds

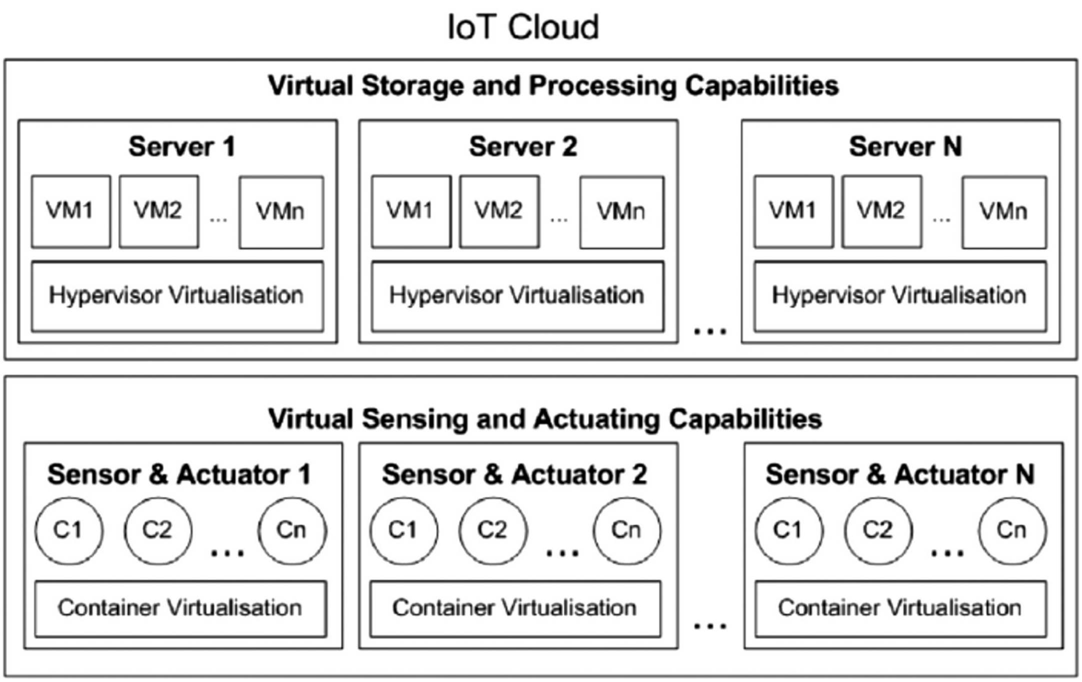


Figure. 1. A demonstration of an Internet of Things cloud that makes use of both HVV and LCV virtualization know-how (Celesti et al. 2019).

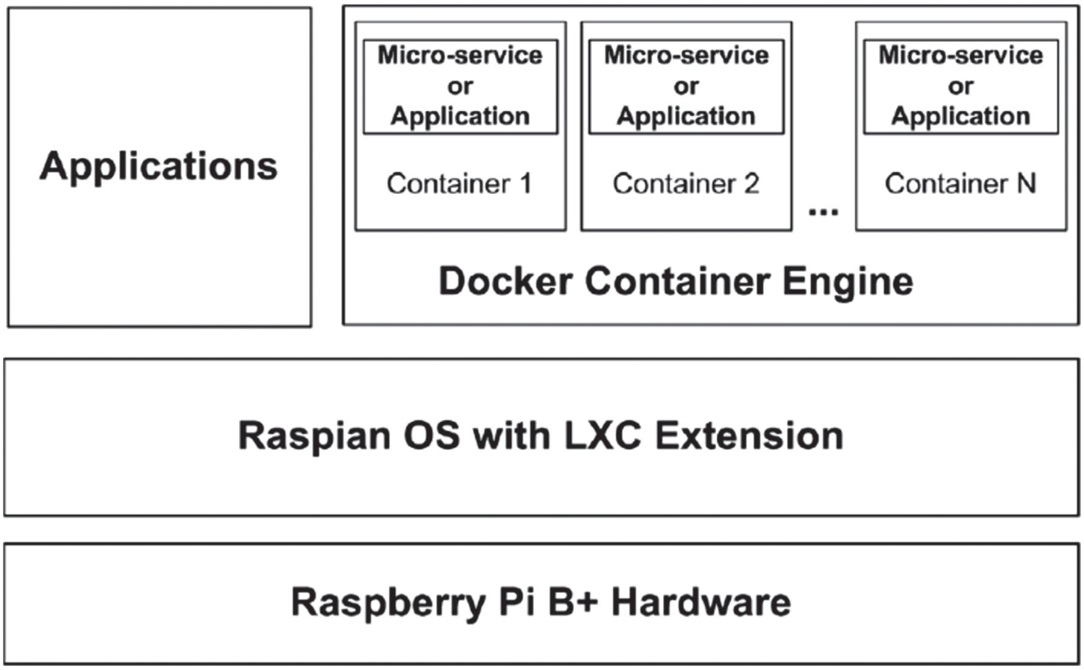


Figure. 2. Container virtualization software planning based on Docker and running on a Raspberry Pi B+ board Antonio (Celesti et al. 2019).

install a variety of containers and virtual machines (VMs) in their infrastructure. It enables the worker to reorganize, improve, and move its computer-generated resources. The operator of the IoT Cloud may fulfil any service allocation request made by its customers by using this infrastructure. IoT Clouds may execute various activities due to LCV functional to SBCs (Single Board Computers).

- Deployment of Dispersed IoTaaS. It is probable to configure dispersed IoTaaS by combining containers installed on a variety of IoT gadgets.
- IoTaaS Repositioning and Optimization. An IoT Cloud may move a container from one IoT gadget to another in order to implement load balancing techniques;
- IoTaaS Consolidation. Using a location-aware IoTaaS constructed by merging the containers deployed on various IoT devices makes it feasible to impose software consolidation methods based on the application logic. This opens up new revenue streams for IoT Cloud workers in the areas of cost-effective asset movement and optimization, energy conservation, and on-demand resource supply.

2. Container Architectures and their Organization

Virtualization is used in cloud computing to provide elasticity in large-scale shared resources. At the infrastructure layer, virtual machines (VMs) usually serve as the backbone. By contrast, containerization enables lightweight virtualization by customizing containers as application correspondences from separate images (often obtained from an image source) that use fewer properties and time. Additionally, they enable more interoperable application packing, which is required for cloud-based moveable and interoperable package applications. Containerization is built to create, test, and deploy programs over a distributed network of computers and link these containers. Containers, therefore, solve Cloud PaaS-related issues. Given the Cloud's overall significance, it is critical to have a consolidated picture of ongoing operations.

2.1. Basis of Container Technology

Container stores that are packed, self-reliant, ready-to-distribute requests and, if essential, middleware and commercial logic (in the form of binaries and libraries), are required to execute applications. Container-based apparatuses such as Docker are based on container machines, which serve as moveable containers for programs. As a consequence, in multi-tier systems, it is necessary to handle container dependencies. In a tiered plan, an orchestration plan may specify components, their needs, and their lifetime. After that, a PaaS cloud may execute the processes through mediators (such as a container appliance). As a result, PaaS clouds may enable the placement of container-based applications. Their coordinated development, deployment, and continuing administration are referred to as orchestration in this context.

Numerous container systems are built on the Linux LXC framework. Current Linux versions, part of the Linux container scheme LXC, have core features such as namespaces and groups that allow

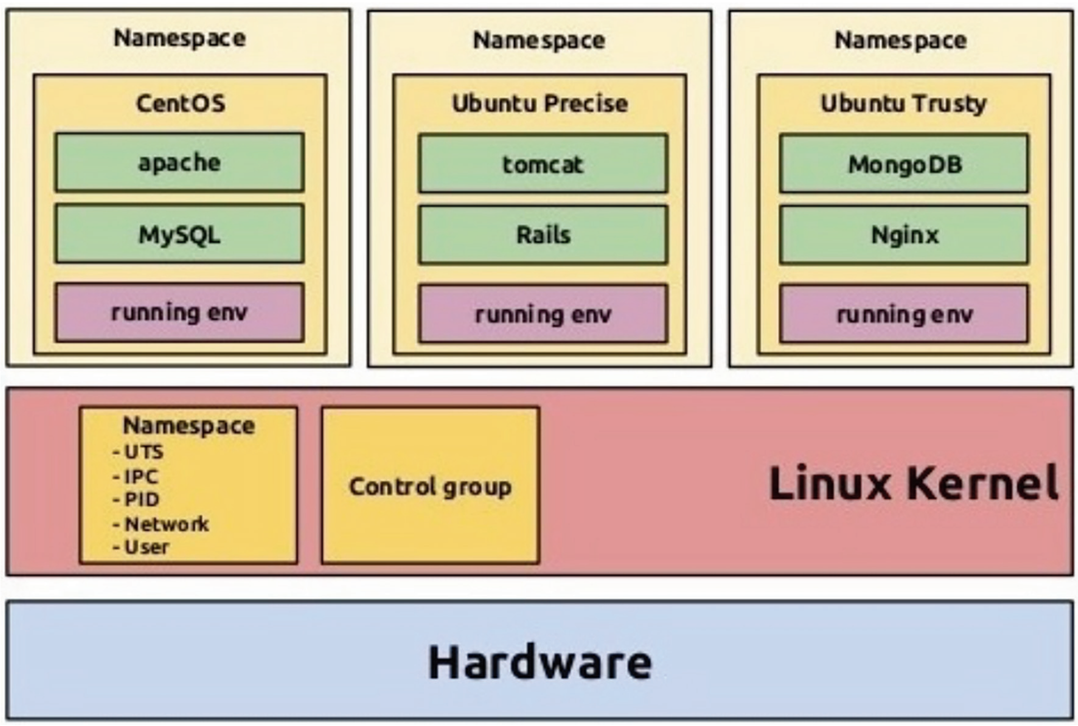


Figure 3: Linux Containers or LXC Containers

processes to be isolated on a joint operating system [S5]. Docker is now the maximum common container explanation and has been used to demonstrate containerization. A Docker copy comprises layered file systems analogous to the Linux virtualization stack's LXC instruments; Docker employs a combination mount to add a writeable file structure on top of the read-only file structure. This enables the coexistence of several read-only file systems. This feature enables the creation of new pictures by layering existing ones. Only the container's top layer is editable.

Containerization enables the transition from standalone containerized apps to clusters of container hosts capable of running containerized applications across cluster hosts. Containers' inherent compatibility facilitates the latter. Clusters of linked container hosts are formed from individual container hosts. Each cluster is comprised of numerous nodes (hosts). Application facilities are reasonable clusters of containers belonging to the identical image. Scaling an application over many host nodes is enabled via application services. Volumes are used to provide persistence methods in applications that need them. These volumes may be mounted for storage in containers. The use of links enables the connection and communication of two or more containers. Orchestration provision for inter-container statement, connections, and provision assembly is required to deploy and manage these container clusters (S. Soltesz et al. 2007).

2.2. Cloud-Based Container Designs

Container orchestration entails more than just starting, stopping, and shifting applications (i.e., containers) among servers. The process of constructing and continuously sustaining groupings of container-based computer program applications that may be geographically scattered is known as orchestration. Container orchestration allows customers to specify how the containers in the Cloud will be coordinated when they install a multi-container request. Container orchestration is the process of deploying and maintaining containers as a single entity. It makes sure that containers are always available, scalable, and connected. Container construction in the cloud is simply a form of orchestration within a distributed cloud environment. Cloud architecture can be conceived of as a layered and scattered structure, with core infrastructure, platform, and software application layers spread across several cloud environments (Antonio Celesti et al. 2019). Container technologies have the potential to assist. As such, container technologies are critical in the imminent application administration, particularly in the context of Cloud PaaS. Container-based architectures, which have recently gained popularity, may be implemented in this cloud platform. Given this shift in architectural style, secondary research may assist practitioners in selecting the best technology.

3. Related Works

Virtualization of resources usually entails installing a software layer on top of the host operating system to manage numerous resources. These virtual machines (VMs) may be thought of as their execution environment. Numerous techniques are used in the process of virtualization (M. Xavier et al 2013). Hypervisor-based virtualization is one common approach. KVM and VMware are two well-known virtualization systems based on hypervisors. A virtual machine monitor must be placed on top of the underlying physical system to take advantage of this technology. Additionally, each virtual machine supports (separate) guest operating systems. This virtualization strategy is feasible for a single host operating system to sustain many visitor operating systems (M. Xavier et al. 2013).

A different approach is represented by container-based virtualization. The hardware resources are partitioned in this approach by implementing many (safe) isolations. Guest processes acquire abstractions instantly using container-based technologies since they function directly at the operating system (OS) level via the virtualization layer.Typically, in container-based systems, one OS core is joint across virtual occurrences. Users perceive containers as autonomous operating systems capable of operating hardware and software self-sufficiently (S. Soltesz et al. 2007).

According to Biederman (E.W. Biederman, 2006), the isolation feature of containers is handled by kernel namespaces. This is a Linux kernel distinctive property that enables developments to achieve the required abstraction stages. Although containers do not communicate external with a namespace layer, there is a separation between the Host OS and Invitee Processes, and an individual container has its operating system. Biederman (E.W. Biederman, 2006) states that namespaces separate file systems, process IDs, networks, and inter-process communication. However, container-based virtualization solutions impose restrictions on resource usage according to process groups.

To be more specific, in container-based virtualization, c groups are responsible for allocating priority to CPU, memory, and I/O usage. However, some technologies that make use of containers consistently manage their resources using cgroups. By using container-based solutions, it is possible to dynamically deploy and consume microservices in packaged hosting settings. Microservice patterns are not a novel concept in the world of software architecture. They are now generally acknowledged as a cost-effective method of developing applications. Before the microservices design, the standard approach to service development was to mainly build monolithic systems. From a practical standpoint, this necessitates the creation of a unified platform capable of managing everything. Many of these problems may be resolved in Cloud settings by using scripting methods that support Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS) (SaaS). Such explanations, however, face difficulties when several ad hoc plans and groups are engaged, each with its different software and information requirements, as is the case when community-specific virtual apparatus images are not accessible for the Cloud. Lightweight Cloud-based solutions are advantageous in these settings. Microservices are an example of this kind of architecture.

The microservice architecture is based on the idea of "split and conquer". Micro-services fundamentally substitute a single big code base with smaller code basics managed by minor, agile teams. These codebases communicate through a sole API. The advantages of this approach are that each group may operate independently and be protected/disconnected from one another—a process referred to as exemption cycles. However, in certain instances, such as when services from various organizations are interdependent, these exemption cycles may be linked. There is currently a plethora of container-based micro hosting providers. Docker, CoreOS, and LXC are the most well-known of them. Docker simplifies the process of encapsulating an application in a container, along with any necessary accessories for operation.

This is accomplished using a focused collection of apparatuses and combined Application Programming Interface Guidance knowhows that include kernel-level structures, such as Linux containers, control clusters, and a copy-on-write file arrangement. Assembling a container's file arrangement, Docker relies on the Advanced Multi-Layer Unification File System (AMULFS) (AuFS). AuFS supports the exact superimposition of one or more accessible file systems. It allows Docker to make use of the images necessary for the container's basis. For example, a user may use one Ubuntu image as the basis for many more containers. Docker makes use of a single copy of Ubuntu by using the Advanced Multi-Layer Unification File Scheme. This significantly lowers the amount of storage used and the amount of RAM used by containers due to their fast launch. AuFS also offers one additional advantage: it supports image versioning. Each new edition is tagged as diff, a file ordering feature that indicates the variance between two records. This enables the reduction of the amount of picture files, for example.

Another advantage of AuFS is that it allows to track the changes made to image versions—similar to how software expansion code versioning schemes work. CoreOS is a very new Linux delivery that was created to function tons of software structures. This technique enables a slimmed-down Linux core intending to minimize outlays to the utmost. Additionally, CoreOS provides capabilities for ensuring redundancy and safeguarding against system failure. CoreOS has announced an original rocket container runtime (rkt), which implements the submission container standard. This technology's primary objective is to create a customized container model.

Additionally, Amazon Machine Images are supported by this method. The rocket container runtime is an alternative to Docker, including enhanced safety and other requirements for server-based production. The rocket container runtime conforms to the Submission Container standard, which defines a new collection of container presentations that make them readily transportable or moveable. Docker executes each process through a daemon, which does not offer the same level of assurance as Rocket in terms of security. To address this issue, some have proposed that Docker be entirely rebuilt.

LXC is integrated into the standard Linux kernel, and the project allows for the management of container and operating system images via instrumentation. Containers may be thought of as lightweight operating systems that run alongside the host operating system. Because containers do not emulate the hardware film, they may run near-native speeds without additional overhead performance. In their normal operation, apps and web loads are created and installed in a specific configuration on bare-metal servers for challenging reasons. For instance, simultaneous installation and configuration of PHP, MySQL, Nginx, and Drupal are possible. Currently, however, programs are installed on a specific computer and cannot be easily moved. A virtual computer may be installed and relocated, and although this provides the illusion of portability, performance suffers as a result. The LXC container provides near-bare-metal performance and the flexibility to move easily across systems by encapsulating comparable stacks in containers. The increased speed and flexibility of an LXC container provide the appearance of a dedicated server. Clones, backups, and snapshots of containers are possible. LXC simplifies container management and adds new levels of freedom to the execution and deployment of applications. Flockport enables the rapid deployment of web stacks and applications packaged in LXC containers on any Linux-based computer. Flexibility and throughput are directly supported by LXC containers, while Flockport is a mechanism for distributing and using LXC containers.

Numerous academics have focused on narrowing the performance and optimization gap between virtualization and non-virtualized methods. Typically, these studies employ a unique approach and a set of instruments. Additionally, the compositions of the methods that have been studied and compared differ. Recent research articles, for example, analyze and compare the opening between inherent systems and their envisaged counterparts and the presentation disparities between container- and hypervisor-based virtualization methods. We acknowledge that this is a rapidly evolving area, and as a result, some of the older articles use out-of-date tools. Additionally, contemporary visualization technology has not been studied. In a recent paper, Hwang et al. evaluated four distinct hypervisor-based virtualization systems (J. Hwang, S. Zeng, T. Wood 2013). The authors did not find a hypervisor that performed much better. As a result, they recommended that customers use various software and hardware stages in Cloud services to meet their needs.

Abdellatief et al. (E. Abdellatief, N. Abdelbaki 2013) compared the presentation of VMware, Microsoft Hyper-V, and Citrix Xen in various situations. The authors utilized a modified SQL instance technique to conduct the assessment. They created millions of items, consumers, and orders using this technique.

(S. Varrette et al. 2013) conducted a similar study using different technologies and testbed environments. The authors conducted tests with high-speed computing using kernel-based computer-generated machines instead of Microsoft Hyper-V. Their research examined power usage, energy efficiency, and scalability.

Although the evidence for virtualization overheads was inconsistent, (S. Varrette et al. 2013) established that the virtualization film for virtually every hypervisor substantially impacted the presentation of virtualized systems, particularly for high-performance computation spheres. Current articles compare and contrast hypervisors with container methods. According to Dua et al. (R. Dua, A.R. Raja, D. Kakadia, 2014), Containers are gaining popularity to enable PaaS facilities. Estrada et al. benchmarked both KVM and Xen-based virtualization technologies, notably KVM, Xen, and LXC (Z. Estrada, et al. 2014). Their primary approach was to compare and contrast the runtime performance of each of the technologies they listed. Their tests and benchmarks were motivated by the goal of facilitating the development of series-based applications.

Heroku, a platform-as-a-service company pioneered containers to professionally and repeatably organize apps. Rather than treating the container as a computer-generated server, Heroku defines it as a procedure with enhanced separation characteristics. As a result, container-based application deployment provides a low-overhead solution with almost the same isolation as virtual machines. As with normal processes, it also includes resource sharing capabilities. Google's infrastructure largely relied on such containers. Additionally, Docker provided a standardized image format for application containers and administration tools for them.

4. Comparison of Micro-hosting Environments

Each container-based knowledge comes with a particular set of advantages and disadvantages. This section discusses some of the main structures of the leading container-based knowledge and their potential effect on performance. While CoreOs Rocket was mentioned in the preceding segment, no presentation assessment was performed since CoreOS has not yet published an official edition of their produce.

4.1. Docker

Docker uses numerous Linux kernel capabilities to enable the isolation of containers (Docker 2021).

4.1.1. Namespaces

Containers are deployed using Docker namespaces. Docker makes use of a variety of different kinds of namespaces to accomplish its job of isolating containers (M. Xavier et al. 2013):

- Docker containers are based on pid, which guarantees that no process in one container may influence processes in another;
- It makes use of the network to accomplish system interfaces, or more specifically, to isolate the system's networking resources;
- ipc is used to isolate particular inter-process communication (IPC) properties, including System V IPC things and POSIX communication queues. This implies that each IPC namespace has its set of properties for interprocess communication.

Docker uses the mnt namespace to provide processes with their view of a file structure and its mount facts.

- Throughout, identifiers for the kernel and version are isolated.

4.1.2. Control Groups

Docker uses cgroups to enable prevailing containers to segment presented hardware properties. It is conceivable that some of these resources may be limited in their usage at any moment. With Docker, some arduous tasks associated with low-level OS virtualization are abstracted away, saving users from enduring these tasks. According to Docker documentation, it aims to create an environment that is similar to that of the application's running environment no matter where it continues to run.Despite different hardware configurations, the software behaves exactly the same. The possibility of working with the software under its original intended conditions is perhaps even more appealing to developers since most unforeseen issues are not present under a different operating system. As a result, the operations team can ship the software to the final destination and expect it to behave similarly if the software runs properly in the container.

- The union file scheme is a kind of file scheme that works by creating films that serve as the foundation for containers.
- The container format may be thought of as a container that encompasses all of the preceding methods.

4.2. LXC

Linux Container is a container-based virtualization solution that allows for the rapid development of lightweight Linux containers via a standardized and extensible API and related implementations (Linux Containers 2021). By contrast, Docker is a container-based application development platform. They have several characteristics but also have a great deal in common. To begin, LXC is a method for virtualizing the operating system at the level of individual Linux containers on a sole LXC host. It does not utilize a computer-generated machine but rather creates a computer-generated environment with its CPU, memory, blocked I/O, system, and resource management system. This is accomplished through the LXC host's Linux kernel's namespaces and cgroups capabilities. As a chroot, but with much greater separation. LXC supports a broad range of virtual network and device types. Second, Docker uses fixed layers to facilitate the reuse of well-structured architectures, albeit at the expense of complexity and performance. Constraints on the number of applications per container limit the potential for usage. LXC allows the formation of single- or multi-threaded programs.

Additionally, LXC enables the creation of numerous system containers, which may be replicas of a single sub-volume that can be accessed through a btrfs file. This LXC capability enables it to address complex file system-level problems. Thirdly, LXC provides a comprehensive set of tools and rights for both container design and execution. In conclusion, LXC allows for the construction of poor containers, ensuring that non-root operators may figure containers. This functionality is currently unavailable in Docker.

4.3. CoreOS rocket

Rocket (Zhanibek Kozhirbayev and Richard O. Sinnott 2017) is a container-based technology developed by CoreOS as a substitute for Docker. Both Rocket and Docker automate application deployment via virtual containers that may run autonomously depending on the host's features. While Docker has evolved into a complex environment capable of supporting various needs and activities, Rocket is designed to execute basic tasks while maintaining a safe environment for application deployment. Rocket is a command-line tool that allows to execute application containers, descriptions of graphic designs. Rocket focuses on CoreOS's application container standard, a collection of descriptions that enables easy migration of containers. As Polvi acknowledged, Rocket may be tougher to use than Docker, as Docker streamlines the complete process of container construction through its informative boundary.

5. Evaluation parameters

There are many angles from which technologies may be compared, most notably in terms of performance. To assess container-based technologies from an overhead viewpoint, it was essential to comprehend (measure) the outlays associated with non-virtualized settings. The study in this section examined various performance metrics, including CPU and memory performance, system bandwidth and inexpression, and storage outlays. Multiple experiments were performed 15 times in each benchmarking phase to determine the precision and reliability of the different findings. Timing data was collected on an average and standard deviation basis (Zhanibek Kozhirbayev and Richard O. Sinnott 2017).

- CPU performance: The first scenario to evaluate the CPU performance was based on use of a compressor.
- Disk I/O performance: A key aspect of performance is the evaluation of disk I/O performance, specifically, volumes given as a non-ephemeral storage were attached to instances.
- Memory performance: The evaluation of Memory I/O performance is presented, The benchmark tool used to test the microhosting environments was the STREAM software (STREAM 2021). STREAM assesses memory throughput utilizing straightforward vector kernel procedures.
- Network I/O performance: The evaluation of Network I/O performance is presented.

6. Container Designs and their Management

The cloud enables the resistance of large-scale collective resources via the use of virtualization methods (P. Mell and T. Grance 2011). At the infrastructure layer, virtual machines (VMs) usually serve as the backbone. By comparison, containerization enables lightweight virtualization by customizing containers as application correspondences from separate images (often obtained from an image source) that use fewer possessions and time. Additionally, they enable more interoperable claim packaging, which is required for cloud-based transferable and interoperable package applications (C. Pahl 2015). Containerization is predicated on creating, testing, and deploying applications across many servers and linking these containers. Containers, therefore, solve cloud PaaS-related issues. Given the cloud's overall significance, it is critical to have a consolidated picture of ongoing operations.

6.1. Container Knowledge Principles

Container stores are packed, self-sufficient, ready-to-disperse components of applications and, if required, middleware and commercial logic (in the form of binaries and reference library), which are required to execute applications. Container-based apparatuses such as Docker are based on container

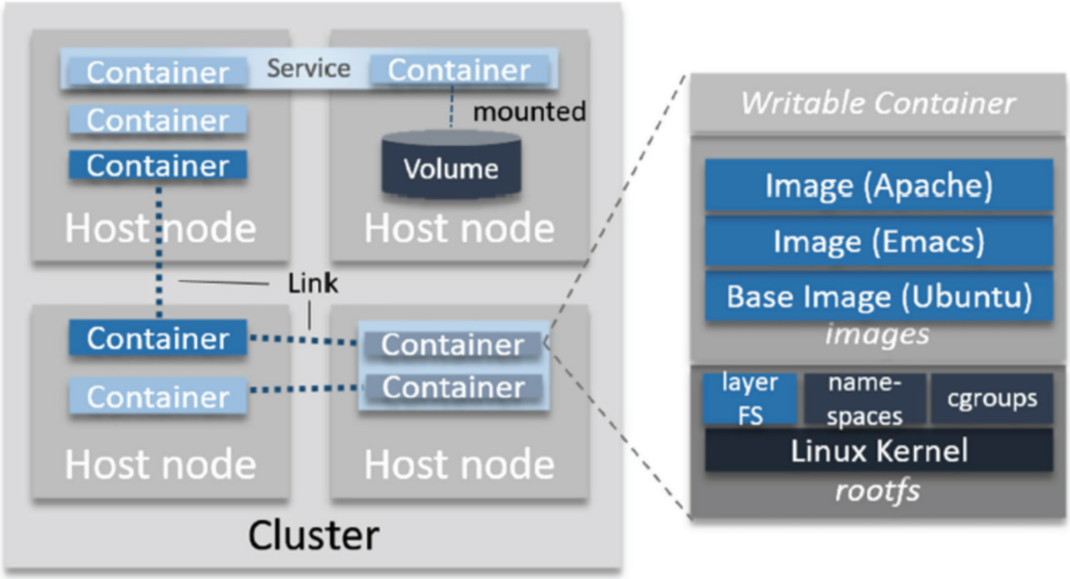


Figure 4: Container Cluster Architectures, (Claus Pahl, Antonio Brogi, Jacopo Soldani, and Pooyan Jamshidi 2019)

appliances, which serve as transferable containers for programs. As a consequence, in multi-tier systems, it is necessary to handle container dependencies. In a tiered plan, an orchestration plan may specify components, their needs, and their lifetime. After that, a PaaS cloud may execute the plan's processes through mediators (such as a container appliance). As a result, PaaS clouds may enable the positioning of container-based applications. Their coordinated development, deployment, and continuing administration are referred to as orchestration in this context.

Numerous container systems are built on the Linux LXC framework. Current Linux versions, a portion of the Linux container scheme LXC, have kernel features such as namespaces and cgroups that allow processes to be isolated on a collective operating system [S5]. Docker is now the utmost common container explanation and has been used to demonstrate containerization. A Docker image comprises tiered file systems analogous to the Linux virtualization stack, which utilizes the LXC instruments, as shown in Figure 3. Docker enhances a writeable file arrangement on top of the read-only file system via a union mount. This enables the coexistence of several read-only file arrangements. This feature enables the creation of new pictures by layering existing ones. Only the container's top layer is editable. Containerization enables the transition from single containerized apps to clusters of container hosts capable of running containerized applications across cluster hosts. Containers' inherent compatibility facilitates the latter. Individual container hosts are organized in clusters, as shown in Fig. 3. Individual clusters are comprised of numerous nodes (hosts). Application facilities are reasonable clusters of containers belonging to the identical image. Scaling an application over many host nodes is enabled via application services.

Volumes are used to provide persistence methods in applications that need them. These volumes may be mounted for storage in containers. The use of links enables the connection and communication of two or more containers. Orchestration provision for inter-container statements, links, and facility assemblies is required (P. Mell and T. Grance 2011).

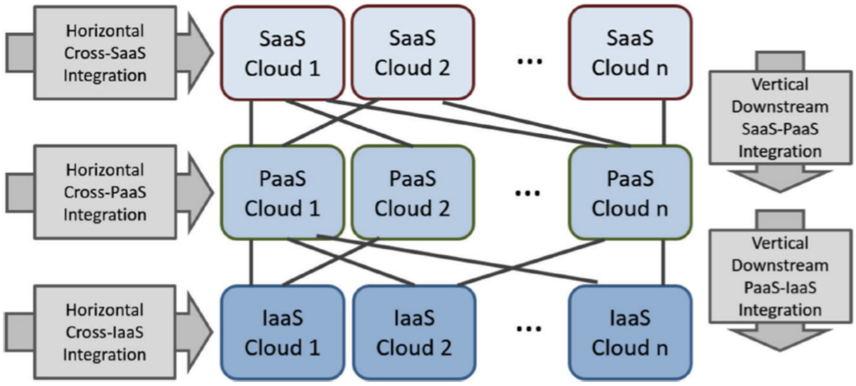


Figure 5: Cloud reference Architecture Model (Claus Pahl, Antonio Brogi, Jacopo Soldani, and Pooyan Jamshidi 2019)

6.2. Cloud-Based Container Designs

Container orchestration encompasses more than just starting and stopping programs (i.e., containers) and relocating them across servers. Orchestration is defined as the process of establishing and constantly maintaining clusters of container-based package applications that may be geographically dispersed. When users install a multi-container claim, container orchestration enables them to specify how the containers in the cloud are to be coordinated. Container orchestration encompasses container deployment and the administration of many containers as a single entity. It ensures that containers are available, scalable, and networked. Cloud-based container building is essentially a method of orchestration inside a distributed cloud atmosphere. The cloud may be thought of as dispersed and tiered, as shown in Figure 2, with core infrastructure, stage, and application layers spread over many cloud environments (A. Brogi, et al. 2016). Container technologies have the potential to assist. As such, container technologies will be critical in the imminent application administration, particularly in cloud PaaS. Containers enable this cloud framework to implement recent and popular microservice-based architectures (N. Kratzke 2015, J. Lewis and M. Fowler, 2014). Given this shift in architectural style, secondary research may assist practitioners in selecting the best technology.

7. Conclusion

Containerization is a lightweight virtualization of any program in cloud computing, which contributes to the extensive acceptance of cloud computing. It provides many advantages for both the development and deployment processes. Containers are categorized in two distinct configurations. Application Container and system container, the Application Container is a purpose-built individual container for the execution of a single kind of application, and system container is a user-space contained inside another container. In this paper, various container architectures and their organization, have been described. This paper has also presented a comparison of micro-hosting environments of containers.

8. References

Abdellatief, E., Abdelbaki, N. 2013, Performance evaluation and comparison of the top market virtualization hypervisors, in: 2013 8th International Conference on Computer Engineering & Systems, (ICCES), IEEE.

Biederman E.W., 2006, Multiple instances of the global linux namespaces, in: Proceedings of the Linux.

Brogi A. et al. 2016, SeaClouds: An Open Reference Architecture for Multi-Cloud Governance. Cham, Switzerland: Springer, pp. 334–338.

Celesti A. et al. 2019, A study on container virtualization for guarantee quality of service in Cloud-of-Things, Future Generation Computer Systems 99, pp. 356–364

Docker 2021, Build, Ship, and Run Any App, Anywhere, viewed 1 September 2021, https://www.docker.com.

Dua, R., Raja, A.R. and Kakadia, D. 2014, Virtualization vs containerization to support PaaS, in: 2014 IEEE International Conference on Cloud Engineering, (IC2E), IEEE.

Estrada, Z. et al. 2014, A performance evaluation of sequence alignment software in virtualized environments, in: 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, (CCGrid), IEEE, 2014.

Hwang, J., Zeng S. and Wood, T. 2013, A component-based performance comparison of four hypervisors, in: IFIP/IEEE International Symposium on Integrated Network Management, (IM 2013), IEEE, 2013.

Kratzke, N. 2015, "About microservices, containers and their underestimated impact on network performance" in Proc. 6th Int. Conf. Cloud Computing, pp. 165–169.

Lewis J. and Fowler, M. 2014. Microservices, http://martinfowler.com/articles/microservices.html

Linux Containers 2021, viewed 1 September 2021, URL http://linuxcontainers.org.Xavier M. et al. 2013, Performance evaluation of container-based virtualization for high performance computing environments, in: 21st Euromicro International Conference on Parallel, Distributed and Network-Based Processing, (PDP), IEEE, 2013.

Liu, C., Loo, B. T. and Mao Y. 2011, "Declarative automated cloud resource orchestration" in Proc. 2nd ACM Symp. Cloud Comput., Art. no. 26.

Mell P. and Grance, T. 2011, "The NIST definition of cloud computing" Recommendations Nat. Inst. Standards Technol., Special Publication, 800-145, http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

Pahl C. 2015, "Containerization and the PaaS Cloud" IEEE Cloud Comput., vol. 2, no. 3, pp. 24–31.

Pahl, C., Brogi, A., Soldani, J. and Jamshidi, P. 2019, Cloud Container Technologies: A State-of-the-Art Review, IEEE Transactions on Cloud Computing, Vol. 7, No. 3.

Soltesz, S. et al. 2007, Container-based operating system virtualization: a scalable, high-performance alternative tohypervisors, SIGOPS Oper. Syst. Rev. 41 (3) 275–287.

Stream 2021, viewed 1 September 2021, http://www.cs.virginia.edu/stream/ref.html.

Varrette, S. et al. 2013, HPC performance and energy-efficiency of Xen, KVMand VMware hypervisors, in: 25th International Symposium on Computer Architecture and High Performance Computing, (SBAC-PAD), IEEE, 2013.

Xavier M. et al 2013, Performance evaluation of container-based virtualization for high performance computing environments, in: 21st Euromicro International Conference on Parallel, Distributed and Network-Based Processing, (PDP), IEEE.

Zhanibek Kozhirbayev and Richard O. Sinnott 2017, A performance comparison of container-based technologies for the Cloud, Future Generation Computer Systems 68, pp 175–182.

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 11 N. 4 (2022), 411-425
eISSN: 2255-2863
DOI: https://doi.org/10.14201/adcaij.28380

A Novel Framework for Ancient Text Translation Using Artificial Intelligence

Dr. Shikha Chadha[1], Ms. Neha Gupta[2], Dr. Anil B C[3], and Ms. Rosey Chauhan[4]

[1, 2, 4] Department of Information Technology, JSS Academy of Technical Education, Noida
[3] Department of CSE(AIML), JSS Academy of Technical Education, Bengaluru
[c]Computer Sciences Department, University of Salamanca, 1 Escuelas St., Salamanca, 37003
shikhaverma@jssaten.ac.in, neha.gutta@jssaten.ac.in, anilbc@jssateb.ac.in, roseychauhan@jssaten.ac.in

KEYWORDS

ancient text, Artificial Intelligence (AI); Long Short Term Memory (LSTM); translation

ABSTRACT

Ancient scripts have been acting as repositories of knowledge, culture and of the history of civilization. To have greater access to the valuable information present in ancient scripts, an appropriate translation system needs to be developed to adapt to the complexity of the scripts and the lack of knowledge of the reader. In this study, a prediction and translation system has been implemented using Artificial Intelligence. The system has been trained using the Sunda-Dataset and a self-generated dataset. The translation of an ancient script, viz. from Sundanese to English, is done using a two layer Recurrent Neural Network. The technique used in this proposal is compared with a pre-existing translator called IM Translator. The results show that the BLEU score increased by 8% in comparison to IM Translator. Furthermore, WER decreased by 10% in contrast to IM Translator. Finally, the N-Gram analysis results indicate a 3% to 4% increase in 100% contrast value.

1. Introduction

Ancient scripts have been a great source of cultural and civilization knowledge with a lot of information of Vedas and Upanishads flowing in them that need to be preserved for future use. Most ancient scripts are available in degraded format and their complexity proves to be a great challenge for researchers who extract and translate their text (Chadha *et al.* 2015), (Wang *et al.*, 2016). To preserve them for future use, a system needs to be developed to translate scripts into a modern language.

Essentially, the paper focuses on a system for translating ancient languages into a universal language (Miyamoto, Y., & Cho, 2016), so that the valuable collection of information can flow from one generation to another.

The main motivation behind this study has been to give travelers from different parts of the world access to information about a rich and valuable culture which has the potential to increase tourism and ensures resources remain preserved for future generations. In contrast to it various scholars working on classical ancient scientific texts have a continuous tradition of discourse.

For the purpose of translating an ancient script into a universal language, Long Short Term Memory (LSTM) (Bijalwan *et al*., 2014), which is a technique based on RNN, has been used. As the LSTM does not require any linguistic rules, it further enables vector sequencing and uses many to many mapping to design the translator. It even solves the problem of gradient vanishing, as it stores the previous output and on its basis, the next state is given.

There are several translators that convert Sundanese text into English, such as IM Translator.net, Google Translator and stars21.com, which mostly use either a phrase-based statistical translator (PBMST) or Dictionary Based Translation, though the result of the dictionary-based translator is not very precise, as dictionary-based translators (Apriyanti *et al.* 2016), (Athiwaratkun & Stokes, 2016) translate the sentences word by word. Whereas PBMST (Bijalwan et al., 2014) translates the whole sentence using gradient vanishing, therefore meaning is preserved and the translation becomes more relevant (Chaudhary & Patel, 2018) (Jussà & Fonollosa, 2016).

The contribution of the paper is to translate the low resource Sundanese language into a universally accepted language i.e., English, being a low resource language pair translation, various sub-objectives are accomplished as stated.

- To collect ancient text using primary and secondary data collection techniques.
- To pre-process the ancient script using text cleansing and vectorization with one hot vector encoding in order to convert the input text into tensors.
- To generate a text translation and prediction model for translating the ancient script using LSTM into a current recognizable language i.e. English.
- To verify and validate (V&V) the developed ancient script text translation and prediction model.

1.1. Definition of Terms

- Vectorization: Is the process of converting a word into a large sequence of numbers that may hold the sequence of a complex structure, which can then be interpreted by a computer using various data mining and machine learning algorithms.
- Word Cleansing: Word cleansing includes the removal of stop words, such as punctuation marks, converting all uppercase letters into lower case letters which helps reduce the size of the source text to 75%.
- One Hot Encoding: One hot encoding is the process of Machine Learning (ML) that converts the data into some number of categorical value, which is fixed to a limited number of sets, with each value representing a different category. Thus, the input text characters are mapped on a numeric index so that each character has a unique index.

The rest of paper is organized as follows: Section 2 provides background on state-of-the-art research into various translation methods; Section 3 summarizes the data collection process of the Sundanese text, Section 4 contextualizes our models with respect to the experiment done using LSTM, Section 5 explores the obtained results and the analysis carried out on the Sundanese dataset, ending with a conclusion and a description of future lines of research.

1.2. Related Work

The technique presented in (Suryani et al. 2017) used Link Grammar (LG) and Statistical-Based Translation Technique (SBTT) to decide on the correct translation of the text. English prepositions can have more than one translation meaning in Indonesian text, so according to LG the word before the preposition is the word explained by it and the word after the preposition explains the preposition. Following the application of LG, the statistical method was used to translate the word. The Bilingual Evaluation Understudy (BLEU) showed that the translation of the preposition "with" in sentences achieved a precision of 81%, which is higher than that of Google's translator. The technique proposed in (Ray *et al.*, 2015) for the translation of a text from a low resource language to a target language with a limited parallel corpus involved making use of a rich resource language which was similar to the original source language, taking advantage of overlapping vocabulary. In the study, BLEU showed improvement in the sentence to 2-5 points. (Muzaini *et al*., 2018) proposed a technique that focuses on clause identification and plays an important role in the extraction of complex sentences. The sentences were fed into a translator for translation and then passed to the parser tree. 2000 sentences had been used and BLUE score was found to be around 31.75%. The study proposed by (Hinton *et al*., 2018) applied Part-Of-Speech Tagger (PoS Tagger) in a translator to get more efficient translation results. The Sundanese dataset was used for modeling and the results were better than if only surface form had been used, nevertheless, some problems were appeared during experiments, such as Out of Word Vocabulary (OOV) which was caused by a low amount of parallel corpus and noise in the ancient text. (Xiaoyuan *et al*., 2017) proposed a study for comparison between Recurrent Neural Network (RNN) and statistical-based network with n-gram model in which English-Indonesian Machine Translation (MT) and vice versa were conducted. The results indicate BLEU score and Rank-based Intuitive Bilingual Evaluation Score (RIBES) increased by 1.1 and 1.6 higher than statistical-based technique. (Lauriola *et al*., 2022) proposed a technique where clause identification plays an important role in the extraction of complex sentences. The sentences were fed into a translator and then passed to the parser tree. The BLEU score indicated that the resulting translation was at around 31.75%. (Chadha *et al*., 2020) used Rule-Based Machine Translation (RBMT), the shift reducing parsing was used for linguistic information on the source language and PoS tagger was used for word class. A bilingual dictionary was used for translation and achieved an accuracy of 93.33%. (Zhou *et al*., 2016) proposed a model for translating English to Indonesian language translation and vice versa using grammatical structure, cultural words, and writing mechanics. The calculated average accuracy was at 0.1163.

The study done by (Huang *et al*., 2018) performed the translation of a sentence from one language to another which required a better understanding of the source and target languages. A hybrid approach was used for translation which combined example-based machine translation and transfer approaches that exhibit an accuracy of 75%. (Zhang *et al*., 2018) proposed a Neural Machine Translation (NMT) system that used character-based embedding in combination in spite of word-based embedding. Convolution layers were used to replace the standard lookup-based word representations. BLEU points increased up to 3 points in the German-English translation task. (Lin & Shen, 2018) proposed an LSTM based language model and a Gated Recurrent Unit (GRU) language model. It used an attention mechanism similar to (Cho *et al*., 2014) from the machine translation.

The study done by (Maitra *et al*., 2015) proposed a system based on RNN and Encoder-Decoder for generating quatrains taking keywords as an input. The system learns the semantic meaning in the sentence and learns semantic meaning among the sentences in the poem. (Nurseitov *et al.*, 2021) proposed a machine translation technique using deep learning, Tanaka corpus was used to convert the Japanese language into the English language. In the above approach neural machine translation was used which belonged to a family of encoder–decoders in which the encoder encodes a source sentence into a fixed-length vector called tensors from which a decoder generates a translation. (Ray *et al*., 2015) proposed a model using RNN-LSTM for translating Arabic text into English. COCO caption dataset was built, the performance of the proposed model on the test dataset gave a result of 46.2 for the BLEU-1 score.

In (Swe & Tin, 2021) a system was proposed for a segmentation free translation system using Long Short Term Memory (LSTM) and training was done on word and text line level. Precision Pattern Rate (PPR) and Recall Precision Rate (RPR) was calculated to be 80%. (Singh *et al*., 2017) proposed the use of RNN-CNN along with a Generative-Adversarial Model for text encoding to convert images into text. The dataset used was Oxford 102flowers, and there were 19 layers of CNN (Convolution Neural Network), achieving an efficiency of 23.4 according to BLEU. (Haroon & Shaharban, 2016) proposed a system based on end-to-end neural networks for converting ancient Chinese language into contemporary Chinese language. It achieved an F-score of 94.4% and a BLEU score of 26.95 for translation from ancient into contemporary language and a score of 36.34 BLEU when translating from contemporary into ancient.

(Windu *et al.*, 2016) focused on various tasks that demonstrated the application of deep learning techniques in Natural Language Processing, different hardware, software and popular corpora were used for this purpose. The multiple NLP classifications were used, such as sequence classification, pair wise classification and word labeling etc. The typical structure was used for sequence word classification, with static vectors used as input. Super GLUE and GLUE Benchmark values were calculated.

In (Singh *et al.*, 2017) proposed a technique for handwritten recognition for Kazak and Russian languages. The model used Deep CNNs for feature extraction and Multi Layer Perception (MLP) was used for word classification and the results were compared with a model combining RNN and CNN. The results were compared with Simple HTR recurrent CNN, the recognition accuracy was found to be 75.8%.

The author suggested (Wicaksono & Purwarianti, 2010) a method for offline handwritten text recognition using encoder decoder technique. CNN was used as an encoder for input text line image whereas the Bidirectional LSTM fully connected to CNN was used as a decoder for the sequential prediction of handwritten Greek characters. The newly created EPARCHOS dataset was used and the results were calculated using MSE and MAR flexibility, the final perimeters showed the gradual increase in parameters using above.

2. Material and Methods

Ancient text translators essentially consist of various modules, implemented using deep learning, that convert the source text into a target language. A Neural Network (NN) has been used for the development of the model. The model consists of an encoder and a decoder, which involves running two LSTM Recurrent Neural Networks which work together simultaneously to transform one sequence

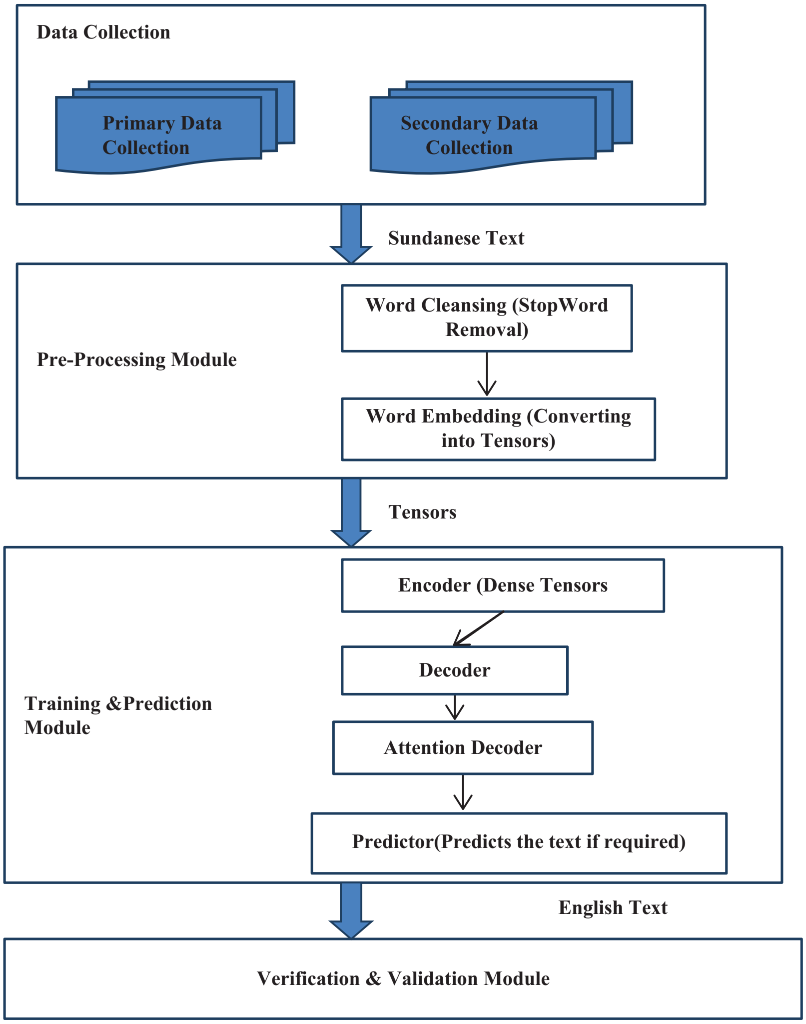


Figure 1. Architecture for Translation

into another. An Encoder network condenses an input sequence into a vector, a Decoder network unfolds that vector into a new sequence.

LSTM is a type of RNN model that computes the probability of occurrence of words in a reference text. The probability of a sequence of T words {w1, ...,wT} is denoted as P(w1, ..., wT). Since the number of words coming before a word, wi , varies depending on its location in the input document, P(w1, ..., wT) is usually conditioned on a window of n previous words rather than all previous words given in equation 1 (Costa et al., 2016). Pytorch is used for deep learning which works by converting the inputs into tensors.

$$P(w1,., wT) = i{=}T \prod i{=}1\ P(wi\ |w1, .,wi{-}1) \approx i{=}T \prod i{=}1\ P(wi\ |wi{-}(n{-}1)\ ,., wi{-}1) \qquad (1)$$

Figure 1 represents the flow for the translation of ancient texts, the first module includes both secondary and primary datasets for data collection. The secondary dataset i.e., Sunda dataset, was obtained from the author (Markou, K. *et al*. (2021), whereas the primary dataset was collected by converting English language into a Sunda dataset using a pre-existing translator. The data was preprocessed using word cleansing and word embedding i.e. converting into tensors.

2.1. Data Collection

In the proposed method of translation, the pre-existing dataset has been collected (Hermanto *et al*., 2015) and a self created corpus has been generated by translating English text into Sundanese using a pre-existing translator, so both secondary and primary data collection was carried out. Due to the unavailability of existing Sundanese to English parallel corpus, the primary data collection was done by generating a parallel corpus, by translating the English text into Sundanese text using Google Translate. The corpus contained 6615 bilingual sentences, 13,230 sentences and 76757 characters. An example of the created parallel corpus is shown in Table 1.

Table 1. Bilingual Corpus of Sundanese to English text.

English Text (Input Text)	i'd rather rather at home go out of milk	i have to make a complaint about	the students from from and and guru	and offered what was left to the little mouse
Sundanese Text (Output Text)	kuringlangkun-gresepcicing di bu-mitinimbangkaluar	Abdikedahnga-damelkeluhanngeu-naan	muriddiajartina guru sarengogé	Sarengmasihanna-onanukéncakabeu-ritsakedik

Furthermore, for secondary data collection, the data collected from (Huang et al., 2018) contains a character level dataset of a size of 6.7 MB, word level annotated dataset of a size of 231 MB with 66 images of the original Sundanese language converted into text format. The used Sundanese text originated from Brahmi Script in 1400 CE in the Java region and consist of 66 classes with 27 consonants, 7 vowels and 10 numerals. An image of the sentence level dataset is shown in Fig.2

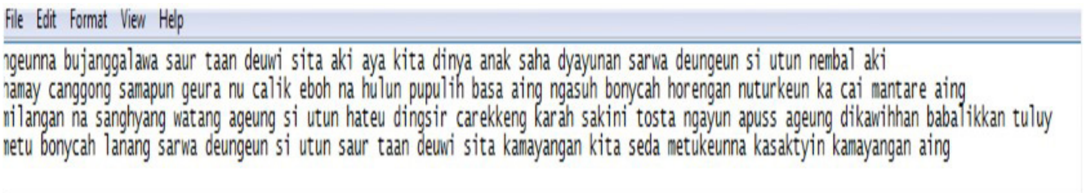

File Edit Format View Help
ngeunna bujanggalawa saur taan deuwi sita aki aya kita dinya anak saha dyayunan sarwa deungeun si utun nembal aki
namay canggong samapun geura nu calik eboh na hulun pupulih basa aing ngasuh bonycah horengan nuturkeun ka cai mantare aing
nilangan na sanghyang watang ageung si utun hateu dingsir carekkeng karah sakini tosta ngayun apuss ageung dikawihhan babalikkan tuluy
netu bonycah lanang sarwa deungeun si utun saur taan deuwi sita kamayangan kita seda metukeunna kasaktyin kamayangan aing

Figure 2. Original Sundanese Text

3. Methodology

The translation and prediction model comprises vectorization and training i.e. taking source text as input and converting it into dense tensors which are further converted into target text with the help of encoder-decoder units. Vectorization further comprises of text cleansing and one-hot encoding, which involves converting the source code i.e. Sundanese text into tensors.

The preprocessing involves text cleansing that reduces the size of the source text by removing stop words, punctuations and finally converting all the upper case into lower case. As the text is very ancient and due to very low level of knowledge of the source language i.e., Sundanese stemming and lemmatization was not be carried out in the proposed technique, thereafter, the size of the text was reduced to 80%, along with the performance of one-hot encoding which involves converting the clean text into tensors.

The training and prediction technique used in this study comprises of three functions, the first function converts characters to numeric indexes so that each character in the text has a unique index, whereas the second function converts those indexes into tensors. Finally, the same thing has been done for the whole pair of English to Sundanese, so the generated output is in the form of one tensor using Pytorch.

After the tensors were created, the training section took the tensors as input and converted them into the target language i.e., English. It was designed using a Neural Network (NN) consisting of an encoder and a decoder, that involved running two RNN simultaneously to transform one sequence to another (Purwarianti *et al.*, 2015). An encoder network condenses an input sequence into a vector, and a decoder network unfolds that vector into a new sequence. The encoder of a seq2seq network is an RNN that outputs some value for every word from the input sentence. For every input word the encoder outputs vectors and a hidden state, and uses the hidden state for the next input word.

Initially, the input is given to the encoder in the form of tensors which is further passed to the embedded layer that is used to give the dense representation to the words and their complicated meaning, input to the layer is integer encoded in order to represent each word uniquely and the same is given to Gated Recurrent Unit (GRU) (Fachrurrozi *et al.*, 2014) which is a kind of LSTM that intends to solve the problem of vanishing gradients, another input to GRU (Markou, K. *et al.*, 2021) is from a previously hidden layer, but as it is the first layer so the previously hidden layer tends to be empty and the result is further transferred to the output layer and hidden layer until the end of the sentence is encountered after which input is given to next layer.

The decoder takes the input from previous hidden layer and output of the encoder which is passed to the embedding layer, furthermore, the embedded output is passed to activation function Rectified Linear Unit activation function (ReLU) to avoid overfitting (Gautam & Chai, 2020) and the output of the function is passed to softmax which calculates a probability for every possible class, thereafter, the output is provided to the hidden layer until the end of the sentence is reached after which the output is provided to the output layer.

The output of the decoder layer is transferred to the attention decoder that is used to improve the performance of the encoder-decoder translation module in parallel, it gets the input from the encoder along with the decoder and a previously hidden layer of the decoder. The input of the decoder layer is given to the embedded layer that again gives a dense representation of tensors, which is further provided to the dropout layer to avoid overfitting, afterwards, this embedded output is given to the attention layer that produces attention weights for each tensor that are further given as input to Batch Matrix Multiplication (BMM) along with embedded output that is further passed to GRU to solve the problem of gradient vanishing and keeps on giving the input to the previously hidden layer until the end of the sentence is encountered. The training and prediction module are used to predict tensors into source language, by using the mapping done by the decoder, to finally convert tensors into a text file of the target language.

The summary of the LSTM based model is given below in Table 2. It consists of Hidden _Size= 256, Input_Size=4345, Output _ Size=3586. The model was trained internally on an 80% dataset and validated on a 20% dataset. The model was tested on 250 sentences and as shown in Fig.3, with a gradual increase of training dataset, the training loss decreases.

Table 2. Summary of the Training Model

Layer(Type)	Output Size	Parameter
conv2d_1 (Conv2D)	(None,32,32,32)	896
conv2d_2 (Conv2D)	(None,30,30,32)	9248
max_pooling2d_1 (MaxPooling 2)	(None,15,15,32)	0
dropout_1 (Dropout)	(None,15,15,32)	0
conv2d_3 (Conv2D)	(None,15,15,64)	18496
conv2d_4 (Conv2D)	(None,13,13,64)	36928
max_pooling2d_2 (MaxPooling 2)	(None,6,6,64)	0
dropout_2 (Dropout)	(None,6,6,64)	0
conv2d_5 (Conv2D)	(None,6,6,64)	36928
conv2d_6 (Conv2D)	(None,4,4,64)	36928
max_pooling2d_3 (MaxPooling 2)	(None,2,2,64)	0
dropout_3 (Dropout)	(None,2,2,64)	0
Flatten_1 (Flatten)	(None, 256)	0

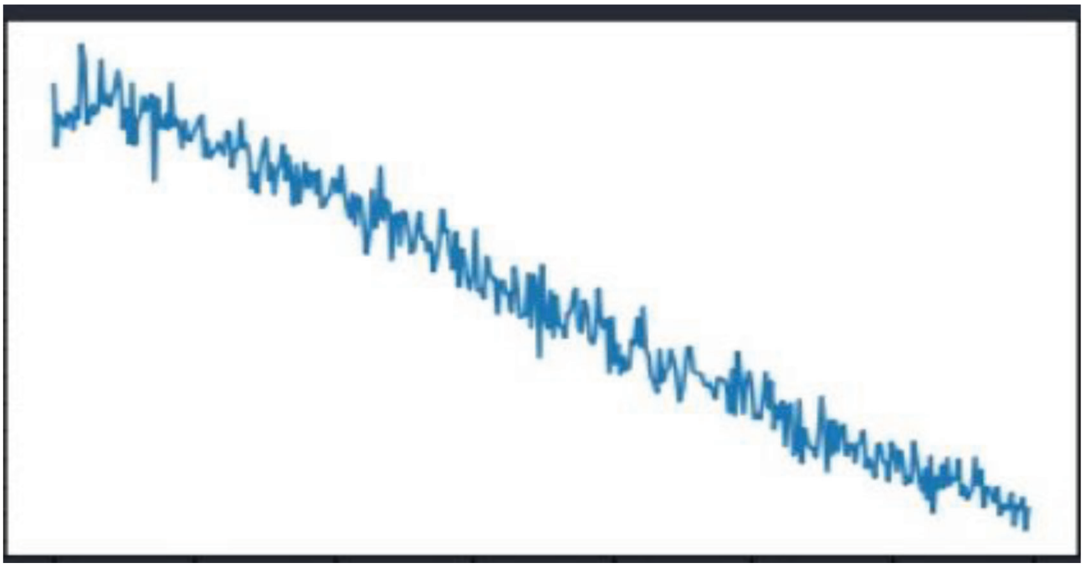


Figure 3. Graph showing Training Loss with 4500 epochs.

4. Results and Discussion

The results of Sundanese to English translation have been evaluated using manual and automatic evaluation. The model proposed by us has been evaluated on 250 sentences and analysis has been performed by evaluating and comparing various parameters, such as Average BLEU score, Word Error Rate WER and N-gram analysis on a pre-existing translator i.e. IM Translator.

4.1. Comparative Analysis

4.1.1. Word Error Rate (WER)

The WER is the difference in length sequence of recognizable text and reference text by Large Vocabulary Continuous Speech Recognition (LVCSR), the word sequence is hypothesized and is aligned with the reference script in order to calculate WER. If there is an N number of total words, the error is calculated as a summation of the number of Substitutions (S), the number of Insertions (I), and number of Deletions (D), WER is calculated by the equation [12].

$$WER = \frac{I+S+D}{N} \times 100 \tag{2}$$

The frequency range has been set by taking the average of Word Error rate for 250 sentences and four ranges, as shown in Fig.4. The WER is calculated, it is then compared with an existing IM Translator shown in Fig 4 and it has been observed that there were more sentences with WER (75-100%) for IM Translator than for Sun Tran, with a difference of approximately 4%. Whereas the total number of sentences with WER (25- 50%) for IM translator was approximately 4% higher than Sun translator, which increases the performance of the Sun Tran.

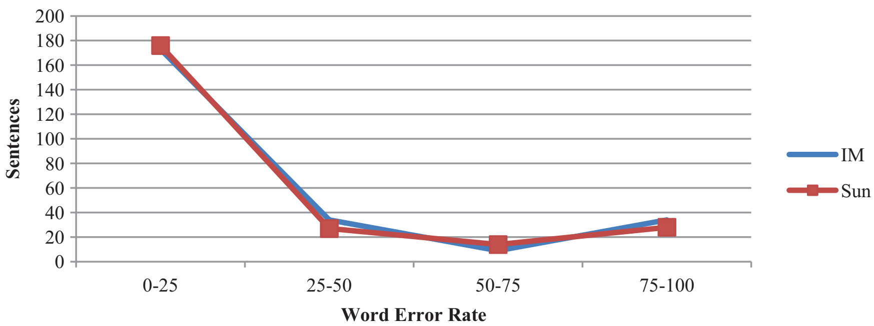


Figure 4. Comparison of Word Error Rate

4.1.2. Average Bleu Score Analysis

BLEU score is an automatic evaluation of the quality of text translated from one language to another, using a reference that was previously translated. The similarity index of the translated value is calculated by counting the same word that appears in both references and translation. The model proposed by us was evaluated on 250 sentences, the BLEU score was calculated, as shown, using equations 3 and 4.

$$B.P = \begin{cases} 1 \; if c > r \\ e\left(1-\frac{r}{c}\right) if c > r \end{cases} \tag{3}$$

$$BLEU = B.P \times \exp\left(\sum_{n}^{1} W_n \log p_n\right. \tag{4}$$

BP is a Brevity Penalty, which counters the length of translation result considering r as the length of reference text, whereas c is the length of translated text and Pn is precision recall of each n-gram. Average BLEU Score was calculated for Sun Trans and IM Translator in which BLEU score value for Sun Trans was 19 times greater than for IM Trans. Therefore, Sun Tran BLEU score was 8% greater than IM translator, as shown in Fig.5.

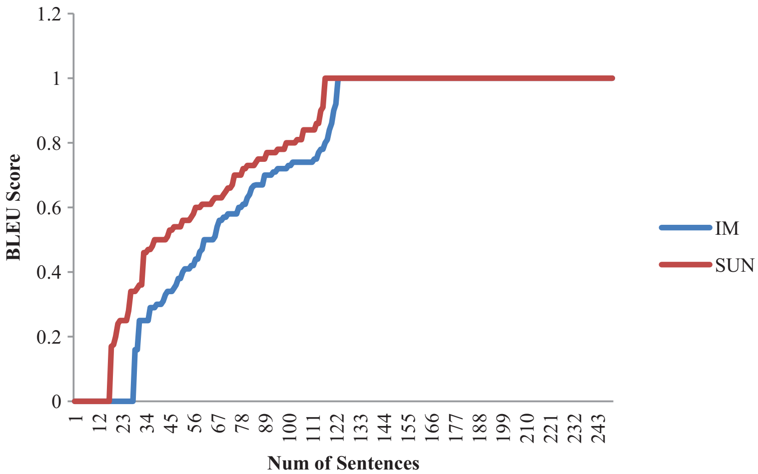


Figure 5. Comparison of Average BLEU Score

4.1.3. N-Gram Analysis

N gram is a continuous sequence of n words from a reference text, which are basically sets of occurring words in the text, when computing, N gram typically moves one word forward. When N=1 it is called 1-gram (uni-gram), if N=2 it is called 2-gram (bi-gram), if N=3 it is called 3-gram (tri-gram), if N is greater than 3 it is called 4-gram or 5-gram analysis is done.

If there are X words in a given sentence K and the number of n-grams in the sentences is computed as per given equation below.

$$N\text{-}gramsK = X\text{-}\ (N\text{-}1) \tag{5}$$

The Meteor score m is calculated using the mapped unigrams between the two strings i.e., the total number of unigrams in translation and the total number of unigrams in reference are shown in the equation.

$$M = t/r \tag{6}$$

UNI-GRAM

A uni-gram analysis is performed on all the words in a sentence, one by one, such as “kuring”, “langkung” ,“resep” ,“cicing”. In uni-gram analysis it is considered that the occurrence of each word in the sentence is independent of the previous word. The results indicate that for the words with Meteor score 0% been decrease by 3% for SUN-Tran, whereas for sentences with Meteor score 100% increases with 4% than IM –Tran as shown in Fig.6.

BI-GRAM

A bi-gram analysis on sequence of two words, such as “kuringlangkung”, “resepcicing”, was performed. In bi-gram it is assumed that the occurrence of each word is only dependent on the word that preceded it. Hence two words are counted as one gram i.e. one feature. The results indicate that meteor score for the sequence with Meteor score 0% has been decrease by 5% for SUN-Tran whereas for sentences with 100% meteor score increases with 5% than IM –Tran as shown in Fig 6.

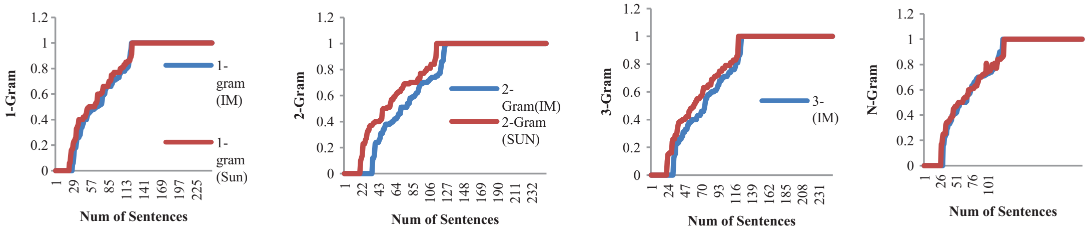


Figure 6. N-gram Analysis for IM and Sun Trans

TRI-GRAM

A tri-gram analysis on a sequence of three words, such as “ayakitadinya” was done. In Tri- Gram it is assumed that the occurrence of the word in the sentences is dependent on its previous two words. In tri-gram three words are taken as 1 gram i.e. one feature. The results has been found that for the words with 0% Meteor score has been decrease by 6% for SUN-Tran whereas for sentences with 100% Meteor score has been increased with 4% than IM –Tran as shown in Fig 6.

N-gram analysis (N>3)

An n-gram analysis is done on a sequence of four or more words, such as “ayakitadinyaanak". In n-gram, a collection of more than 3 words is taken as 1-gram and the occurrence of the word depends upon the N previous words. In an n-gram sequence, N words are taken as one-gram or one feature. The result indicates that for the words with 0% meteor score has been decrease by 2% for SUN-Tran whereas for sentences with 100% Meteor score has been increases with 4% than IM –Tran as shown in Fig.6.

For the construction of the API, as shown in Fig 7, two different layers are mapped using URL path (‘/’) for initial and final webpages. Two HTML template pages are rendered in API. When Python code is run, it connects to the Flask server at ‘http://localhost:5000/’. Flask then checks if there is a similarity match between the provided path and the defined function and shows us our HTML markup.

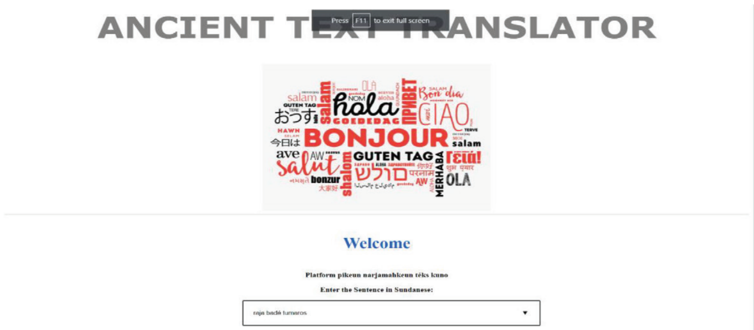


Figure 7. Screenshot of the Sun Translator

First HTML markup caters a home page for welcoming the visitors to the site hosting the API. It contains a dropdown menu to select the desired languages, a text area to enter a sentence and a convert button. Hitting this convert button will make Flask run our Deep Learning Model, redirecting it to our output webpage.

Second HTML markup is a webpage which displays the result returned from our Deep Learning Model. This page ends with a note thanking the user.

5. Conclusions

In comparison to the early translations of Sundanese text into English using the IM translator, it was found that the developed Sun-Tran model achieves more accurate results, as the Average Bleu Score increased by 8%, WER decreased at certain frequency ranges from 14% and 10%. N-gram analysis showed that Uni-gram increased by 4%, Bi-gram increased by 5% and Tri-gram increased by 4% and 4-gram increased by 4% when compared with the previously developed IM Translator for Sundanese to English Translation.

Several problems had to be addressed during the translation of Sundanese text into English due to a limited bilingual corpus and a large number of complex non-translated words, as very little knowledge of the language is available. Furthermore, the translator can also be used to support the translation of low resource source languages to universal languages.

6. Annexure

Abbreviation	Nomenclature
ML	Machine Learning
LSTM	Long Short Term Memory
PBST	Phrase-Based Statistical Translator
RNN	Recurrent Neural Network
LG	Link Grammar
SBTT	Statistical-Based Translation Technique
BLEU	Bilingual Evaluation Under Study
OOV	Out of Word Vocabulary
NMT	Neural Machine Translation
PPR	Precision Pattern Rate
CNN	Convolution Neural Network
GRU	Gated Recurrent Unit
ReLU	Rectified Linear Unit activation function
BMM	Batch Matrix Multiplication
WER	Word Error Rate
HTML	Hyper Text Markup Language
R^2RNN	Recursive Recurrent Neural Network

7. References

Ali and Renals. S, “Word Error Rate Estimation for Speech Recognition: e-WER.” [Online]. Available: https://github.com/qcri/e-wer.

Al-Muzaini. H. A,. Al-Yahya. T. N, & Benhidour. H, “Automatic Arabic Image Captioning using RNN-LSTM-Based Language Model and CNN,” 2018. [Online]. Available: www.ijacsa.thesai.org.

Apriyanti, T., Wulandari, H., Safitri, M., & Dewi, N. (2016). Translating Theory of English into Indonesian and Vice-Versa. In Indonesian Journal of English Language Studies (Vol. 2, Issue 1).

Athiwaratkun and J. W. Stokes, “Malware classification with lstm and gru language models and a character-level CNN.”, 2016.

Bijalwan, V., Kumar, V., Kumari, P., & Pascual, J. (2014). KNN based machine learning approach for text and document mining. International Journal of Database Theory and Application, 7(1), 61–70. https://doi.org/10.14257/ijdta.2014.7.1.06

Chadha, S., Mittal, S., & Singhal, V. (2019). An insight of script text extraction performance using machine learning techniques. International Journal of Innovative Technology and Exploring Engineering, 9(1), 2581–2588. https://doi.org/10.35940/ijitee.A5224.119119.

Chadha, S., S. Mittal, and V. Singhal. "Ancient text character recognition using deep learning." International Journal of Engineering Research and Technology 3.9 (2020): 2177-2184.

Chaudhary. J and Patel. A, “IJSRSET1844500 | Bilingual Machine Translation Using RNN Based Deep Learning,” vol. 4, 2018, [Online]. Available: www.ijsrset.com.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. https://arxiv.org/abs/1406.1078

Fachrurrozi, M., Yusliani, N., & Agustin, M. M. (n.d.). "Identification of Ambiguous Sentence Pattern in Indonesian Using Shift-Reduce Parsing", 2014

G. Lin and W. Shen, “Research on convolutional neural network based on improved Relu piecewise activation function,” in Procedia Computer Science, 2018, vol. 131, pp. 977–984, https://doi.org/10.1016/j.procs.2018.04.239.

Gautam N. & Chai S. (2020). Translation into Pali Language from Brahmi Script. In: Sharma D.K., Balas V.E., Son L.H., Sharma R., Cengiz K. (eds) Micro-Electronics and Telecommunication Engineering. Lecture Notes in Networks and Systems, vol 106. Springer, Singapore. https://doi.org/10.1007/978-981-15-2329-8_12.

Hermanto, A Adji. T, & Setiawan, N (2015)“Recurrent neural network language model for English-Indonesian Machine Translation: Experimental study,” 2015 International Conference on Science in Information Technology (ICSITech), Oct. 2015, doi: https://doi.org/10.1109/icsitech.2015.7407791.

Hinton, G. E., & Zemel, R. S. (n.d.). Autoencoders, Minimum Description Length and Helmholtz Free Energy.

Lauriola, I., Lavelli, A., & Aiolli, F. (2022). An introduction to Deep Learning in Natural Language Processing: Models, techniques, and tools. Neurocomputing, 470, 443–456. https://doi.org/10.1016/j.neucom.2021.05.103.

M. R. Costa-Jussà and J. A. R. Fonollosa, “Character-based Neural Machine Translation,” Mar. 2016, [Online]. Available: https://arxiv.org/abs/1603.00810.

M. Suryani, E. Paulus, S. Hadi, U. A. Darsa, and J. C. Burie, “The Handwritten Sundanese Palm Leaf Manuscript Dataset from 15th Century,” in Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, Jul. 2017, vol. 1, pp. 796–800, doi: 10.1109/ICDAR.2017.135.

M. Zhang, Y. Zhang, and D.-T. Vo, “Gated Neural Networks for Targeted Sentiment Analysis.” [Online]. Available: www.aaai.org.

Maitra, D. sen, Bhattacharya, U., & Parui, S. K. (2015). CNN based common approach to handwritten character recognition of multiple scripts. Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, 2015-Novem, 1021–1025. https://doi.org/10.1109/ICDAR.2015.7333916

Markou, K. et al. (2021). A Convolutional Recurrent Neural Network for the Handwritten Text Recognition of Historical Greek Manuscripts. In: , et al. Pattern Recognition. ICPR International Workshops and Challenges. ICPR 2021. Lecture Notes in Computer Science(), vol 12667. Springer, Cham. https://doi.org/10.1007/978-3-030-68787-8_18.

Miyamoto, Y., & Cho, K. (2016). Gated Word-Character Recurrent Language Model. https://arxiv.org/abs/1606.01700.

Nurseitov.D, Bostanbekov.K, Kanatov.M, Alimova.A 1,2 , Abdallah.A, Abdimanap.G (2021). “Classification of handwritten names of cities and Handwritten text recognition using various deep learning models”, Advances in Science, Technology and Engineering Systems Journal Vol. 5.

P. Wang, P. Nakov and H. T. Ng, “Source Language Adaptation Approaches for Resource-Poor Machine Translation,” 2016, doi: 10.1162/COLI.

P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao, and B. Xu, “Text Classification Improved by Integrating Bidirectional LSTM with Two-dimensional Max Pooling,” Nov. 2016, [Online]. Available: https://arxiv.org/abs/1611.06639.

P.Y. Huang, F. Liu, S.-R. Shiang, J. Oh, and C. Dyer, “Attention-based Multimodal Neural Machine Translation.”, 2018.

Purwarianti.A, Yayat.D, Fakultas.P, “Experiment on a Phrase-Based Statistical Machine Translation Using PoS Tag Information for Sundanese into Indonesian”, International Conference on Information Technology Systems and Innovation (ICITSI),p.p 1-6, 2015.

R. P. Haroon and T. A. Shaharban, “Malayalam machine translation using hybrid approach,” 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), Mar. 2016, doi: 10.1109/iceeot.2016.7754839.

Ray, A., Rajeswar, S., & Chaudhury, S. (2015). Text recognition using deep BLSTM networks. ICAPR 2015 - 2015 8th International Conference on Advances in Pattern Recognition. https://doi.org/10.1109/ICAPR.2015.7050699

Ray, A., Rajeswar, S., & Chaudhury, S. (2015). Text recognition using deep BLSTM networks. ICAPR 2015 - 2015 8th International Conference on Advances in Pattern Recognition. https://doi.org/10.1109/ICAPR.2015.7050699

S. P. Singh, A. Kumar, H. Darbari, L. Singh, A. Rastogi and S. Jain, "Machine translation using deep learning: An overview," 2017 International Conference on Computer, Communications and Electronics (Comptelix), 2017, pp. 162-167, https://doi.org/10.1109/COMPTELIX.2017.8003957. Results should be clear and concise.

Singh. S, Kumar. A, Darbari. H, Singh.L, Rastogi. A & Jain.S, (2017). “Machine translation using deep learning: An overview,” 2017 International Conference on Computer, Communications and Electronics (Comptelix), Jul. 2017, https://doi.org/10.1109/comptelix.2017.8003957.

Swe, T & Tin. P. (2005). Recognition and Translation of the Myanmar Printed Text Based on Hopfield Neural Network. 6th Asia-Pacific Symposium on Information and Telecommunication Technologies, APSITT 2005 - Proceedings. 2005. 99 - 104. https://doi.org/10.1109/APSITT.2005.203638.

Swe, T., & Tin, P. (n.d.). Recognition and Translation of the Myanmar Printed Text Based on Hopfield Neural Network.

Wicaksono, A and Purwarianti. A, “HMM Based Part-of-Speech Tagger for Bahasa Indonesia Implementing Deep Learning Using Sequence-to-sequence for Automatic Question Generator View project Game refinement theory (M/P) View project HMM Based Part-of-Speech Tagger for Bahasa Indonesia,” 2010. [Online]. Available: https://students.itb.ac.id/home/alfan_fw@students.itb.ac.id/IPOSTAgger.

Windu, M., Kesiman, A., Burie, J.-C., & Ogier, J.-M. (2016). A New Scheme for Text Line and Character Segmentation from Gray Scale Images of Palm Leaf Manuscript. https://doi.org/10.1109/ICFHR.2016.63

Xiaoyuan.Y, Ruoyu. L and Maosong. S, (2017).“Generating Chinese Classical Poems with RNN Encoder-Decoder”, China National Conference on Chinese Computational Linguistics, pp 211-222, 2017.

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 11 N. 4 (2022), 427-450
eISSN: 2255-2863
DOI: https://doi.org/10.14201/adcaij.28394

Analyzing Social Media Sentiment: Twitter as a Case Study

Yaser A. Jasim[a], Mustafa G. Saeed[b] and Manaf B. Raewf[c]

[a]Department of Accounting, Cihan University-Erbil, Kurdistan Region, Iraq.
[b]Department of Computer Science, Cihan University Sulaimanyia, Sulaimanyia,46001, Kurdistan Region, Iraq.
[c]Department of Human Resource Management, Cihan University-Erbil, Kurdistan Region, Iraq.
yaser.jasim@cihanuniversity.edu.iq, mostafa.swe@gmail.com,
manaf.basil@cihanuniversity.edu.iq

KEYWORDS

deep learning; artificial intelligence; social media; data modeling; Twitter; CNN

ABSTRACT

This study examines the problem of Twitter sentiment analysis, which categorizes Tweets as positive or negative. Many applications require the analysis of public mood, including organizations attempting to determine the market response to their products, political election forecasting, and macroeconomic phenomena such as stock exchange forecasting. Twitter is a social networking microblogging and digital platform that allows users to update their status in a maximum of 140 characters. It is a rapidly expanding platform with over 200 million registered profiles and 100 million active users. Half of Twitter's active users log on daily, tweeting over 250 million tweets. Public opinion analysis is critical for applications, including firms looking to understand market responses to their products, predict political choices, and forecast socio-economic phenomena such as bonds. Through deep learning methodologies, a recurrent neural network with convolutional neural network models was constructed for Twitter sentiment analysis to predict if a tweet is positive or negative using a dataset of tweets. The applied methods were trained via a publicly available dataset of 1,600,000 tweets. Several model architectures were trained, with the best achieving a (93.91%) success rate in recognizing the tweets' matching sentiment. The model's high success rate makes it a valuable advisor and a technique that might be improved to enable an integrated sentiment analyzer system that can work in real-world situations for political marketing.

1. Introduction

Pattern classification and data mining are two concepts that are very closely connected and may be described officially as a process, either unsupervised or supervised, identifying relevant patterns in a large amount of data. Sentiment analysis has gained prominence in research as it enables understanding individuals' thoughts on an issue through the evaluation of big data. Many individuals utilize social networking sites, forums, and commonplace review websites to offer their perspectives on various services or goods. Obtaining active feedback from individuals is significant, not just for businesses measuring customer loyalty and competitors, but also for consumers who prefer to research a product or service before making a purchase (Abdullah and Mohammed, 2019) (Deborah *et al.*, 2021).

Subjectivity and Sentiment Analysis (SSA) has garnered considerable attention in recent years because of its importance in marketing and politics. A subjectivity classification is the classification of texts as objective or subjective. The sentiment categorization tool detects whether a subjective text is good, bad, unbiased, or mixed (Elnagar *et al.*, 2021)]. These many forms of SSA are becoming more prevalent because they give an automatic means of summarizing enormous quantities of text (review sites, forums, and Twitter feeds) into the conveyed ideas. Because of the broad availability of social media platforms, anyone may share their views or opinions about an occurrence, product, or problem (Niloufar *et al.*, 2021). Mining these informal and homogenous data is critical for making conclusions in fields. However, mining is challenging due to the massively unstructured nature of the sentiment data available on the web (Govindarajan, 2021). On the internet, textual information is divided into two categories: fact data and sentiment data (Xipeng *et al.*, 2020); By extracting important patterns and characteristics from a large dataset of tweets, the paper mainly relies on natural language processing techniques; and machine learning techniques to identify individual training dataset samples (tweets) based on the pattern model that best represents data.

Sentiment analysis offers a wide range of applications. The capacity to learn from social data is widely used in business. The following are some notable areas of sentiment analysis: Social media opinion analysis, social discussions in their social environment, and sentiment from the audience. Although businesses must track their mentions, sentiment analysis delves into the good, bad, and neutrals, as well as what individuals think about the company's goods or services (Mohammed, 2019) (Thabit and Jasim, 2017). However, sentiment analysis enables someone to monitor consumer reactions to the specific features of a product. This method might be carried out by monitoring general public sentiment toward that business over time and using economic tools to evaluate the link between public sentiment and the firm's stock market valuation (Thabit and Yaser, 2017).

The purpose of this paper is to examine the topic of sentiment analysis on Twitter. Whereas, tweets are classified based on whether they express a good, bad, or neutral attitude. Twitter is a microblogging and social networking website where users may submit 140-character status updates (Geetika and Yadav, 2014). This process is a fast-developing site, with over 200 million registered users, where 100 million users are active users, with half of them logging on daily, resulting in almost 250 million tweets daily. The authors aim to establish a representation of public opinion for decision-making by studying the emotions conveyed in tweets (Juan, 2021). Analyzing public opinion is essential for a variety of reasons, including corporations seeking to anticipate customer reactions to their products, forecasting political elections, and anticipating socioeconomic events such as the stock exchange. This work seeks to develop a practical classifier based on long short-term memory (LSTM) in the field of deep learning that can reliably and automatically assess the sentiment of an unknown Twitter stream (Alexander, 2019).

2. Related Works

In recent years, a considerable amount of research has been conducted on sentiment analysis. The strategy proposed by Chen et al. (2014) classified student data into several categories and solved student issues reported on Twitter (Peng *et al*., 2017). As Bhilare and Kasture argued, to mine opinions posted on social media sites, it is necessary to know grammar, how tags work, vocabulary, and semantic networks (Neethu and Rajasree, 2013), which covers sentiment classification and data collection methods (Sagar *et al*., 2014). In the domain of electronic goods, the accuracy of the classification technique using the specified feature vector and several classifiers, such as maximum probability, support vector machine, and ensemble classifiers, was investigated (Neha and Bhilare, 2015). The authors described a hybrid technique for identifying polarity in subjective consumer-product texts that incorporates sentiment lexicons and a machine-learning classifier (Seyed-Ali and Andreas, 2013). Gautam and Yadav (2014) provided a set of machine learning algorithms with semantic analysis to detect phrases and product evaluations based on Twitter data using WordNet for higher accuracy (Geetika and Yadav, 2014). Gokulakrishnan et al. evaluated the output of several classifiers for Twitter data categorization, such as social media optimization (SMO), SVM, and Random Forest (Balakrishnan et al., 2012). The authors proposed a method for normalizing noisy or nonsensical tweets and classifying them according to polarity, in other words, positive or negative. They also utilized a mixed model technique to generate numerous emotional words (Asil *et al.*, 2010).

3. Sentiment Analysis

Sentiment analysis, commonly referred to as opinion mining or emotion AI, is indeed the systematic recognition, extraction, assessment, and evaluation of emotional responses and opinions of the respondents utilizing natural language processing (NLP), text mining, machine learning, and biometrics. The voice in client materials, such as online surveys and reviews, as well as web-based social networking sites, is the primary focus of sentiment analysis (Abdullah, A. and Faisal, 2017). Sentiment analysis, in general, aims to analyze the attitude of a speaker, author, or other subjects in terms of topic, by noticing powerful psychological or intense reactions to an archive, conversation, or incident (Jasim and Mustafa, 2018). The attitude seems to be an emotional decision or evaluation, such as the creator's or presenter's passionate condition or an expectation of enthusiastic responses, in other words, the author's or dealer's intended impact.

User evaluations or comments on a broad range of concerns are already widely available on the Internet, and inspections may provide, among other things, user or media fault-finding questionnaires (Mustafa *et al.*, 2021). Surveys are becoming increasingly popular as individuals seek to express themselves online. As a result, there are a significant number of surveys accessible for every topic, making it difficult for users to decide which one to take, as they must read each one before deciding. A sentiment's polarity can be categorized as positive, negative, or neutral. It is indeed essential to mention that emotion mining may be done on three different levels (Nik *et al.*, 2020):

1. Document-level sentiment classification: At this level, a document can be classed entirely as positive, negative, or neutral.
2. Sentence-level sentiment classification: At this level, each statement is graded as good, negative, or impartial.

3. Aspect and feature level sentiment classification: At this level, sentences/documents can be labeled as positive, negative, or non-partisan depending on particular features of the sentences/archives, a process known as perspective-level evaluation grouping.

Central to the discipline of sentiment analysis is numerical representations of words, phrases, or documents. When using machine learning or deep learning techniques in text classification problems, the input characteristics used to represent text data in the model are word representations or word embedding (Muhamad *et al.*, 2021). There are various forms of representations: one-hot, bag-of-words, word2vec, fast text, glove, word embedding, and others (Judith *et al*, 2017). As a consequence, mining this data and separating and organizing customer reviews is a responsibility. Sentiment mining is a work that involves scanning a huge number of archives with natural language processing (NLP) and information extraction (IE) techniques to synthesize the sentiments in the remarks of various authors (Andreas *et al.*, 2017). In this approach, two techniques are used: computational etymology and information retrieval (IR). The basic idea underlying sentiment analysis is to assess the polarity of text documents or short phrases and then categorize them (Christopher, 2020).

4. Word Embedding

Using a word embedding model to turn all words into a real-valued vector is one method of expressing a text numerically. Word embedding is a high-dimensional vector produced by word mapping with a parameterized function developed using a data set typical of the source language. Word embedding enables the mapping of related words to the same location, which is a significant advantage. The t-distributed stochastic neighbor embedding (t-SNE) is a probabilistic approach for dimensionality reduction that may be used to display high-dimensional datasets, such as word embedding (Christopher, 2020).

In Figure 1, the similarities between data points are converted into joint probabilities using t-SNE. Furthermore, the t-SNE tries to decrease the disparity between high-dimensional data and joint probabilities (Matthew *et al*., 2020). Since the dimensionality is reduced, axes and their units on t-SNE plots have no meaningful relevance; however, t-SNE displays which point is closest to the source domain. Figure 1 shows a zoomed-in view of the quantity and the job regions in the t-SNE visual developed using word embedding as an example of bene t. In addition, there are four basic types of embedding: word2vec algorithms, glove, and embedding layer learning, as well as deep learning. The embedding

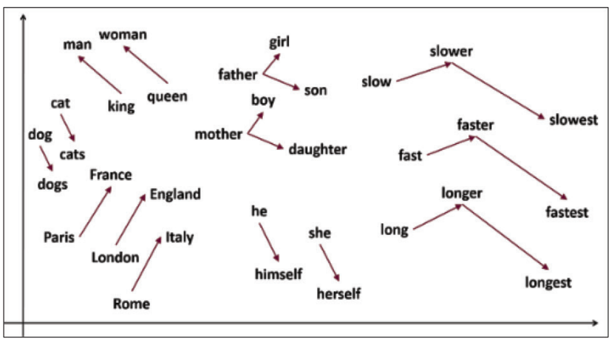


Figure. 1. t-SNE of Word Embedding

layer was employed in this study to represent the words as vectors, and this layer was learned using the deep learning model (Nik *et al*., 2020).

4.1 Embedding Layer

As a consequence, a layer is created to translate the one-hot representations into word embedding, and given a series of T words {w1, w2,…., wT}, each word wt ∈ W is embedded into a word-dimensional vector space using the following Equation as 1:

$$\phi\theta(wt)= Eft \tag{1}$$

The matrix **E** ∈ $\mathbb{R}d$wdd×|$\mathscr{W}$|, as the other network's other parameters, represents all of the word embedding to be learned in this layer. In reality, the authors employ a lookup table to replace this computation with a more basic array indexing operation, where Ewt ∈ $\mathbb{R}d$wrd relates to the word wt embedding. This lookup table action is then performed on each word in the sequence. A popular strategy is to concatenate all of the resultant word embeddings, as illustrated in Equation 2; this vector may then be given to further neural network layers (Mohammed and Jose, 2020).

$$\phi\theta\ (w1,\ w2,\ \ldots,\ wT) = [Ew1;\ Ew2;\ \ldots;\ EwT] \in \mathbb{R}(d\mathrm{wrd}\times T) \tag{2}$$

5. Deep Learning

Deep learning is a subclass of machine learning that uses multi-layer Neural Networks to transmit data from input to output and achieve improved accuracy in translation, speech recognition, and detection (Rohit, 2018), as seen in Figure 2.

Deep Learning differs from traditional ML methodologies. It can learn characteristics from data, such as photos, video, or text, autonomously, with no intervention from humans or implicitly supplied actions (Jan *et al*, 2021). When additional data is provided, the architectures of Neural Networks (NNs) may improve their prediction accuracy by directly learning from the raw data. DL is responsible for a variety of innovative developments, including voice assistants and self-driving automobiles (Rohit, 2018).

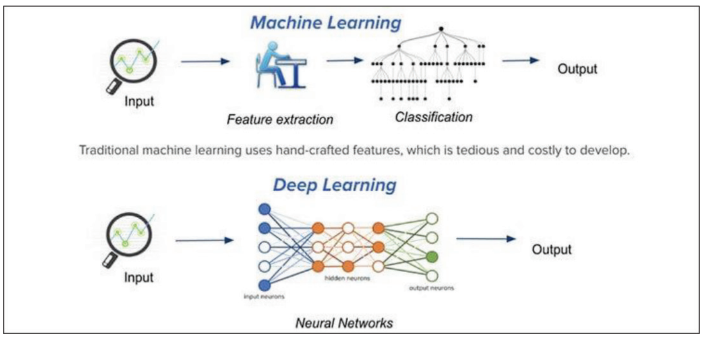


Figure. 2. Difference between ML and DL

6. Types of Deep Learning Algorithms and Techniques

Deep Learning techniques are classed as unsupervised, supervised, partially supervised, or semi-supervised. Another form of learning approach known as Reinforcement Learning (RL) or Deep Reinforcement Learning (DRL) is occasionally contested inside semi-supervised or frequently within unsupervised learning techniques (Priyanka and Silakari, 2021).

6.1. Deep Supervised Learning

Deep Supervised Learning makes use of labeled data. In this category, the environment has a set of inputs and equivalent outputs. For example, if the intelligent agent forecasts for input xt, the agent obtains loss values. Following that, the agent iteratively changes the network parameters to obtain more adequate approximations for desired output values. During practical training, the agent can ensure accurate responses to inquiries posed by the environment. There are several supervised learning algorithms for DL, including Convolutional Neural Networks (CNNs), Deep Neural Networks (DNNs), and Recurrent Neural Networks (RNNs), which use Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs) (Fahad *et al.*, 2020) (Nasim *et al.*, 2019).

6.2. Deep Semi-Supervised Learning

Deep semi-supervised learning occurs using partially labeled datasets. As semi-supervised learning algorithms, for example, Generative adversarial networks (GANs) and Deep Reinforcement Learning are used. RNNs, such as GRU and LSTM, were also used for semi-supervised learning (Ajay and Ausif, 2019) (Manuel *et al.*, 2021).

6.3. Deep Unsupervised Learning

Deep unsupervised learning is capable of doing so in the absence of data labels. In this situation, the agent will learn crucial characteristics or internal models to identify new correlations or structures in input data (Thabit and Yaser, 2015). Dimensionality reduction and clustering algorithms are also referred to as unsupervised learning techniques. Furthermore, several DL family members, such as GAN, Restricted Boltzmann Machines (RBMs), and Auto-Encoders (AE), specialize in clustering and nonlinear dimensionality reduction; recurrent neural networks, such as RL and LSTM, are also used for unsupervised learning in a variety of application areas (Marina and Emanuele, 2020).

7. Recurrent Neural Network

A recurrent neural network (RNN) is a kind of NN in which the connections between neurons produce a single directed cycle, as shown in Figure 3. Because sentences are a collection of words and documents, this feature is useful in NLP. RNNs are called recurrent because they conduct the same operations in each element (Yaser *et al.*, 2021). Furthermore, RNNs may remember the information they have seen in the sequence. However, they can only reflect a few steps in training, making them unsuitable for modeling long-term dependencies; This has a diminishing curve problem. To address this problem, most applications employ a specifically built RNN known as LSTM, which is significantly less vulnerable to these concerns (Marina and Emanuele, 2020).

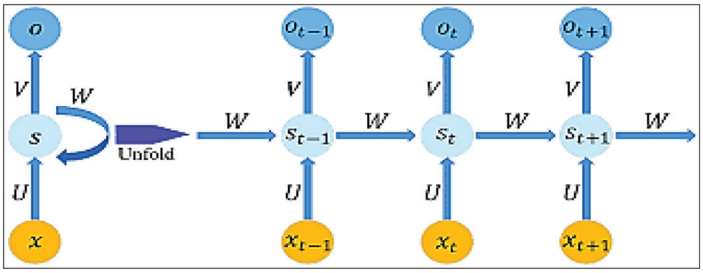


Figure. 3. A Simple RNN Architecture

7.1. Long Short-Term Memory

A long short-term memory is primarily an advanced RNN that is less vulnerable to the diminishing curve problem, and also better capable of analyzing long-distance relationships in sequence; these features make it useful for NLP applications where sentences and documents are sequential in nature (Yaser, 2018). LSTM is only intended to memorize a portion of the sequence observed thus far as illustrated in Figure 4. This behavior may be performed using network gates to determine what to store in memory depending on the current input and concealed state (Jasim and Mustafa, 2018).

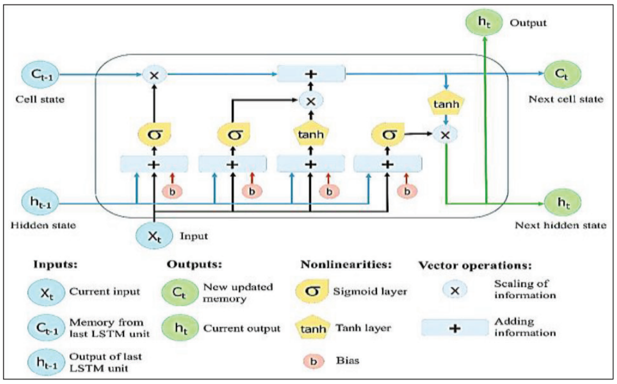


Figure. 4. The Basic Architecture of an LSTM Cell

The same effect occurs so often in separate time. It describes the governing in LSTM Equations (3-8):

$$it=\sigma(Wixt+Uiht-1+bi) \quad (3)$$

$$ot=\sigma(Woxt+Uoht-1+bo) \quad (4)$$

$$ft=\sigma(Wfxt+Ufht-1+bf) \quad (5)$$

$$Cin=tanh(Wcxt+Ucht-1+bc) \quad (6)$$

$$Ct=(it*Cin+ft*Ct-1) \quad (7)$$

$$ht=ot*tanh(Ct) \quad (8)$$

Where *it* = input gate at time-step t; *ot* representing the output gate at time-step t; *ft* representing the forget gate at time-step t; *Cin* representing the value of candidate state at time-step t; *Ct* representing the state of the cell at time-step t; and *ht* represents hidden LSTM cell output at time-step t. In contrast, the gate values are based on the current input and output regarding the preceding cell. Indicating the previous and current information for making decisions about that to keep in memory.

A new cell memory is developed in equation 5 by discarding a portion of the present memory and absorbing some recent input xt. In equation 8, the LSTM cell produces a portion of the new memory as output, however, the last cell output is normally treated as the final output, reflecting the full sequence. Within a few instances, a version related to LSTM known as peephole LSTM was used. The distinction is that the gates are based on the value of the preceding cell rather than the output of the previous cell (Alexander, 2019).

7.2. Activation Function Type in Deep Learning

The activation function is critical in the architecture of ANN models. The activation function applies a nonlinear adjustment to the input signal to learn and do more complex nonlinear tasks. Nonlinear activation functions are often utilized in neural networks. Sigmoid functions, Rectified Linear Unit (ReLU), and hyperbolic tangent (tanh) are the most widely employed activation functions (Chigozie *et al.*, 2018). The sigmoid function, often known as the logistic function, is a commonly used activation function. It is defined as a monotonically growing function in Equation 9:

$$\sigma(x) = sigmoid(x) = \frac{1}{1+e-x} \tag{9}$$

7.2.1. Sigmoid Function

The sigmoid function converts a real-valued number into a range between 0 and 1 as shown in Figure 5 (Safwan and Yaser, 2013). As a result, it has a very good interpretation of the output neurons that conduct the categorization task. Nevertheless, there are certain difficulties in the use of the sigmoid function due to diminishing or weak variation values at saturation points, either tail of 0 or a tail of 1. The network is either rejected for further learning or becomes extremely inefficient (Chigozie *et al.*, 2018).

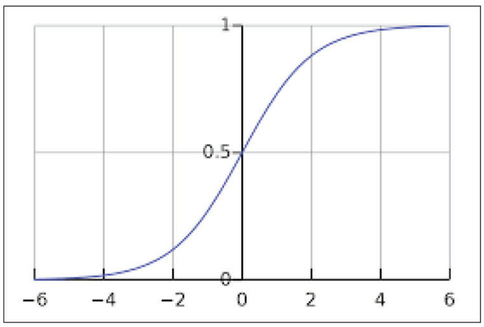


Figure. 5. Sigmoid Function

7.2.2. Hyperbolic Tangent Function

Another sort of activation function is the hyperbolic tangent (tanh) function. The hyperbolic tangent function is a nonlinear S-shaped function. Where the key distinction is that the tanh function's output range is zero-centered [-1, 1] rather than [0, 1], as illustrated in Figure 6 (Chigozie *et al.*, 2018). As a result, in training, the hyperbolic tangent function is favored, as shown in Equation 10:

$$\sigma(x) = tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (10)$$

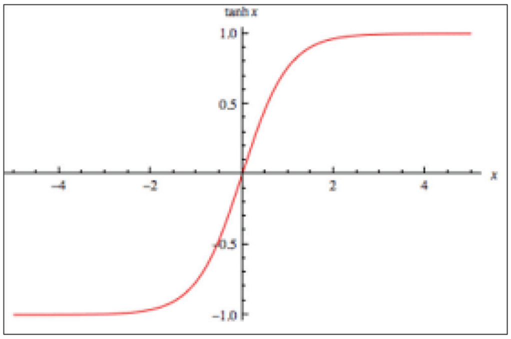


Figure. 6. Tanh Function

7.2.3. Rectified Linear Unit Function

Rectified Linear Unit (ReLU) functions, as defined mathematically in Equation 11, compress the net input to a value greater or equal to zero by setting the negative input values to zero; this is depicted in Figure 7. ReLU calculations were less expensive than sigmoid and tanh function computations since there is no calculation of the exponential function in activations, and sparsity may be used. The benefits of utilizing ReLU in neural networks include quick and efficient convergence and no gradient diminishing (Chigozie *et al.*, 2018).

$$\sigma(x) = ReLU(x) = max\ (0,\ x) \quad (11)$$

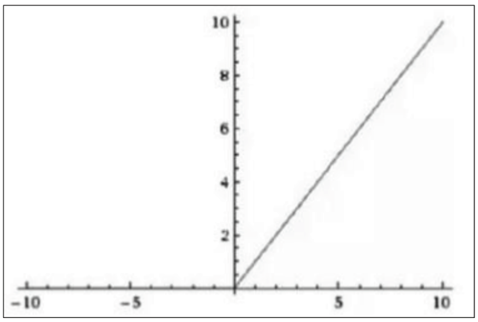


Figure. 7. ReLU Function

7.2.4. SoftMax Function

The SoftMax function is a function that is occasionally used in the output layer of NNs for classifications and is technically stated as demonstrated in Equation 12. The SoftMax function is a more extended logistic activation function that is used for multi-class classifications (Chigozie *et al.*, 2018).

$$\sigma(xj) = \frac{e^x j}{\sum_{k-1}^{n} e^x k} \tag{12}$$

7.3. Training

The neural network is trained using a training set that uses supervised learning to discover network parameters with the lowest error rate. The method generates a function that associates a given input with a class label; Where the goal denotes that the learning function may be utilized for previously unknown mapping inputs, which necessitates the function to model the underlying connection in the labeled samples from the training dataset (Sherry *et al.*, 2018).

7.4. Loss Function

The capacity of machine learning prediction or statistical models is dependent on a loss function known as the cost or objective function. The core principle behind the loss function is to calculate the error rate between anticipated and correct target values. As a result, minimizing the output of the losing function yields the best-performing machine learning model; nevertheless, there are numerous types of loss functions (Safwan and Yaser, 2013). Furthermore, these loss functions are used in conjunction with certain activation functions as indicated in Equation 13. The mean squared error (MSE) is a loss function that is used to calculate the average squared difference between actual and predicted target data.

$$MSE(y, \hat{y}) \frac{1}{N} \sum_{i=1}^{N} (y - \hat{y})^2 \tag{13}$$

Where N is the number of training samples, *y^i* is the model prediction value, and yi is the actual anticipated output. With hyperbolic tangent and linear activation functions, the Mean Squared Error (MSE) is commonly utilized. Moreover, it is presumptuous that the errors are regularly distributed.

7.5. Learning Rate

The learning rate is a measure of fit that specifies how much the model should change due to the predicted error each time the model weights are updated. Essentially, the researchers must decide where to step before determining the most recent direction at each advancement of traveling vertically to the curve. If the authors set a shallow learning rate, the training process may take a long time (Nikhil, 2017). However, if the researchers set a learning rate that is too high, then it would most likely begin to drift away from the minimum, as illustrated in Figure 8.

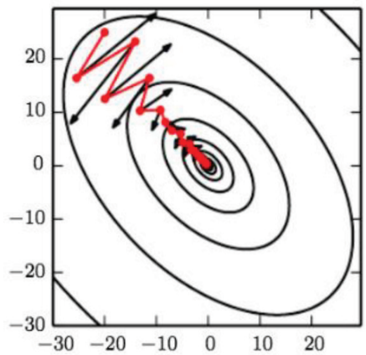


Figure. 8. Learning Rate at a Different Rate

8. Evaluation Metrics Methods

After developing a model, one of the most crucial phases is to evaluate its training and forecasting performance. Claims of the success of multi-layer algorithms frequently characterize the algorithm's quality using an essential set of performance measures (Guy *et al.*, 2019).

8.1. Accuracy

An algorithm might be evaluated using test data, with predictions from testing separated into four sets. In terms of categorization, the True Positives (TP) data was positive; it is also anticipated to be positive, whereas the True Negatives (TN) data was negative and is predicted to be negative. The data of False Positives (FP) was negative but was predicted to be positive. The data of False Negatives (FN) was positive yet forecasted to be negative, as indicated in Equation 14. A classification accuracy rate is calculated by dividing correct predictions by the total number of produced predictions (Georgios *et al.*, 2019) (Sherry *et al.*, 2018).

$$Accuracy = \frac{TP + TN}{N} \tag{14}$$

8.2. Precision/Recall

To assess the algorithm's efficacy, two metrics (precision and recall) were used. As illustrated in Equations 15 and 16, recall (R) is a measure of the rate of documents effectively described by the algorithm, whereas precision (P) is a measure of the percentage of returned documents that were true (Arvinder *et al.*, 2019).

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

$$Recall = \frac{TP}{TP + FN} \tag{16}$$

8.3. F1-Score

A system with a high recall but low accuracy produces large outcomes. Nonetheless, when compared to the training labels, the majority of its predicted labels would be inaccurate. Systems with great accuracy but low recall provide few results, but the overwhelming majority of their forecasted labels are accurate when compared to training labels. When determining how well a system operates, it may be useful to have a single number to define the performance (Mohammed, 2021). This may be accomplished by calculating the combined metric F1-score associated with the method as the harmonic mean of the recall and precision ratios, as stated in Equation 17. The F1-score may be regarded as the average of the two, considering how comparable the two results were (Arvinder *et al*., 2019).

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{17}$$

9. Methodology

This section discusses the architectural design for utilizing a deep learning model (LSTM) to categorize Twitter sentiment analysis. These deep networks are trained using a set of tweets, where the models are assessed using evaluation metrics. This paper's approach is separated into two sections. First, a diagram to show a simple description of the functioning mechanism; second, model development describing how to build deep learning models with their algorithms and mathematical foundations.

9.1. Twitter Sentiment Analysis Classification Steps

This work is completed in five stages. The first stage involves acquiring the dataset and then preprocessing it and using word embedding to represent the text and split it, using the outlier approach. Later, the LSTM model was developed, and ultimately, multiple metrics such as (accuracy, recall, and precision) are used to evaluate model performance, as shown in Figure 9 (Halina and Mohammed, 2017).

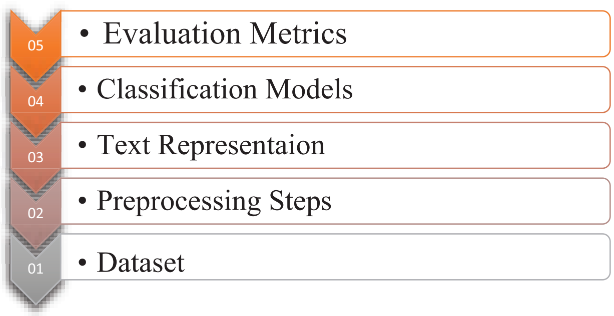


Figure. 9. Classification Steps of Twitter Sentiment Analysis

9.1.1. Dataset

The dataset for this paper (Twitter sentiment analysis.CSV file) was obtained from kaggel.com. This dataset contains around 1,600,000 records gathered through Twitter API streaming, and their features are shown as follows (target, ids, flag, user, text). The dataset was separated into 800,000 positive tweets and 800,000 negative tweets after the missing rows were removed. The dataset is read by the pseudocode in Algorithm 1.

Algorithm 1: Read Dataset and Removing the Missing Row
Input: All tweets dataset (CSV-File) ***Output: Dataset without missing rows and visualization labels*** ***Begin*** ***Step 1:*** *Read all dataset (.CSV-File) with missing row* ***Step 2:*** *Remove a missing row from the dataset // if any row of the datasets is a null, drop it from the dataset* ***Step 3:*** *Count the labels and display the counts in the chart* ***End***

9.1.2. Preprocessing Steps

The content in this section was first modified by eliminating (stopwords, punctuation, lower case letters, white spaces, and special characters.

Stage (1) Stopword removal: Stopwords are meaningless words in a language that generate distortion when considered as text classification data. These terms are frequently used in sentences to effectively link ideas or assist with sentence structure. Prepositions, conjunctions, and some pronouns, such as {that, this, too, was, what, when, where, who, and would, are all called stop words}.
Stage (2) Punctuation removal: Punctuation that does not have many variants of the same word is eliminated. If these punctuations are not eliminated, data should be considered as {been, been@ been!} Separately as distinct terms {! @, [/], and?} are some punctuation examples.
Stage (3) Lower case and white space removal: Lowercasing is eliminated from all of our text data, which is one of the simplest and most successful ways of text preparation. It is applicable to the majority of text categorization and NLP issues with predicted output consistency. Furthermore, joining and splitting functions have been used to eliminate all white spaces in a string.
Stage (4) Special character removal: Finally, non-alphabetic or numeric special characters such as {$, %, &.... etc.} should be eliminated from all text as shown in Figure 10, that depicts the steps of text cleaning.

Stemming, a preprocessing step utilized in most text categorization algorithms, is also provided. Stemming returns words to their original form (root), reducing the number of word forms or classes in the data. Running, Ran, and Runner, for example, will be reduced to the word run.

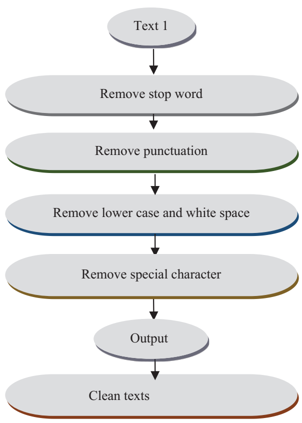


Figure. 10. Stages of Texts Cleaning

9.1.3. Text Representation Step

In this section, the text representation procedures, carried out as part of preprocessing, involved the following stages:

Stage (1): Prior to text representation, the dataset was fragmented using the holdout technique, with the dataset divided into two partitions. The training set receives 80% of the data, whereas the testing set receives 20%.

Stage (2): Every word frequency inside the dataset was collected to identify (N) main common words (after trial results, the researchers identified the most common words that offer the best result equals 6000), and each of these words was assigned a unique integer ID. For example, ID0 is assigned to the most major common word, ID1 to the next significant word, and so on. Then, replace each common term with its assigned ID and delete any single words. It should be noted that the 6000 most common terms barely cover the majority of the content. As a result, the cleaned text loses some information.

Stage (3): The LSTM units require a fixed length input vector. More than 300 words have been omitted. The original text string was converted to a fixed-length integer vector while the word order was preserved. Finally, a word embedding is used to move each word indicated by its ID to a 32-dimensional vector. Every word is represented as a vector based on word concurrences using word embedding. When two words appear often in the text, they are more similar, and the distance between the respective vectors is short. Furthermore, text representation is regarded as a critical stage since it involves converting every tweet in raw text to fixed-size vectors, as shown in detail in Algorithm 2.

Stage (4): An embedding layer is a word embedding developed in collaboration with a neural network model on a particular natural language processing activity, such as language modeling or document categorization.

Algorithm 2: Preprocessing Steps

Input: *a cleaned collection of labeled texts extracted from tweets dataset*
Output: *fixed-size vector for each tweet*
Begin
Step 1: *Splitting the dataset into two parts using holdout method*
80% for training (train-x, train-y)
20% for testing (test-x, test-y)
Step 2: *Computing the frequency of the words*
For *each word in (train-x)* ***do***
Compute the frequency of the word and put common words (top words)
into a vector
Visualize these words in the word cloud
End for
Step 3: *After computing the frequencies, encode the most common words (top words) in the text into an*
integer number (give ID for each top word) and delete the uncommon words
Step 4: *repeated step 3 to 4 in (test-x)*
Step 5: *padding and truncate //unify the length of all tweets to 300*
For *each tweet in (train-x)*
IF *tweet-length < 300*
Padding the tweet by adding zeros // Fill the rest of the tweet with
zeros
Else
Truncate the sentence into (300 words) // If the tweet length is longer than
Three hundred words, then truncate the remaining words.
End IF
End for
Step 6: *repeated step 4 for (test-x)*
Finally: *convert train-x and test-x to vectors*
End

9.1.4. Keras in Python

Keras in Python provides an embedding layer for neural networks on text data. It requires the input data to be an encoded integer ID, such that each word is represented by a unique number. The embedding layer is seeded with random weights and learns embedding for all training datasets. It is a versatile layer that may be utilized in a variety of ways, including:

- It may be used on its own to learn a word embedding that can subsequently be stored and utilized in another model.
- It may be utilized as part of a deep learning model in which the model learns the embedding itself.
- As a sort of transfer learning, it may be used to load a pre-trained word embedding model.

The embedding layer, a deep learning model in which the model itself learns the embedding, was utilized by the researchers in this article. The first hidden layer of a network is designated as the embedding layer. Three parameters must be supplied in this layer:

- Input dim: This is the words in the text data's vocabulary. For example, if your data is integer encoded to values ranging from 0 to 10, the vocabulary size would be 11.
- Output dim: This is the dimension of the vector space in which the words are to be embedded. It specifies the length of the output vectors generated by this layer for each word. It might, for example, be 32, 100, or much more significant.
- Input length: This is the length of input sequences, as defined for every Keras model input layer. This would be 1000 if all of the input documents were of a length of 1000 words. An embedding layer with a vocabulary of 30 encoded words, a vector space of 32 dimensions in which words are to be embedded, and input documents (tweets) containing 30 words apiece are used.

9.1.5. Classification Steps

The purpose of this section is to demonstrate the most important aspect of the proposed model. It shows how to use DL models (LSTM) and creates neural networks to assist in accurately identifying tweets. The input dimensions were (train-x) samples, while the output dimensions (train-y) were binary outcomes (one for positive and one for negative). This model had dropout and sigmoid output layers as shown in Figure 11. A generic block diagram was used to summarize the proposed models (LSTM).

9.2. Construction of the LSTM Model

To provide predictions, the proposed deep neural network model (LSTM model) was conceived and implemented. To accomplish so, the layers used to build the LSTM model were specified. One input layer, three hidden layers (embedding layer, convolutional layer, and LSTM layer), and one output layer comprise the deep neural network structure. The LSTM model is seen in Figure 12. In the structure of the LSTM model, tweet samples are trained and evaluated in the LSTM model.

9.2.1. Training Phase

This section describes the first classification model, LSTM. The (train-x) data is fed into the model to train the LSTM; linked weights adjust until the error value between the goal (train) and the LSTM predicted value is minimized. Because the LSTM model is binary, the binary cross-entropy loss function

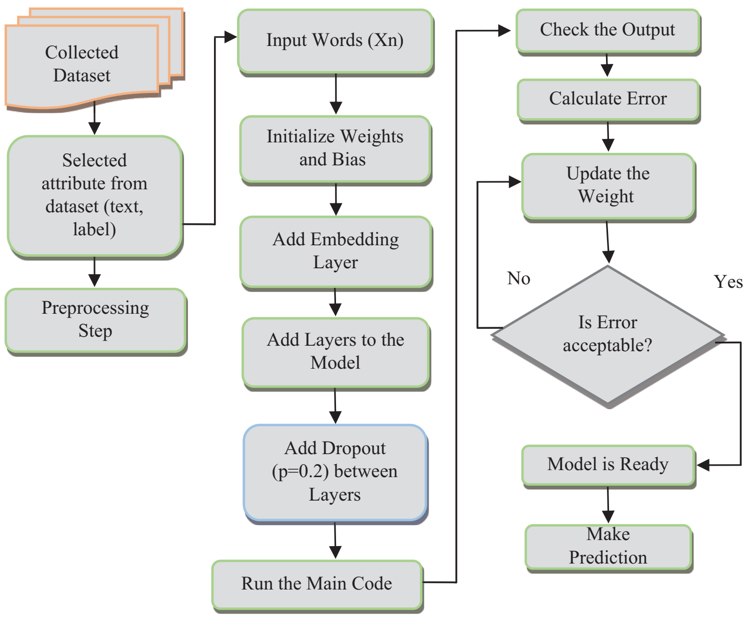


Figure. 11. The Main Diagram of The Proposed Models

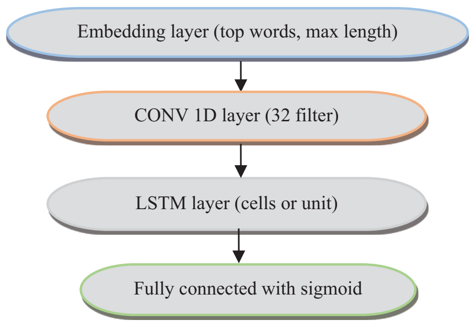


Figure. 12. The Proposed Model of LSTM

was used to calculate the difference between the exact distribution (which the LSTM method is attempting to match) and the predicted distribution. The training phase of this network is divided into two parts: feed-forward and backpropagation. The LSTM model's feed-forward phase consists of two steps:

Step (1): The input layer in this step consists of train-x samples of a vocabulary set of 6k words supplied to the embedding layer. The embedding layer is the model's initial hidden layer. These words are represented as numbers contained in a vector of 32 dimensions. The embedding layer learns with the model. This layer converts words into a fixed-size vector and searches for similarity-related terms. To avoid overfitting, we added a dropout layer and adjusted the dropout probability to 0.5 during the training phase. The embedding layer's retrieved words are subsequently placed into the LSTM layer. This layer includes (100 cells), each with three gates.

Step (2): These three gates analyze the words to create a word sequence. They are then sent to a fully linked layer. Using a sigmoid function, the last dense layer turns the vector into a single output in the range of (0 or 1) The error is calculated using a binary cross-entropy loss function. Finally, during the backpropagation phase, we compute the difference between expected and goal output and then apply an Adam optimization technique to change the weight values.

9.2.2. Testing Phases

The evaluation function may be used to assess the trained model after creating the LSTM model and adapting it to the training dataset. This is an estimate based on unobserved data. The assessment is based on the correct prediction of the tested labels (test-y). The accuracy measure is used to assess the model's real-world performance. The LSTM model's training and testing procedures are described in Algorithm 3. The final LSTM model was built after selecting the best hyperparameter from those (LSTM cell (100), learning rate (0.1), batch size (64) fixed in the algorithm below based on the trial-and-error method, which is considered by varied attempts that are continuously repeated until success is achieved or until the practice stops trying.

Algorithm 3: Training and Testing Steps of LSTM Model

Input: *{X1, X2…… Xn} (samples of [text: integer numbers] records)*
Output: *Classification results*
Required hyperparameter
num of epoch =10, batch size =64, num cells of LSTM=100, Size of words 30 k, max length=300, Vector length=32, learning rate =0.001, β1=0.9, β2=0.999, η=10−8
Initialize Weights (W) *= random number between (0,1)*
Initialize Bias *=1*
Define hidden layers: *(Embedding layer, LSTM layer)*
For *1 to (number of epoch)* ***do***
For *1 to (batch size)* ***do***
Step 1: *Create an embedding layer with 6 k input words, 32 vector length, and 500 max lengths of each sentence to represent each word as a vector.*
Step 2: *LSTM layer: the output of the embedding layer inserted into the LSTM layer input. The bellow formulas are applied to compute the three gates (input, output, and forget gates), cell, and hidden state that generates a sequence of words.*
$it = \sigma(Wixt + Uiht - 1 + bi)$
$ot = \sigma(Woxt + Uoht - 1 + bo)$
$ft = \sigma(Wfxt + Ufht - 1 + bf)$
$Cin = \tanh(Wcxt + Ucht - 1 + bc)$
$Ct = (it{*}Cin + ft{*}Ct - 1)$
$ht = ot{*}\tanh(Ct)$
Step 3: *The output of the LSTM layer pass through a fully connected layer with a sigmoid function to predict the final output results (0 or 1)*
Step 4: *Calculate the error by using binary cross-entropy*
Step 5: *Update weights by using (Adam optimization)*
End for
End for
Step 6: *Calculate the final results using (Accuracy, Recall, Precision, and F1-score) as defined in equations (14) to (17).*
Step 7: *Shows the results in the confusion matrix and history charts.*
END

9.3. Experimental Results of the Proposed Model

The outcomes of the practical stages of the proposed model are discussed first, with an illustration of the Twitter dataset details and their contents after selecting only the text and label characteristics. Table 1 displays five entries before selecting only the text and label from the dataset and working on them.

Table 1. Five Records from the Dataset of Tweets

#	Target	IDS	Date	Flag	User	Text
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	no_query	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl-Awww,t...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	no_query	scotthamilton	Is upset that he can't update his Facebook by…
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	no_query	mattycus	@Keinichan I dived many times for the ball. Man…
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	no_query	ElleCTF	My whole body feels itchy and like it's on fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	no_query	Karoli	@nationwideclass no, it's not behaving at all…

There was no need for a balancing step because the dataset was relatively balanced. Figure 13 shows each tweet segment in its original text and its new text after preprocessing processes. Preprocessing comprises removing stop words, punctuation, white space, and non-characters from the text to achieve a better representation when utilizing a word embedding layer to obtain an understanding of the text. In addition, all letters are uniform to be small letters. The data must be cleaned to improve word representation; Otherwise, the common words would be learned; and the keywords would have been disregarded, lowering classification accuracy.

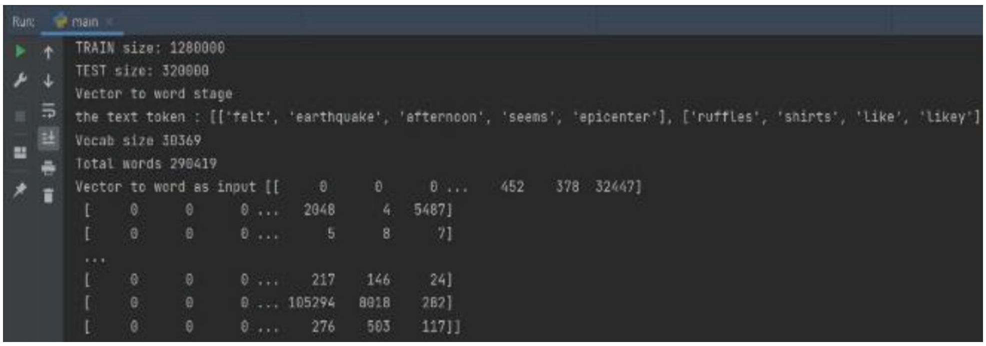


Figure. 13. Text Before and After Preprocessing Steps

As input for the proposed model, the dataset had been divided into two parts (20% for testing and 80% for training). The frequency of words in all texts is estimated to illustrate which words are the most influential in the texts and which are the least important, as shown in Figure 14.

Each word is allocated a unique ID based on its frequency. For example, because the word 'said' has the highest frequency, it is allocated to ID0, and the word with the next highest frequency is assigned to ID1, and so on. The ID of the terms in texts is shown in Figure 15.

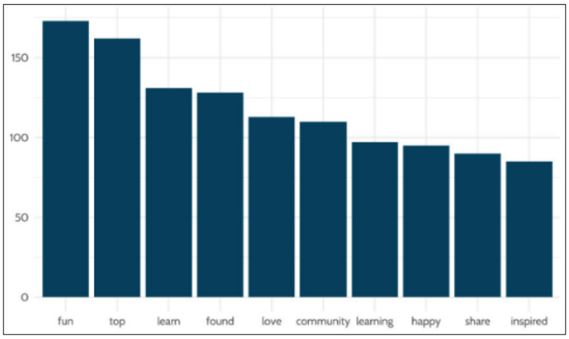


Figure. 14. Words Associated with Positivity

```
Run: main
TRAIN size: 1280000
TEST size: 320000
Vector to word stage
the text token : [['felt', 'earthquake', 'afternoon', 'seems', 'epicenter'], ['ruffles', 'shirts', 'like', 'likey']
Vocab size 30369
Total words 290419
Vector to word as input [[     0      0      0 ...    452    378  32447]
 [     0      0      0 ...   2048      4   5487]
 [     0      0      0 ...      5      8      7]
 ...
 [     0      0      0 ...    217    146     24]
 [     0      0      0 ... 105294   8018    282]
 [     0      0      0 ...    276    503    117]]
```

Figure. 15. ID Number of Each Word

Furthermore, because the proposed model demands a fixed-size entry, the researchers needed to know how many words we would utilize and the maximum length of the input phrases. As Figure 16 depicts the model diagram, which illustrates the proposed model in detail.

Table 2 shows the optimum hyperparameter that yields the highest accuracy result (93.91%) in the proposed model.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 300, 300)          79973100
_________________________________________________________________
dropout_1 (Dropout)          (None, 300, 300)          0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 300, 32)           28832
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 150, 32)           0
_________________________________________________________________
lstm_1 (LSTM)                (None, 100)               53200
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 101
=================================================================
Total params: 80,055,233
Trainable params: 80,055,233
Non-trainable params: 0
```

Figure. 16. The Proposed Model

Table 2. The Hyperparameters of the Proposed Model

Hyperparameter	Choice
Split dataset	80% train 20% test
Max length	300 max lengths
LSTM units	100 units
Dropout	0.2
Batch size	64
Num of Epochs	10 epochs
Learning rate	0.001

The hyperparameter values that the researchers selected are the most acceptable for the dataset. Figures 17, 18, and 19, respectively, depict the model's accuracy and the loss model in the proposed model.

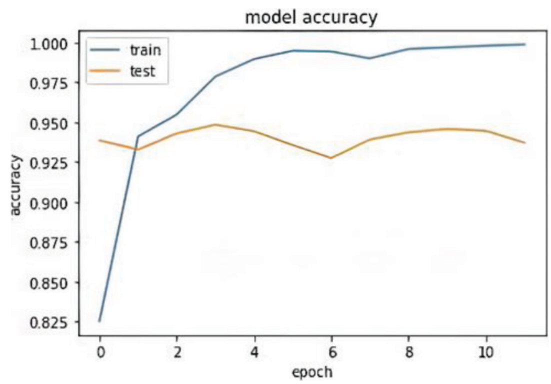


Figure. 17. Accuracy Result of LSTM Model

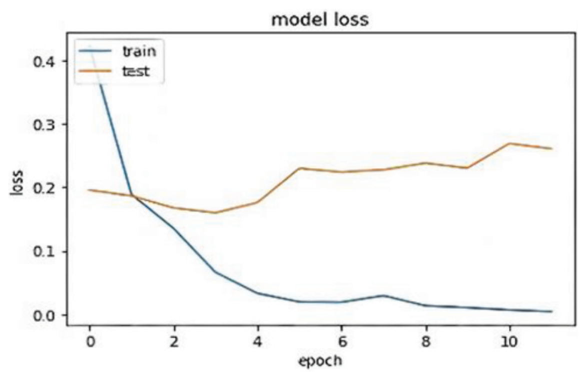


Figure. 18. Loss Result of LSTM Model

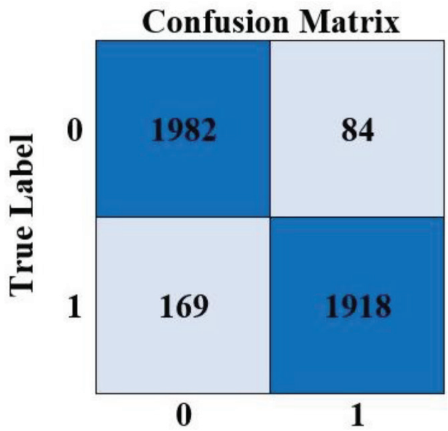


Figure. 19. Confusion matrix of proposed model

The confusion matrix contrasts the classification model's correct and erroneous predictions with the actual results (the target value). The classification results revealed that the recommended classifier had an accuracy of 93.91 percent. A confusion matrix containing TP, TN, FP, and FN values produced Figure 19.

10. Conclusions

In conclusion, the method's findings were extracted from the experiment results of the proposed technique as follows: First, a technique has been proposed for gathering, filtering, analyzing, and interpreting Twitter data using sentiment analysis; Second, in this study, more than 1,600,000 tweets were used to develop a functional and attempted method. By gathering tweets over a long period or using the Twitter Search API more often, the volume of data may rise; Additionally, machine learning methods, such as CNN and LSTM have the highest accuracy. The hybrid model (CNN-LSTM classifier) was used to construct the model, which had the highest accuracy. The overall accuracy rating was 93.91%; Furthermore, the preprocessing step is crucial since it has a significant impact on the classification's accuracy. When applying the proposed technique to this dataset, Adam's optimization strategy improved classification accuracy significantly. Generally, this research examines and compares current opinion mining methodologies, such as machine learning, text mining, and assessment criteria.

11. References

Abdullah, A., and Faisal, S. 2017. Data privacy model for social media platforms. In 2017 6th ICT International Student Project Conference (ICT-ISPC) (pp. 1-5). IEEE.

Abdullah, A., and Mohammed, H. 2019. Social Network Privacy Models. Cihan University-Erbil Scientific Journal, 3(2), 92-101.

Ajay, Sh., and Ausif, M. 2019. Review of deep learning algorithms and architectures. IEEE access, 7, 53040-53065.

Alexander, V. 2019. LSTM networks for detection and classification of anomalies in raw sensor data (Doctoral dissertation, Nova Southeastern University).

Ali, N., Elnagar, A., Shahin, I., and Henno, S. 2021. Deep learning for Arabic subjective sentiment analysis: Challenges and research opportunities. Applied Soft Computing, 98, 106836.

Andreas, S., and Faulkner, Ch. 2017. NLP: the new technology of achievement, NLP Comprehensive (Organization), New York, Morrow.

Arvinder, B., Mexson, F., Choubey, S., and Goel, M. 2019. Comparative performance of machine learning algorithms for fake news detection. In International conference on advances in computing and data sciences (pp. 420-430). Springer, Singapore.

Asil, C., Dilek, H., and Junlan, F. 2010. Probabilistic model-based sentiment analysis of twitter messages. In 2010 IEEE Spoken Language Technology Workshop (pp. 79-84). IEEE.

Balakrishnan, G., Priyanthan, P., Ragavan, T., Prasath, N., and Perera, A. 2012. Opinion mining and sentiment analysis on a twitter data stream. In International conference on advances in ICT for emerging regions (ICTer2012) (pp. 182-188). IEEE.

Chigozie, N., Ijomah, W., Gachagan, A., and Marshall, S. 2018. Activation functions: Comparison of trends in practice and research for deep learning. arXiv preprint arXiv:1811.03378.

Christopher, C. 2020. Improving t-SNE for applications on word embedding data in text mining.

Deborah, W., Ardoin, N., and Gould, R. 2021. Using social network analysis to explore and expand our understanding of a robust environmental learning landscape. Environmental Education Research, 27(9), 1263-1283.

Fahad, M., Ahmed, A., and Abdulbasit, S. 2020. Smiling and non-smiling emotion recognition based on lower-half face using deep-learning as convolutional neural network. In Proceedings of the 1st International Multi-Disciplinary Conference Theme: Sustainable Development and Smart Planning, IMDC-SDSP 2020, Cyperspace, 28-30 June 2020.

Geetika, G., and Yadav, D. 2014. Sentiment analysis of twitter data using machine learning approaches and semantic analysis. In 2014 Seventh international conference on contemporary computing (IC3) (pp. 437-442). IEEE.

Georgios, G., Vakali, A., Diamantaras, K., and Karadais, P. 2019. Behind the cues: A benchmarking study for fake news detection. Expert Systems with Applications, 128, 201-213.

Govindarajan, M. 2021. Educational Data Mining Techniques and Applications. In Advancing the Power of Learning Analytics and Big Data in Education (pp. 234-251). IGI Global.

Guy, H., Kok, H., Chandra, R., Razavi, A., Huang, S., Brooks, M., and Asadi, H. 2019. Peering into the black box of artificial intelligence: evaluation metrics of machine learning methods. American Journal of Roentgenology, 212(1), 38-43.

Halina, I., and Mohammed, H. 2017. A Review of Blended Methodologies Implementation in Information Systems Development Abdullah AbdulAbbas Nahi1. Muhammad Anas Gulumbe2. Nurul Syazana Selamat3. Noor Baidura.

Jan, L., Mohammadi, M., Mohammed, A., Karim, S., Rashidi, S., Rahmani, A., and Hosseinzadeh, M. 2021. A survey of deep learning techniques for misuse-based intrusion detection systems.

Jasim, Y, and Mustafa, S. 2018. Developing a software for diagnosing heart disease via data mining techniques.

Juan, C. 2021. Efficient Digital Management in Smart Cities.

Judith, B., Michael, B., Danica, K., and Kjellstrom, H. 2017. Deep representation learning for human motion prediction and classification. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 6158-6166).

Manuel, R., Gütl, C., and Pietroszek, K. 2021. Real-time gesture animation generation from speech for virtual human interaction. In Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems (pp. 1-4).

Marina, P., and Emanuele, F. 2020. Multidisciplinary pattern recognition applications: a review. Computer Science Review, 37, 100276.

Matthew, C., Castelfranco, A., Roncalli, V., Lenz, P, and Hartline, D. 2020. t-Distributed Stochastic Neighbor Embedding (t-SNE): A tool for eco-physiological transcriptomic analysis. Marine genomics, 51, 100723.

Mohammed, Sh. 2019. Design a Mobile Medication Dispenser based on IoT Technology. International Journal of Innovation, Creativity and Change, 6(2), 242-250.

Mohammed, Sh. 2021. Improving Coronavirus Disease Tracking in Malaysian Health System. Cihan University-Erbil Scientific Journal, 5(1), 11-19.

Mohammed, T., and Jose, C. 2020. Embeddings in natural language processing: Theory and advances in vector representations of meaning. Synthesis Lectures on Human Language Technologies, 13(4), 1-175.

Muhamad, A., Yaser, J., Mostafa, W., Mustafa S., and Sadeeer A. 2021. High-Performance Deep learning to Detection and Tracking Tomato Plant Leaf Predict Disease and Expert Systems.

Mustafa, A., Yaser, J., Tawfeeq, F., and Mustafa, S. 2021. On Announcement for University Whiteboard Using Mobile Application. CSRID (Computer Science Research and Its Development Journal), 12(1), 64-79.

Nasim, R., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., and Courville, A. 2019. On the spectral bias of neural networks. In International Conference on Machine Learning (pp. 5301-5310). PMLR.

Neethu, S., and Rajasree, R. 2013. Sentiment analysis in twitter using machine learning techniques. In 2013 fourth international conference on computing, communications and networking technologies (ICCCNT) (pp. 1-5). IEEE.

Neha, K., and Bhilare, P. 2015. An Approach for Sentiment analysis on social networking sites. In 2015 International Conference on Computing Communication Control and Automation (pp. 390-395). IEEE.

Nik, H., Prester, J., and Wagner, G. 2020. Seeking Out Clear and Unique Information Systems Concepts: A Natural Language Processing Approach. In ECIS.

Nikhil, B. 2017. Fundamentals of Deep Learning: Designing Next-generation Artificial Intelligence Algorithms/c Nikhil Buduma. Beijing, Boston, Farnham, Sebastopol, Tokyo: OReilly.

Niloufar, Sh., Shoeibi, N., Hernández, G., Chamoso, P., and Juan, C. 2021. AI-Crime Hunter: An AI Mixture of Experts for Crime Discovery on Twitter. Electronics, 10(24), 3081.

Peng, Ch., Sun, Z., Lidong B., and Wei, Y. 2017. Recurrent attention network on memory for aspect sentiment analysis. In Proceedings of the 2017 conference on empirical methods in natural language processing (pp. 452-461).

Priyanka, D., and Silakari, S. 2021. Deep learning algorithms for cybersecurity applications: A technological and status review. Computer Science Review, 39, 100317.

Rohit, K. 2018. Fake news detection using a deep neural network. In 2018 4th International Conference on Computing Communication and Automation (ICCCA) (pp. 1-7). IEEE.

Safwan, H., and Yaser, J. 2013. Diagnosis Windows Problems Based on Hybrid Intelligence Systems. Journal of Engineering Science & Technology (JESTEC), 8(5), 566-578.

Sagar, B., Doshi, A., Doshi, U., and Narvekar, M. 2014. A review of techniques for sentiment analysis of twitter data. In 2014 International conference on issues and challenges in intelligent computing techniques (ICICT) (pp. 583-591). IEEE.

Seyed-Ali, B., and Andreas, D. 2013. Sentiment analysis using sentiment features. In 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT) (Vol. 3, pp. 26-29). IEEE.

Sherry, G., Amer, E., and Gadallah, M. 2018. Deep learning algorithms for detecting fake news in online text. In 2018 13th international conference on computer engineering and systems (ICCES) (pp. 93-97). IEEE.

Thabit, Th., and Jasim, Y. 2017. Applying IT in Accounting, Environment and Computer Science Studies. Scholars' Press.

Thabit, Th., and Yaser, J. 2015. A manuscript of knowledge representation. International Journal of Human Resource & Industrial Research, 4(4), 10-21.

Thabit, Th., and Yaser, J. 2017. The role of social networks in increasing the activity of E-learning. In Social Media Shaping e-Publishing and Academia (pp. 35-45). Springer, Cham.

Xipeng, Q., Sun, T., Xu, Y., Shao, Y., Dai, N., and Huang, X. 2020. Pre-trained models for natural language processing: A survey. Science China Technological Sciences, 63(10), 1872-1897.

Yaser, J. 2018. Improving intrusion detection systems using artificial neural networks. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal, 7(1), 49-65.

Yaser, J., Mustafa, O., and Mustafa, S. 2021. Designing and implementation of a security system via UML: smart doors. CSRID (Computer Science Research and Its Development Journal), 12(1), 01-22.

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 11 N. 4 (2022), 451-474
eISSN: 2255-2863
DOI: https://doi.org/10.14201/adcaij.28395

Factors of Blockchain Adoption for FinTech Sector: An Interpretive Structural Modelling Approach

Somya Gupta[a] and Ganesh Prasad Sahu[b]

[a] School of Management Studies, Motilal Nehru National Institute of Technology Allahabad, India
[b] School of Management Studies, Motilal Nehru National Institute of Technology Allahabad, India
15somyagupta@gmail.com, gsahu@mnnit.ac.in

KEYWORDS

blockchain; payment systems; FinTech, Interpretive Structural Modelling (ISM); MICMAC

ABSTRACT

Blockchain Technology (BT) is rapidly becoming one of the most promising emerging economy innovations. Financial Technology (FinTech) has been disrupted by blockchain, and its market size is growing by the day. FinTech is closely associated to banking, and blockchain has become very famous in the banking industry. This study aims to analyse the factors influencing behavioural intention to adopt blockchain in FinTech. A total of 13 factors were extracted from the literature, and later relations among these variables were analysed using Interpretive Structural Modelling (ISM). The study's conceptual model was built and validated by academic experts working in blockchain. Later, MICMAC analysis was performed to study these variables' driving and dependence power. Although blockchain has various challenges, its implementation is recommended in FinTech due to the advantages it offers. As per our results, the implementation of blockchain in FinTech is required as it promotes data privacy and traceability and involves more trust than traditional means.

1. Introduction

Several prominent financial organisations have organised research teams to look into the suitability of blockchain technology for the finance and banking industries. According to a World Economic Forum (WEF) report 2016, more than "$1.4 billion has been invested in this technology in the recent past to explore and implement its use in the financial services industry" (Clauson et al., 2018). The

State Bank of India (SBI), India's largest government-controlled institution, is experimenting with blockchain-enabled smart contracts in a beta launch. This endeavour is supported by 27 banks who have grouped to investigate and develop blockchain-based banking solutions.

Currently, the financial services sector is adopting cutting edge technology. To service their business, these financial institutions will need to operate at a lower cost, with high volume payment lines, while also catering to the growing need for security and mobility. Institutional counterparties and financial service regulators are increasingly raising the demands for efficiency and transparency in business and technological processes. Over the past few years, digitalisation and cyber security have attracted investments from various banks and financial institutions, building test-to-scale approaches in trade finance, banking, and know your customer (KYC). The banking system is one of the key drivers of any country's economy, which helps mobilise resources and disburse cash in various spheres (Crookes and Conway, 2018; Guo and Liang, 2016). The banking system provides financial services such as exchanging deposits, providing multiple types of loans, etc.

On the other hand, the revenue collection system allows the government to manage its finances and provide its citizens with effective public services that foster economic growth. These systems are immensely affected by fraud and forgery, with critical cases. There is a need to secure the high-stake information resources that come with banking and e-governance systems; a part of national infrastructure that may be a target of intensive professional attacks (Sawant, 2011; Mills et al., 2016). It can eradicate and uproot corrupt practices in financial transactions, in corporate and other government machinery.

Many researchers have contributed to secure and authorised content retrieval or tamper-free access/storage methods to offset some significant factors. One such ground-breaking technology is blockchain. Since 2009, Bitcoin has operated without a central authority, making it the most valuable cryptocurrency by market capitalisation and the most well-known embodiment of the blockchain principle. It is based on the Peer-to-Peer (P2P) model. With a market valuation of almost $800 billion, over 18 million bitcoins are presently in circulation. The unspent transaction output set can be determined using a distributed record of transactions in Bitcoin (Tapscott and Tapscott, 2016). The financial industry investigated Bitcoin as a Distributed Ledger Technology (DLT).

1.1. Distributed Ledger Technology and Blockchain Technology

Distributed Ledger Technology (DLT) can be thought of as an umbrella term to designate blockchains and other distributed ledgers that operate in decentralised distributed environments spread across multiple participants in the network. DLT allows transactions and information to be recorded on shared and synchronised ledgers across a distributed network. Blockchain is one particular type of DLT.

Blockchain is an append-only distributed ledger, where records are stored in hash-linked blocks, and these blocks form a chain, as shown in Figure 1 (Casino et al., 2019; Niranjanamurthy et al., 2019; Yadav and Kushwaha, 2021). It consists of characteristics, such as distributed data storage, point-to-point transmission, consensus mechanisms, and encryption algorithms (Pereira et al., 2019; Yadav, Singh, et al., 2022).

A block in a blockchain is a data structure that aggregates and batches multiple transactions into a suitable size on the network. It is a collection of recent and verified transactions stored in the form of a Merkle tree (Pereira et al., 2019). The first block in a blockchain is the genesis block (Pereira et al., 2019).

Blockchain, in terms of financial technology, has recently received much attention. In the finance industry, the payments sector has been through various technical upgradations and it is also one of the

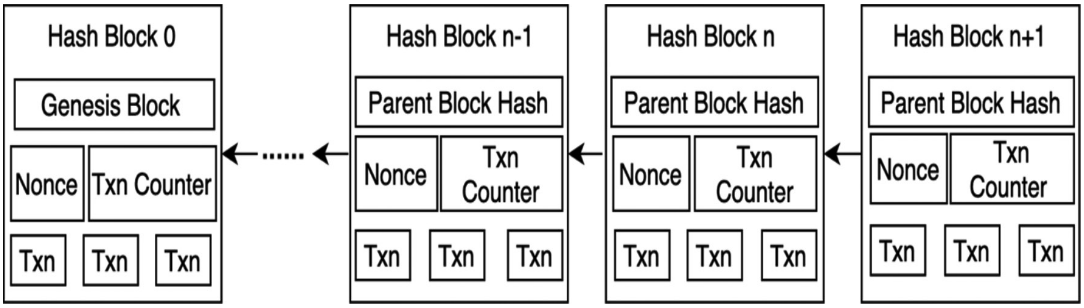


Figure 1. Blockchain consists of a continuous sequence of blocks

most crucial business areas for financial institutions. According to the Estonian government, using X-Road (a blockchain similar to middleware) and related process acceleration might save roughly 800 years of working time every year (Fridgen et al., 2018). In India, 11 big lenders launch the country's first blockchain-linked funding for Small and Medium Enterprises (SMEs) to deliver several benefits.

The potential cost savings that blockchain technology provides encourage financial institutions to take a closer look at this technology, and in some circumstances, this technology will actively promote their growth. Traditional financial technology must be redesigned in this era of increased competitiveness and security threats. To our knowledge, there has not been any research into the factors that influence blockchain adoption in FinTech in a hierarchical linkage.

The term "financial technology" can be analysed in two ways (Tapscott and Tapscott, 2017). The first dimension addresses traditional financial organisations (commercial banks, insurance companies) undertaking technological transformations. In contrast, the second aspect is the IT companies that provide innovative financial services utilising modern creative technologies which entice customers with a personalised experience through user-friendly, efficient, transparent, and automated products and services (i.e., peer to the peer lending, rob advisors, cryptocurrencies etc.). First dimension talks about modernising traditional means of finance, whereas second dimension considers FinTech companies leveraging innovative technologies for seamless delivery, personalization, agility, and relevance for customer experience.

1.2. FinTech

According to (Lee and Teo, 2015), FinTech is defined through five components: less profit margins, less number assets, expandability, innovative thinking, and simple compliance. The Financial Stability Board (FSB) 2017 of the United States (US) defined FinTech as technological innovations in various economic business models, applications, processes, or products. FinTech is an important aspect of Industry 4.0 since it necessitates the use and integration of multiple technologies such as AI and Data Science and delivering a platform as a service and software as a service. FinTech has piqued the interest of practitioners, investors, regulators, and advisors and researchers interested in learning more about how this novel technology and innovations affect enterprises.

In the FinTech sector, blockchain technology provides intelligent technology solutions, applications, and new market opportunities (Rosenberg, 2018). The current use of DLTs in finance is still very much in its infancy stage. However, the financial service industry's current business models are set to be reshaped by blockchain technology. Industry participants see an opportunity to integrate this technology into their goods and services to create organised strategies and help address current industry problems by increasing openness and enhancing behaviour. Blockchain is poised to become a mainstream technology with the ability to transform a wide range of sectors, including banking. The banking industry has begun to explore various blockchain applications in recent years. This technology is very much required in the FinTech sector.

The previous studies on blockchain and FinTech have considered various factors that led to blockchain adoption due to the advantages it offers, however, there are several barriers to blockchain adoption. (Boulos et al., 2018) described interoperability, transparency and security as blockchain adoption challenges. Whereas (Clohessy and Acton, 2019) discussed the characteristics of blockchain in both negative and positive aspects. For contrast, the authors also looked at the influence of organisational factors on blockchain adoption in organisations in a developed country. (Chang et al., 2020) described the challenges and ethical issues associated with blockchain adoption in the financial industry with interview methods. (Biswas and Gupta, 2019) discussed the characteristics of blockchain in the negative prospects, i.e., market-based risks, transaction-level uncertainties, high sustainability costs, poor economic behaviour, in the long run, privacy risk, legal and regularity uncertainties etc. Despite obstacles, blockchain is highly promising, as seen by the fact that numerous business models presently rely on decentralised finance i.e., decentralised currencies, payment systems, fundraising, lending, and contracting. Given the FinTech industry's fragmented and scattered grasp of the subject, academic study in this field is even more critical. As a result, the present study revolves around the following research questions:

Q1. What are the main inhibitors of blockchain adoption in the context of FinTech, particularly in India, and whether they are interlinked with each other?
Q2. Could it be possible to construct a framework based on the identified factors that may be utilised to increase blockchain adoption?

The purpose of this article is to thoroughly examine how blockchain technology can influence FinTech and analyse, on the basis of existing literature, various factors that lead to its adoption. This research's secondary objective is to determine the relationship between factors using literature and in-depth interviews with academic experts by adding reasoning and analysing the correlations between the discovered elements. The main objectives of this article are as follows:

1. To identify and investigate the factors that influence the adoption of blockchain technology in the FinTech industry.
2. To examine the interrelationships and interdependencies of the factors that have been determined.
3. To create a conceptual model of the factors that have been discovered.

In terms of theoretical contribution, this research analyses the contextual relationship between the variables that affect the adoption of blockchain technology in FinTech, using a systematic methodology. This study investigates the many correlations between the criteria chosen to deploy and adopt blockchain for FinTech. This study adds significantly to the present body of knowledge on blockchain and FinTech. In this study, thirteen factors have been identified that challenge the financial industry's adoption of blockchain in their business model. These discovered parameters were investigated further and used to create the ISM-MICMAC model.

The rest of the article is organised as follows. Section 2 briefs the literature in the area of blockchain and FinTech. This section also clearly explains all the factors derived from the literature.

Section 3 presents the proposed ISM-MICMAC methodology with a flowchart. Section 4 represents data collection and analysis with the formulation of the research model. Section 5 explains and analyses the results and subsection 5.1 discusses the theoretical contribution. Section 6 concludes this article and defines the scope of future research, followed by subsection 6.1 on implications for practice and policy.

2. Literature Review

As of today, blockchain is representative of an agile technology, as cryptocurrencies were the original application of blockchain, however, later its application extended to other fields, such as the supply chain, e-governance, healthcare systems, delivering various financial products and services etc. Blockchain is a technological breakthrough that has gained a great deal of attention all around the globe. Blockchain technology has been most directly connected with the Bitcoin cryptocurrency, first introduced in 2008 in a white paper by (Pereira et al., 2019). This technology has been described as having the ability to "shift market paradigms" (Guo and Liang, 2016), to "reverse the fortunes of the post-crisis financial sector" (Gomber et al., 2018), and to become the technology "most likely to transform the next decade of business" across all sectors (Till et al., 2017). Blockchain is a decentralised, distributed ledger that enables secure cryptocurrency transfers. Multiple and diverse groups are interested in the implementation of blockchain technology.

Blockchain cross-border transactions, explicitly correspondent banking, B2B transactions, and peer-to-peer money transfers could all benefit from blockchain. Yadav et al. proposed the blockchain framework that can reduce transaction costs and time by transmitting transactions directly from the sender to the recipient, bypassing various mediators (Yadav, Agrawal, et al., 2022; Yadav, Singh, et al., 2022). (Holotiuk et al., 2019) conducted a Delphi study on blockchain-related payment systems; on that basis, they identified how blockchain affects critical aspects of radical innovation. Whereas (Lindman et al., 2017) discussed the various issues of blockchain in payment systems. The authors focused on the organisational issues related to the competitive environment, and various technological issues. Later, they also discussed various themes within each of these areas.

In a recent article on DLT, the Bank for International Settlements' Committee on Payment and Market Infrastructures (CPMI) offered an analytical framework for investigating DLT applications in payments and settlements. Ripple is a blockchain-based payment network that focuses on commercial cross-border and inter-bank transfers. Various other blockchain platforms for payments are ABRA, which offers instant P2P money transfers without any transaction fee; another example of a blockchain-based payment system is Bitpesa. Bitpesa also provides cross-border payments to businesses and individuals between several African countries and China.

(Clohessy and Acton, 2019) used qualitative content analysis methodologies to determine the most important technology-organization-environment blockchain adoption determinants. Finally, the authors discovered three patterns: Top-level support and organisational preparation are critical facilitators for blockchain implementation. Small and medium-sized businesses are less likely to use blockchain than large corporations (SMEs) (Casino et al., 2019).

Under the title, abstract, and keywords, Scopus and Web of Science databases were searched with terms "FinTech" OR "Financial Technology" AND "Blockchain" AND "adoption." A total of 31 articles were chosen, by manually searching the relevant studies it was decided that they, matched the research's critical themes. All the factors and their possible correlations between the identified parameters were examined in the context of company performance following the acquisition of an affiliated firm based on previous studies. In this study, thirteen variables were considered. Using ISM-MICMAC approach, study create a hierarchical structure of identified factors of this study.

2.1. Factors Associated with Blockchain Adoption in FinTech

This study identified 13 variables of blockchain adoption in FinTech based on a survey of the literature. Experts were asked to rate the significance of these aspects on a Likert scale of [1–5], with "5" representing "not significant" and "1" alternatives specifying "extremely significant." All the factors, their implicit meanings, and the studies that support them are described below.

- **Scalability:** As the number of transactions increases, the blockchain grows more voluminous (Chang et al., 2020; Fridgen et al., 2018; Lacity, 2018; Reyna et al., 2018; Sharma et al., 2021; Yadav, Singh, et al., 2022). Blockchain transactions take longer to implement due to their complexity, encryption, and distributed nature, according to Marr (Bernard Marr, 2018a). Because all information, transactions, and balances are public, blockchain cannot prevent transactional information leakage. Visa conducts 24,000 transactions per second, PayPal 193 transactions per second, and Ethereum and Bitcoin only 20 transactions per second, according to Marr (Bernard Marr, 2018b). The reason for this is that blocks have a finite capacity, causing some small transactions to be delayed while miners prioritise transactions with high fees (Biais et al., 2019). The amount and frequency of blockchain records (or blocks) are limited, posing a scalability problem. Several blockchain adoption problems, such as scalability, were also identified in various other studies (Atlam et al., 2018; Lacity, 2018). Scalability was also cited by Reyna (Sawant, 2011) as a concern while deploying blockchain.
- **Security:** According (Zheng et al., 2018), blockchain-based systems are susceptible. Blockchain is subject to collusive self-centred miners, according to (Sharma et al., 2021), and many more attacks have proved that blockchain is insecure. Bitfinex, a bitcoin exchange, hacked 120,000 bitcoins on August 2nd, 2016, resulting in a $60 million loss. Finally, in 2018, a half-billion-dollar bitcoin theft was disclosed by a Japanese exchange. Although blockchain technology has shown great practicability and innovation in e-government, supply chain management, and capital markets, authorities are still grappling with the technology's immaturity (Cong and He, 2019). The use of blockchain in FinTech may be hampered by several concerns (endpoint risks, vendor concerns, full-scale testing, untested code etc.) (Chang et al., 2020; Wamba et al., 2020; Ji and Tia, 2021; Rana et al., 2022; Toufaily et al., 2021). According to (Alketbi et al., 2018), while blockchain promises to address various security concerns, such as secure data interchange and data integrity, it also adds new security threats that must be studied and managed. According to (Sawant, 2011; Kamel Boulos et al., 2018), blockchain confronts the same issues as any other technology that threatens conventional institutions, with security and privacy being the primary concerns.
- **Energy Consumption:** Although blockchain is one of the most popular technologies in recent years, it consumes a tremendous amount of energy (Chang et al., 2020; Truby, 2018). Blockchain is a peer-to-peer mechanism, which means no intermediaries are involved in the transaction. It necessitates many hash calculations to achieve the best outcomes, which requires even more significant energy consumption. As a result, it is common to encounter arguments that blockchain technology's energy consumption is a concern in general (Truby, 2018). Given the current debates over climate change and sustainability, these statements may hinder or delay the

broad implementation of blockchain technology (Biais et al., 2019). Big data applications can be more cost-effective in terms of execution and storage than long-term storage for electronic money transfers and transaction data (Stewart and Jürjens, 2017). The computational power required to execute blockchain is constantly rising. Based on Proof of Work consensus (PoW), the bitcoin system uses a lot of electricity, whereas Proof of Stake (PoS) is a more energy-efficient alternative to PoW. PoS is also more energy-efficient and effective. Indeed, a single bitcoin transaction necessitates a significant amount of electricity around terawatt-hour.

- **Regulations & Law:** With the expanding application of blockchain technology outside of cryptocurrency, regulators are paying considerably more attention to blockchain regulation to prevent fraud and other illegal behaviours that harm the interest of stakeholders and the market (Alketbi et al., 2018; Biais et al., 2019; Chang et al., 2020; Mary C. Lacity, 2018; Rana et al., 2022). The technological difficulty of blockchain is that it will never be able to guarantee the integrity of offline data, no matter how advanced the technology grows. Apart from the technical difficulties, determining which laws should be followed and which courts have the authority to decide on what matters for blockchain-related issues is a difficult and sometimes contradictory task because each node of a blockchain ledger could be in a different part of the world (Chang et al., 2020). Whereas in the case of FinTech, no clear rules or guidelines are available to control the financial transactions (Lacity, 2018), apart from the other challenges studied, management challenges, such as framed rules and regulations, are one of the major concerns for blockchain adoption.
- **Traceability of Transaction:** Traceability refers to the ability to determine where a product originated and follow its path through the whole transformation and distribution process (Samad et al., 2022; Shin, 2019). The specifics of every single transaction in the network can be recorded in an immutable manner due to DLT technology. These data are used to create audit trails, which authorised parties can inspect. The information provided by blockchain is trustworthy and cannot be doubted. The many users in the chain can notice flaws or threats in real-time using blockchain. One of the primary advantages of blockchain is traceability, which clarifies the benefits for people (Jonsson, 2018). Traceability and security are fundamental elements of operating blockchain systems; hence, they can be called technological affordances in blockchains (Rana et al., 2022; Shin, 2019).
- **Decentralisation:** Blockchain is a decentralised peer-to-peer computer network. Blockchain does not require a central control point or a single storage point because it is a decentralised system (Chang et al., 2020; Chen and Bellavitis, 2020; Ji and Tia, 2022; Kabir, 2021). Decentralisation also lowers the risk of a single point of failure. As the accounting activities are scattered over many distinct entities on the network, there is no need for a central organisation to execute smart contracts. According to (Zheng et al., 2018), each transaction in a normal centralised transaction system must be confirmed by a central trusted agent (such as the central bank). Without the need for a third party, any member of the blockchain can access the database and examine the transaction history (Till et al., 2017). The key benefit of this chain is that it has several copies of replicated data scattered across a distributed network. As a result, if a criminal or oppressive government wishes to remain undetected, they must upgrade all blockchain versions at the same time (Prasad et al., 2018).
- **Privacy/ Transparency:** For public blockchains, 'privacy poisoning' issues, in which recorded personal data violates privacy norms, are a problem (Biswas and Gupta, 2019; Chang et al.,

2020; Rana et al., 2022; Truby, 2018). It is challenging to strike a balance between an individual's right to privacy in an open network, especially on cross-border blockchain platforms. Blockchain technology can generate perpetual and unchangeable records for stakeholders, enhancing the privacy dangers for specific entities (Toufaily et al., 2021). Meanwhile, establishing confidentiality in open blockchain-based technologies is challenging since all network participants have access to information by default (Stewart and Jürjens, 2017). Users demand privacy, but transparency is necessary to identify ownership and eliminate double-spending (Casino et al., 2019; Shin, 2019). According to (Feng et al., 2019), participants' addresses, transaction values, timestamps, and sender signatures are all included in blockchain transactions, allowing data miners to track transaction flows and extract important information.

- **Leadership Readiness:** One of the main CSFs, according to IBM, is top-down executive support for new blockchain use cases (Prasad et al., 2018). Technology adoption requires organisational leadership, particularly in terms of championing breakthroughs. According to (Holotiuk et al., 2019), blockchain adoption is hampered by a lack of senior management expertise. Furthermore, because blockchain promotes a new way of doing business, it necessitates leadership to delegate responsibility to various nodes, most of which are anonymous and outside their control (Prasad et al., 2018; Truby, 2018). Executive leadership must also comprehend this new technology to bring out meaningful use cases from their organisation.
- **Robust & Mature Smart Contract:** Nick Szabo coined the phrase "smart contract" in a 1997 paper (Kabir, 2021; Tang and Veelenturf, 2019) to describe an automated legal agreement with automatic penalties. However, it was not until the Ethereum Foundation coined the phrase in 2013 to refer to programmatic code distributed and operated on the Ethereum DLT that it became widely accepted. A building block of Distributed Apps (DApps) on Ethereum is now referred to as a "smart contract" in the DLT community. Smart contracts are implemented differently than typical software programmes. The programmatic source code that is used to build intelligent contracts is stored in the DLT's immutable data storage.
- **Price Value:** The cost of installing, maintaining, and securing blockchain has a detrimental influence on its adoption and use in FinTech (Gomber et al., 2018; Rana et al., 2022). The price value is the cost associated with the adoption of a particular technology (Biswas and Gupta, 2019; Mishra and Kaushik, 2021; Rana et al., 2022; Truby, 2018). More secure, agile, and cost-effective structures can be offered thanks to the blockchain network, rather than insecure and costly present solutions (BDO, 2020).
- **Trust:** Teubner et al., 2016, 2017 conceptualise trust as "a consumer's willingness to rely on a host's actions and intentions, which can be further separated into the trusting beliefs of ability, integrity, and benevolence" as with the initial phase of blockchain, when implemented in bitcoin there were several cases of fraud and theft. Many businesses are concerned about the cost and efficiency of blockchain (Casino et al., 2019; Liu and Ye, 2021; Mishra and Kaushik, 2021; Sharma et al., 2021; Shin, 2019). Although blockchain technology can reduce expenses, it is nevertheless limited by traditional systems. Blockchain infrastructure development and initial investment are costly (Reyna et al., 2018). Smaller financial institutions and banks may choose not to invest in this technology (Lakhani and Iansiti, 2017). The adoption of blockchain technology is also hampered by high maintenance expenses (Casino et al., 2019). Several challenges must be overcome before this technique may be implemented. Knowledge-hiding, which was observed in the early days of adoption to prevent

institutions or management from succeeding, is one of the most challenging human-related issues with blockchain adoption (Chang et al., 2020).

- **Immutable:** Due to the immutability of blockchain, it is impossible to edit or amend transactions (Rana et al., 2022). Each block includes data, its hash, and the preceding block's hash. If someone modifies any of the blockchain's blocks, the hash will change, and subsequent blocks will fail to acknowledge the tempered block. Thus, tampering with a single block enables the observer to identify it without requiring the observer to examine each and every block. Due to blockchain's immutability, it can displace conventional financial intermediaries, and its benefits have been recognised across a range of financial industries (Chong et al., 2019; Till et al., 2017).
- **Behavioural Intension:** Behavioural intention has been repeatedly found to play a significant influence in determining the actual usage and acceptance of new systems in preceding IS/IT literature (Prasad et al., 2018; Rana et al., 2022; Shardeo et al., 2020). It has been investigated that the intention to use it can be considered the significant construct.

3. Methodology

In this research article, the ISM-MICMAC methodology has been used to develop theory and contribute to the literature in a realistic and context-rich manner. ISM was created by Warfield (Warfield, 1974) as an interactive learning tool that depends on expert judgement to link a phenomenon to multiple relevant factors (directly or indirectly). To describe the type and context-dependent interrelationships, ISM uses a hierarchical organisation of elements (Mathiyazhagan et al., 2013). The MICMAC approach determines a variable's driving and dependence power. The ISM-MICMAC combination has been extensively used in the literature (Agi and Nishant, 2017; Haleem et al., 2016), however, in literature on blockchain, the ISM methodology is quite limited to application on supply chain (Agi and Nishant, 2017; Rana et al., 2022; Samad et al., 2022; Sharma et al., 2021; Xu et al., 2021; Yadav and Kushwaha, 2022), but no ISM methodology applications have been found in relation to the adoption of blockchain in FinTech. The ISM-MICMAC based methodology was chosen over other exploratory research methodologies after a systematic evaluation of the literature and gaps form the literature. ISM- MICMAC was employed over other methodologies because of several advantages:

1. This systemic approach evaluates all possible pairwise correlations of system parts, whether through participant response or transitive extrapolation.
2. The reachability Matrix's use of transitive interpretation has boosted the efficiency of relational queries by 50-80%.
3. It directs and records the results of group discussions on complicated issues effectively and methodically.
4. Focusing specialists' attention on one single subject at a time improves the efficiency of multidisciplinary and interpersonal communication.
5. It is a learning tool for a better understanding of the meaning and importance of the listed factors and a well-accepted method in the literature for revealing contextual links in blockchain adoption.

Total interpretive structural modelling is a variation of the ISM-based technique (TISM). TISM interprets both nodes and connections in the digraph, whereas ISM only interprets nodes. To provide a better explanatory framework, TISM includes several critical transitive linkages, whereas ISM does not. ISM- MICMAC based flowchart is illustrated in Figure 2. According to (Rana et al., 2022) the ISM-MICMAC approach entails:

1. Using a literature survey and expert comments to identify the factors associated with the study topic (e.g., 13 critical factors of blockchain adoption in the FinTech sector).
2. Construct contextual linkages between factors through data collection, using paired comparisons and expert guidance, and produce a Structural Self-Interaction Matrix (SSIM) of the identified components.
3. Using SSIM to generate an initial "reachability matrix" (IRM) and then refining it with transitivity relations to generate a final "reachability matrix" (FRM) among factors.
4. Estimating each factor's driving power and dependencies by summing the data (FRM rows and columns) and establishing an FRM hierarchy using reachability and antecedent sets. The intersection set is derived by merging sets.

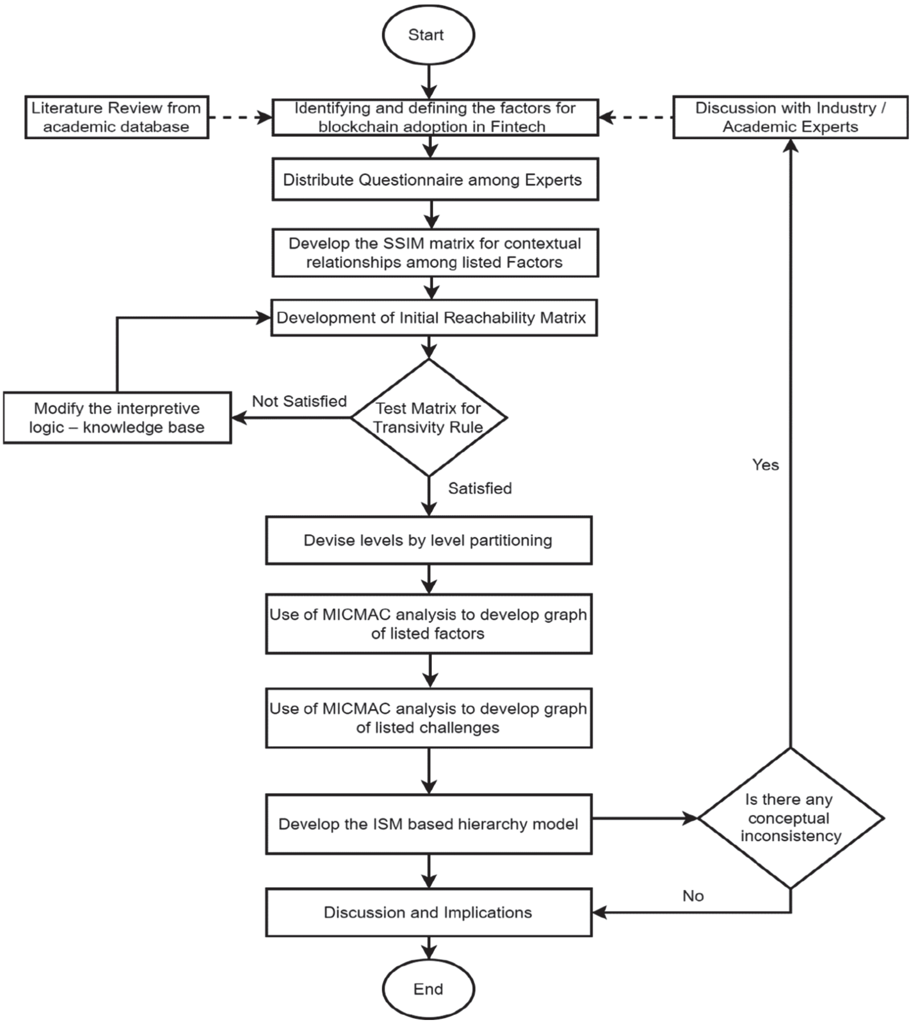


Figure 2. Interpretive Structural Modelling Process

5. Using the computed driving powers and factors dependencies, the following stage employs the MIC-MAC method to generate a graph of listed factors using the added driving powers and factors relationships. Expert inputs are grounded in four unique regions (autonomous, dependent, linkage, and drivers).
6. The next step is to use the reachability and intersection data sets to develop an ISM-based structural model. The outcome ISM model can be analysed for consistency (with expert input), and necessary actions can be advised to policymakers and financial institutions service providers and practising improvement facilitators.

4. Data Collection and Analysis

Experts from the FinTech and blockchain industry who met the required qualifications provided the data. This exercise was challenging because there was a limited pool of experts with this level of in-depth expertise, particularly those with practical experience in blockchain technology. According to our requirement, experts must comprehend blockchain technology and FinTech and demonstrate neutrality by not employing a software vendor or favouring a specific solution. A panel of experts from academic institute and bank personnel from India make up the panel. These specialists worked on blockchain implementation initiatives, and experts from educational institutions have also worked on various implementation projects. The information was gathered from eight experts. These experts have previously worked on a blockchain project and conducted research. The ISM-MICMAC technique was used to analyse the data. Table 1 presents the respondents' demographic summary regarding their job title, education, years of experience.

Table 1. Demographic Information of Experts

Demographic Classification	**Category**	**No. of Experts**
Job title	Academic member	5
	Industry	3
Year of Experience	0-5 Years	6
	10-15 Years	1
	15-20 Years	1

4.1. Selection of the Factors Relevant to Blockchain Adoption in FinTech

The initial literature study revealed certain factors influencing blockchain adoption in the FinTech sector, out of which the questionnaire confirmed 13 factors through expert opinion. A brainstorming session with an expert panel was held, and the challenges that hinder blockchain adoption in the public sector (from the literature and a "5" point Likert scale questionnaire) were discussed. The expert panel's subsequent debate and decision determined those factors with a rating of "3" or higher would be kept. The inclusion of the 13 literature-based determinants for blockchain adoption in FinTech was agreed upon and supported by all of the experts.

4.2. Development of SSIM, IRM and FRM

Pairwise comparisons were used to examine the collection of factors to find contextual links (direction) between them. The expert panel analysed the data from the questionnaire to identify the contextual relationship between each of the elements. Each expert was asked to score each difficulty to evaluate how closely it linked to other challenges. The relationships were coded with pre-existing symbols. "A" – when factor "i" will lead to factor "j"; (2) "B" – when factor "j" will lead to factor "i" (3) "C" – when factors "i" and "j" will facilitate each other; and (4) "D" – when factors "i" and "j" are unrelated to each other. The SSIM results are shown in Table 2, where each expert's perspective on interdependence is depicted using the required notation.

Table 2. SSIM for Critical Factors of Blockchain Adoption in FinTech

i/j	Variables	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1
F1	**Scalability**	A	D	A	A	B	A	B	C	D	D	A	D	$
F2	**Security**	A	C	C	A	B	C	C	C	C	B	D	$	-
F3	**Energy Consumption**	A	B	D	A	B	A	D	B	B	B	$	-	-
F4	**Regulations & Law**	A	A	A	D	C	A	A	D	A	$	-	-	-
F5	**Traceability of Transactions**	A	B	A	D	A	A	D	B	$	-	-	-	-
F6	**Decentralisation**	A	A	A	A	B	A	A	$	-	-	-	-	-
F7	**Privacy / Transparency**	A	C	A	D	B	A	$	-	-	-	-	-	-
F8	**Leadership Readiness**	A	D	B	D	B	$	-	-	-	-	-	-	-
F9	**Robust and Mature Smart Contract**	A	A	A	D	$	-	-	-	-	-	-	-	-
F10	**Perceived Value**	A	D	D	$	-	-	-	-	-	-	-	-	-
F11	**Trust**	A	B	$	-	-	-	-	-	-	-	-	-	-
F12	**Immutable**	A	$	-	-	-	-	-	-	-	-	-	-	-
F13	**Behavioural Intension**	$	-	-	-	-	-	-	-	-	-	-	-	-

Following the specified rules, the above information in Table 2 as compiled SSIM is a reachability matrix. This is accomplished by substituting A, B, C, and D with 1 or 0, depending on the case. The SSIM is converted into the reachability matrix using the following rules illustrated in Table 3:

1. If the SSIM (i, j) entry is A, the reachability matrix (i, j) entry becomes 1 and the (j, i entry) becomes 0.
2. If the SSIM (i, j) entry is B, the reachability matrix (i, j) entry becomes 0, and then (j, i) entry becomes 1.
3. If the (i, j) entry in the SSIM is C, the (i, j) entry in the reachability matrix becomes 1 as well as the (j, i) entry also become 1.
4. If the (i, j) item in the SSIM is D, the (i, j) entry in the reachability matrix becomes 0 and even the (j, i) entry also become 0.

The application of transitivity is required for the changeover from IRM to FRM. Table 4 displays the results. The following are examples of transitivity: If A is connected to B (A → B) and B is connected to C (B → C), then A and C (A →C) have a transitive relationship. Within the FRM, transitive relationships are denoted by emphasising "1*" for each transitive relationship instance. Each challenge's driving and dependence power are computed, and a sum of the (i, j) entries within the FRM is quantified.

Table 3. Initial Reachability Matrix for Blockchain Adoption in FinTech

i/j	Variables	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	Driving power
F1	Scalability	1	0	1	0	0	1	0	1	0	1	1	0	1	**7**
F2	Security	0	1	0	0	1	1	1	1	1	1	1	1	1	**10**
F3	Energy Consumption	0	0	1	0	0	0	0	1	0	1	0	0	1	**4**
F4	Regulations &Law	0	1	1	1	1	0	1	1	1	1	1	1	1	**11**
F5	Traceability of Transactions	0	1	1	1	1	0	0	1	1	0	1	0	1	**12**
F6	Decentralisation	1	1	1	0	1	1	1	1	0	1	1	1	1	**11**
F7	Privacy / Transparency	1	1	0	0	0	0	1	1	0	0	1	1	1	**7**
F8	Leadership Readiness	0	1	0	0	0	0	0	1	0	0	0	0	1	**3**
F9	Robust and Mature Smart Contract	1	1	1	1	0	1	1	1	1	0	0	1	1	**10**
F10	Perceived Value	0	0	0	0	0	0	0	0	0	1	0	0	1	**2**
F11	Trust	0	0	0	0	0	0	0	1	0	0	1	0	1	**3**
F12	Immutable	0	1	1	0	1	0	1	1	0	1	0	1	1	**8**
F13	Behavioural Intension	0	0	0	0	0	0	0	0	0	0	0	0	1	**1**
	Dependence Power	**4**	**8**	**6**	**3**	**5**	**4**	**6**	**11**	**4**	**7**	**7**	**6**	**13**	

Table 4. Final Reachability Matrix for Blockchain Adoption in FinTech

i/j	Variables	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	Driving power
F1	Scalability	1	0	1	0	0	1	0	1	0	1	1	1*	1	**8**
F2	Security	**1***	1	**1***	0	1	1	1	1	1	1	1	1	1	**12**
F3	Energy Consumption	0	0	1	0	0	0	0	1	0	1	0	0	1	**4**
F4	Regulations & Law	**1***	1	1	1	1	**1***	1	1	1	1	1	1	1	**13**
F5	Traceability of Transactions	**1***	1	1	1	1	**1***	**1***	1	1	0	1	1*	1	**12**
F6	Decentralisation	1	1	1	0	1	1	1	1	0	1	1	1	1	**11**
F7	Privacy / Transparency	1	1	0	0	**1***	**1***	1	1	0	0	1	1	1	**9**
F8	Leadership Readiness	0	1	0	0	0	0	0	1	0	0	0	0	1	**3**
F9	Robust and Mature Smart Contract	1	1	1	1	**1***	1	1	1	1	0	0	1	1	**11**
F10	Perceived Value	0	0	0	0	0	0	0	0	0	1	0	0	1	**2**
F11	Trust	0	0	0	0	0	**1***	0	1	0	0	1	0	1	**4**
F12	Immutable	**1***	1	1	0	1	0	1	1	0	1	0	1	1	**9**
F13	Behavioural Intension	0	0	0	0	0	0	0	0	0	0	0	0	1	**1**
	Dependence Power	**8**	**8**	**8**	**3**	**7**	**8**	**7**	**11**	**4**	**7**	**7**	**8**	**13**	**99/99**

4.3. Level Partitioning

The matrices are then split into importance levels to form a "causal" hierarchical structure. This is accomplished by applying reachability, antecedent, and intersection sets. Reachability entails analysing each factor and the other highlighted elements that could be altered. On the other hand, the antecedent is made up of the elements as well as any other variables that may affect it. The intersection set is discovered by combining the reachability and antecedent sets, which is done for all factors. The factors on level 1 were achieved when the reachability and intersection sets were equal for every factor (for example, "behavioural intension to adopt blockchain in FinTech (F13)"). The recording of a factor at any level was deleted, and the cycle was repeated until all factors were exhausted. Five iterations, illustrated in Table 5, were carried out, yielding the ISM-based factor model for blockchain adoption in the FinTech context and the importance levels.

Table 5. Level Partitioning Matrix

Element	**Reachability Set**	**Antecedent Set**	**Intersection Set**	**Level**
		(a): Iteration- 1		
F1	1,3,6,8,11,12,13	1,2,4,5,6,7,9,12	1,6,12	
F2	1,2,3,5,6,7,8,9,10,11,12,13	2,4,5,6,7,8,9,12	2,5,6,7,8,9,11,12	
F3	3,8,10,13	1,2,3,4,5,6,9,12	3	
F4	1,2,3,4,5,6,7,8,9,10,11,12,13	4,5,9	4,5,9	
F5	1,2,3,4,5,6,7,8,9,11,12,13	2, 4,5,6,7,9,12	2,4,5,6,7,9,12	
F6	1,2,3,5,6,7,8,10,11,12,13	1,2,4,5,6,7,9,11	1,2,5,6,7,11	
F7	1,2,5,6,7,8,11,12,13	2,4,5,6,7,9,12	2,5,6,7,12	
F8	2,8,13	1,2,3,4,5,6,7,8,9, 11, 12	2,8	
F9	1,2,3,4,5,6,7,8,9,12,13	2,4,5,9	2,4,5,9	
F10	10,13	1,2,3,4,6,10,12	10	
F11	6,8,11,13	1,2,4,5,6,7,11	6,11	
F12	1,2,3,5,7,8,10,12,13	1,2,4,5,6,7,9,12	1,2,5,7,12	
F13	13	1,2,3,4,5,6,7,8,9,10,11,12,13	13	**Level I**
		(b): Iteration- 2		
F1	1,3,6,8,11,12	1,2,4,5,6,7,9,12	1,6,12	
F2	1,2,3,5,6,7,8,9,10,11,12	2,4,5,6,7,8,9,12	2,5,6,7,8,9,11,12	
F3	3,8,10	1,2,3,4,5,6,9,12	3	
F4	1,2,3,4,5,6,7,8,9,10,11,12	4,5,9	4,5,9	
F5	1,2,3,4,5,6,7,8,9,11,12	2, 4,5,6,7,9,12	2,4,5,6,7,9,12	
F6	1,2,3,5,6,7,8,10,11,12	1,2,4,5,6,7,9,11	1,2,5,6,7,11	
F7	1,2,5,6,7,8,11,12	2,4,5,6,7,9,12	2,5,6,7,12	
F8	2,8	1,2,3,4,5,6,7,8,9, 11, 12	2,8	**Level II**
F9	1,2,3,4,5,6,7,8,9,12	2,4,5,9	2,4,5,9	
F10	10	1,2,3,4,6,10,12	10	
F11	6,8,11	1,2,4,5,6,7,11	6,11	
F12	1,2,3,5,7,8,10,12	1,2,4,5,6,7,9,12	1,2,5,7,12	
		(d): Iteration-3		
F1	1,3,6,11,12	1,2,4,5,6,7,9,12	1,6,12	**Level III**
F2	1,2,3,5,6,7,9,10,11,12	2,4,5,6,7,9,12	2,5,6,7,9,11,12	

Table 5. Level Partitioning Matrix (Continiue)

Element	Reachability Set	Antecedent Set	Intersection Set	Level
F3	3,8,10	1,2,3,4,5,6,9,12	3	**Level III**
F4	1,2,3,4,5,6,7,9,10,11,12	4,5,9	4,5,9	
F5	1,2,3,4,5,6,7,9,11,12	2, 4,5,6,7,9,12	2,4,5,6,7,9,12	
F6	1,2,3,5,6,7,10,11,12	1,2,4,5,6,7,9,11	1,2,5,6,7,11	**Level III**
F7	1,2,5,6,7,11,12	2,4,5,6,7,9,12	2,5,6,7,12	
F9	1,2,3,4,5,6,7,9,12	2,4,5,9	2,4,5,9	
F10	10	1,2,3,4,6,10,12	10	
F11	6,11	1,2,4,5,6,7,11	6,11	
F12	1,2,3,5,7,10,12	1,2,4,5,6,7,9,12	1,2,5,7,12	**Level III**
		(e): Iteration -4		
F2	2,5,7,9,10,11	2,4,5,7,9	2,5,7,9	**Level IV**
F4	2,4,5,6,7,9,10,11	4,5,9	4,5,9	
F5	2,4,5,7,9,11	2,4,5,7,9	2,4,57,9	**Level IV**
F7	2,5,7,11	2,4,5,7,9	2,5,7	**Level IV**
F9	2,4,5,7,9	2,4,5,9	2,4,5,9	
F10	10	2,3,4,10	10	
F11	11	2,4,5,7,11	11	**Level IV**
		(f): Iteration -5		
F4	4,9,10	4,9	4,9	**Level V**
F9	4,9	4,9	9	**Level V**
F10	10	4,10	10	**Level V**

4.4. MICMAC Analysis

The MICMAC analysis is used to determine the driving and reliance powers. To obtain these values, the FRM was examined, and the accumulation of the rows and columns of the FRM was determined and described in Table 4. Figure 3 depicts the MICMAC plot. The structural analysis of MICMAC generates factor sets (autonomous, dependent, linkage and driver). This is useful in practice as it provides a wealth of information and in-depth insight into the causes and implications of the primary problems associated with blockchain implementation and adoption in FinTech. The four sets consist of:

1. **Autonomous set (I quadrant):** A collection of items with a low driving and reliance power (lower left quadrant), but with a negligible system impact. It has less influence on the adoption process. However, no such component was detected in the current investigation.
2. **Dependent set (II Quadrant):** This set of factors has a low driving force but a strong dependence force (lower right quadrant) and is, therefore, more essential in the model. These variables reflect output-dependent variables throughout the system. Five characteristics were identified: "behavioural intention to use blockchain in FinTech (F13)", "Trust (F11)", "Perceived Value (F10)", "Energy Consumption (F3)", and "Leadership Readiness (F8)".
3. **Linkage set (III Quadrant):** provides a lot of driving and dependence power (upper right quadrant). In ISM hierarchical models, it has a lower priority level, and the following factors were found in this category: "Robust and Mature Smart Contract (F9)", and "Regulations and Law (F4)". These variables are less stable, and practitioners should observe them at all times. These factors are

essential since they are highly dependent on others, and practitioners must nurture the supporting aspects to attain these output factors and enable remarkable development achievement.

4. **Independent set (IV Quadrant):** This set is the foundation for successful improvement since it has high driving and low dependence power (upper left quadrant). "Scalability (F1)", "Security (F2)", "Transaction Traceability (F5)", "Decentralisation (F6)", "Privacy/Transparency (F7)" and "Immutability (F12)" are among the factors. Practitioners must evaluate these key driving factors to ensure that they have the least impact on the other aspects.

4.5. Development of ISM-based Model

After the MICMAC study, the FRM was utilised to form the ISM model (using nodes/vertices and lines of edges), which is now known as a digraph. The transitivity linkages are deleted, and the ISM model assigns nodes. The model depicts factors at all levels and their links to factors at the same and higher levels.

The levels in Figure 3 are also related to four quadrants of the MICMAC diagram, with the primary driving problems all described in the MICMAC diagram's independent quadrant. These factors have a lot of driving power, and they cause other factors in the model. None of the factors requires a lot of driving and depending on others. This suggests that the problems are not random. The dependent factors are presented in the second quadrant. These issues have a low driving power but a high dependence power. Therefore, they are influenced by interdependent factors with larger driving power. All factors with a high degree of dependence and driving power are presented in the third quadrant. In the ISM model,

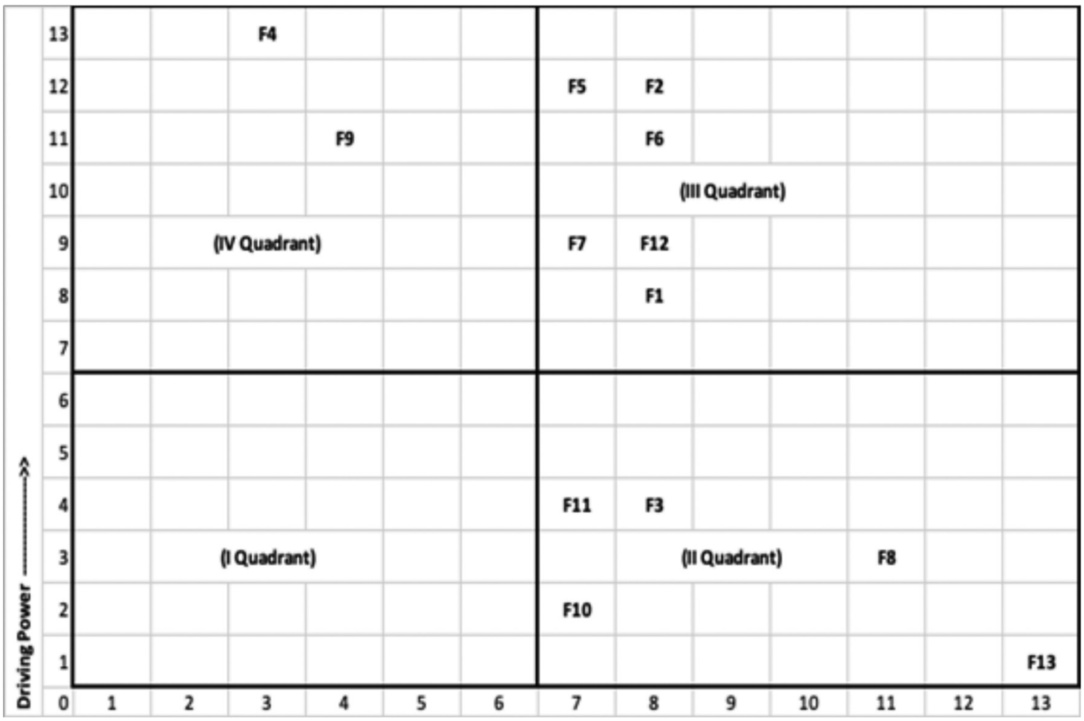


Figure 3. MICMAC Analysis

these factors are generally found between the top and bottom layers of variables. They effectively serve as mediators between the primary driving and dependent factors. All problems with high driving and low dependent power are included in the fourth and final quadrant. These issues mostly affect the bottom half of the ISM model, posing additional factors in the suggested paradigm illustrated in Figure 4.

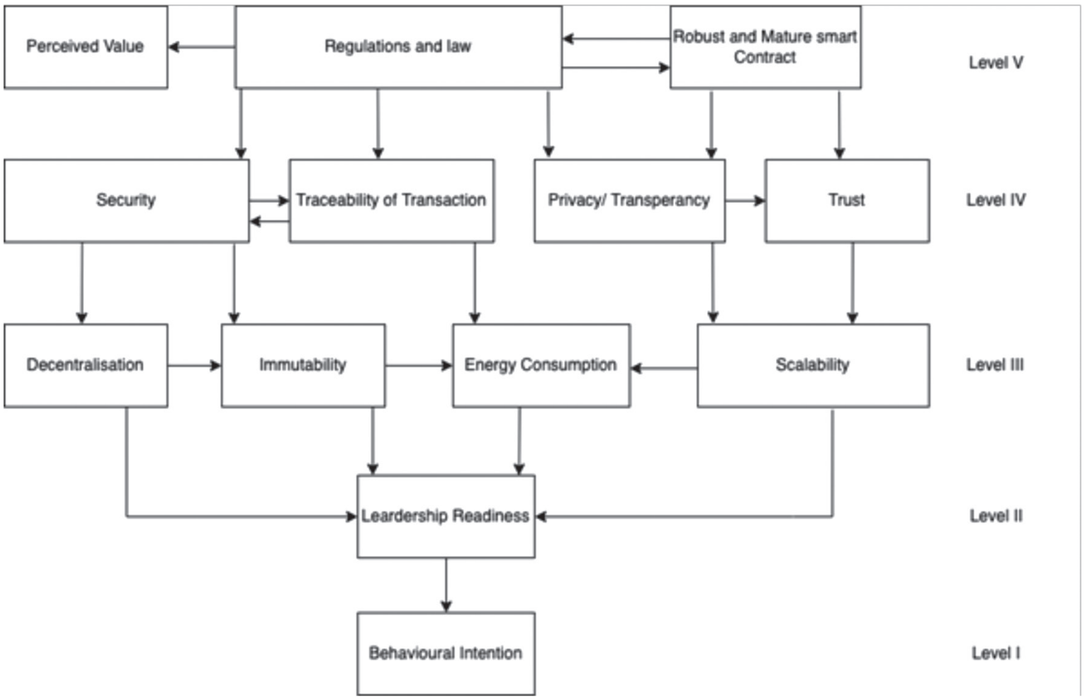


Figure 4. ISM Based Research Model

5. Discussion and Analysis of ISM based Model

This study depicts the growing demand for blockchain technology and the worldwide trend toward more innovative FinTech transactions. Other variables for blockchain adoption in FinTech tech are "Perceived Value (F10)", "Regulation and Legislation (F4)", and "Robust and Mature Smart Contract (F9)", according to the ISM model shown in Figure 3. This indicates that the price included in implementing blockchain in the financial sector plays a significant role with well-drafted regulations and laws. Implementing blockchain requires a substantial initial cost, whereas new blocks carry certain charges. There is also an increase in smart contracts in daily digital transactions, and the inclusion of intelligent contracts in financial blockchain makes it more robust (Sharma et al., 2021). Testing and deploying blockchain pilots successfully can impact how the technology is embraced and deployed in FinTech.

FinTech was classified into two forms, innovating traditional forms and introducing new forms of technical financial services. So out of the various factors selected for the study, both would help FinTech service providers. The interoperability of blockchain technology should be established with the existing systems and new ventures to ensure the seamless functioning of the new ventures and encourage innovation within the traditional systems. In contrast, blockchain is being integrated with FinTech. The government should frame regulations and laws to validate the implementation of blockchain in the delivery of financial services. The lack of regulations and laws is a serious concern among early and new adopters of blockchain in FinTech. These three factors further lead to "Security (F2)", "Traceability of Transactions (F5)", "Privacy / Transparency (F7)", "Trust (F11)".

These hierarchical relationships make perfect sense as regulations and law could lead to security and traceability of transactions, enhancing privacy and transparency. In contrast, robust and mature smart contracts in financial blockchains can also improve privacy/transparency and trust. Even though blockchain technology provides a highly secure method of exchanging payments via a network, the lack of regulations and laws can cause these relationships to act reversibly. Traditional financial institutions should ensure standard rules and law by the central governing body to implement blockchain in their existing systems to become more secure, transparent, and trustworthy. While executing smart contracts, there is no need for a central authority or third party to support these transactions because they are automatically executed when particular circumstances are met, further enhancing trust among the users.

The factors (i.e., security, traceability of transaction, privacy/ transparency and trust) in blockchain implementation illustrated at level 4 lead to factors at the next level, including "Decentralisation (F6)", "Immutability (F12)", "Energy Consumption (F3)" and "Scalability (F1)". The integration of these factors leads to any primary blockchain. Blockchain can be understood as a decentralised ledger, storing transaction records on multiple computers simultaneously, further enhancing the security and transparency of the data stored on it. Blockchain requires a huge electricity consumption, and abundant energy supplies can require using or adopting blockchain in FinTech. Blockchain becomes voluminous with the ever-increasing number of transactions, so scalability is a concern for blockchain, as every block addition consumes tremendous energy and requires more storage. However, various consensus algorithms such as Proof-of-Stake (PoS), IOTW's Proof-of-Assignment (PoA) Protocol have reduced both, the computational power and the risk of memory loss, making the process much easier.

These four factors further lead to "Leadership Readiness (F8)". According to (Holotiuk et al., 2019) blockchain adoption is hampered by a lack of top-level management skills. Leadership (CEO influence, opinion leadership, and top management support) is identified by several theoretical frameworks as a vital component in adopting innovations (Till et al., 2017). All above levels of factors have an impact on leadership readiness as leadership readiness takes into consideration all the primary level III (F6, F12, F3 and F1), security features level IV (F2, F5, F7 and F11) and regulatory features at level V (F4, F9 and F10) to adopt and implement blockchain in FinTech. On the other hand, these elements represent a mix of problems and benefits for implementing blockchain in FinTech. According to the performed analysis, energy consumption and scalability is on level II of the model, but it is also one of the drawbacks of blockchain usage in Fintech. Blockchain is an energy-intensive new technology, and because financial blockchains grow larger with every single transaction, scalability is also a significant concern. Besides these disadvantages, blockchain has several advantages, i.e., perceived value, security, mature smart contracts, traceability of transactions, privacy/ transparency, trust, decentralisation and immutability. Whereas leadership readiness can be identified as a neutral construct since it is challenging to apply innovative technology for any top management despite the technology's benefits, determining whether or not an institution is prepared to accept that technology is complex.

5.1. Theoretical Contributions

This research makes several theoretical contributions. This is the first study of its kind to compile a diverse collection of factors for blockchain adoption in FinTech and review related literature. Using the ISM-MICMAC technique, this study takes a step ahead in establishing links between various elements. The conceptual framework offers us a plethora of data regarding these issues and defines clear categories for them, making driving and dependant factors easier to comprehend. This would help researcher better grasp the driving factors and the other factors reliant on these drivers if they were classified in this way. Researchers can also use the ISM-based approach to understand factors that have both driving and dependant aspects. This proposed framework clarifies how various factors considered for blockchain adoption in FinTech are linked and interrelated, as depicted in the ISM-based model.

6. Conclusion and Future Work

The study identifies a unique set of factors for implementing blockchain technology in FinTech. This article explores 13 such factors and the projected value of blockchain adoption in FinTech using the ISM-MICMAC-Based technique based on a qualitative research design. According to the findings, factors such as "Price Value (F10)", "Regulations and Law (F4)", and "Robust and Mature Smart Contract (F9)" are the main roadblocks to blockchain implementation in FinTech. Similarly, "Leadership Readiness (F8)" is largely dependent. The findings of the ISM and MICMAC approaches provide a significant knowledge of the hierarchies of various factors and the power that drives and depends on them.

Further on, researchers can also extend ISM-MICMAC with TISM to estimate these key factors. Whereas this research does not classify factors and independent and dependent factors, further research can explore factors with cause-and-effect relationships. Future researchers could operationalise the constructs in the proposed model and assess its validity.

6.1. Implications for Practice and Policy

The implementation of blockchain requires enormous capital investment, as using a new technology requires initial investments for both businesses and consumers, including learning costs linked to getting acquainted with the system in general and the adoption of the technology in particular. However, when it comes to transactions and payments, the cost of blockchain adoption is an investment which gives long-term efficiency benefits. Additionally, blockchain requires a significant amount of electricity, and an abundant energy supply, which may prevent its use or deployment. So, before getting aligned with blockchain technology, the organisation should consider the benefits and limitations of adopting this technology. In terms of energy consumption, blockchain will require high-end computers and abundant energy supplies to make safe and secure transactions and run thousands of calculations per second.

Like bitcoin, other financial blockchains will also require intelligent algorithms to operate reliably every second behind the scenes. Dedicated teams with diverse expertise must ensure blockchain deployments and services perform well. High-level encryption algorithms, the removal of personal identifiers, a mix of passwords and biometric verification, and specific access control can all help make blockchain application safer in FinTech. Clients can use analytics functions to run simple blockchain requests behind the scenes. The procedures are convenient to carry out. Outcomes can be achieved in a matter of minutes. For the effective implementation of blockchain the FinTech sector should also consider the law and regulatory compliance related to this technology for better adoption. Organisations implementing blockchain must train their employees well, to keep up with current advancements in blockchain technology and deal with the rising complexity of this technology over time. This will also assist them in overcoming their apprehensions about using more modern technologies, to improve work productivity.

Funding and Informed

No funding was received.

Author Contribution

Somya Gupta study design performed the experiments and wrote the manuscript; Writing – Review & editing Ganesh Prasad Sahu. All authors have read and agreed to the published version of the manuscript.

Data Availability

The study does not report any data.

Ethical Statement

The manuscript in part or in full has not been submitted or published anywhere.

Conflicts of Interest

The authors declare no conflict of interest.

7. References

Agi, M.A. and Nishant, R., (2017). Understanding influential factors on implementing green supply chain management practices: An interpretive structural modelling analysis. *Journal of environmental management*, *188*, pp.351-363. https://doi.org/10.1016/j.jenvman.2016.11.081

Alketbi, A., Nasir, Q. and Talib, M.A., (2018). Blockchain for government services—Use cases, security benefits and challenges. *2018 15th Learning and Technology Conference (L&T)*, 112-119. https://doi.org/10.1109/LT.2018.8368494

Atlam, H. F., Alenezi, A., Alassafi, M. O., & Wills, G., (2018). Blockchain with internet of things: Benefits, challenges, and future directions. *International Journal of Intelligent Systems and Applications*, *10*(6), 40-48. https://doi.org/10.5815/ijisa.2018.06.05

Bernard Marr. (2018a). *Blockchain and the internet of things: 4 important benefits of combining these two mega trends*.

Bernard Marr. (2018b). *The 5 big problems with blockchain everyone should be aware of*.

Biais, B., Bisiere, C., Bouvard, M. and Casamatta, C. (2019). The blockchain folk theorem. *The Review of Financial Studies*, *32*(5), 1662-1715. https://doi.org/10.1093/rfs/hhy095

Biswas, B., & Gupta, R. (2019). Analysis of barriers to implement blockchain in industry and service sectors. *Computers & Industrial Engineering*, *136*, 225-241. https://doi.org/10.1016/j.cie.2019.07.005

Casino, F., Dasaklis, T.K. and Patsakis, C., 2019. A systematic literature review of blockchain-based applications: Current status, classification and open issues. *Telematics and informatics*, *36*, 55-81. https://doi.org/10.1016/j.tele.2018.11.006

Chang, V., Baudier, P., Zhang, H., Xu, Q., Zhang, J., & Arami, M., 2020. How Blockchain can impact financial services–The overview, challenges and recommendations from expert interviewees. *Technological forecasting and social change*, *158*, 1201-66. https://doi.org/10.1016/j.techfore.2020.120166

Chen, Y., & Bellavitis, C. (2020). Blockchain disruption and decentralized finance: The rise of decentralized business models. *Journal of Business Venturing Insights*, *13*, e00151. https://doi.org/10.1016/j.jbvi.2019.e00151

Chong, A.Y.L., Lim, E.T., Hua, X., Zheng, S. and Tan, C.W. (2019). Business on chain: A comparative case study of five blockchain-inspired business models. *Journal of the Association for Information Systems*, 1308-1337. https://doi.org/10.17705/1jais.00568

Clauson, K.A., Breeden, E.A., Davidson, C. and Mackey, T.K. (2018). Leveraging Blockchain Technology to Enhance Supply Chain Management in Healthcare: An exploration of challenges and opportunities in the health supply chain. *Blockchain in healthcare today*. https://doi.org/10.30953/bhty.v1.20

Clohessy, T. and Acton, T. (2019). Investigating the influence of organizational factors on blockchain adoption: An innovation theory perspective. *Industrial Management & Data Systems*, *119*(7), 1457–1491. https://doi.org/10.1108/IMDS-08-2018-0365

Cong, L. W., & He, Z. (2019). Blockchain disruption and smart contracts. *The Review of Financial Studies*, *32*(5), 1754-1797. https://doi.org/10.1093/rfs/hhz007

Crookes, L. and Conway, E. (2018). Technology challenges in accounting and finance. In *Contemporary issues in accounting* (pp. 61-83). Springer International Publishing. https://doi.org/10.1007/978-3-319-91113-7_4

Feng, Q., He, D., Zeadally, S., Khan, M.K. and Kumar, N. (2019). A survey on privacy protection in blockchain system. *Journal of Network and Computer Applications*, *126*, 45-58. https://doi.org/10.1016/j.jnca.2018.10.020

Fosso Wamba, S., Kala Kamdjoug, J.R., Epie Bawack, R. and Keogh, J.G. (2020). Bitcoin, Blockchain and Fintech: a systematic review and case studies in the supply chain. *Production Planning & Control*, *31*(2-3), 115-142. https://doi.org/10.1080/09537287.2019.1631460

Fridgen, G., Radszuwill, S., Urbach, N. and Utz, L. (2018). Cross-organizational workflow management using blockchain technology: towards applicability, auditability, and automation. In *51st Annual Hawaii International Conference on System Sciences (HICSS)*.

Gomber, P., Kauffman, R.J., Parker, C. and Weber, B.W. (2018). On the fintech revolution: Interpreting the forces of innovation, disruption, and transformation in financial services. *Journal of management information systems*, *35*(1), 220-265. https://doi.org/10.1080/07421222.2018.1440766

Guo, Y. and Liang, C. (2016). Blockchain application and outlook in the banking industry. *Financial innovation*, *2*(1), 24. https://doi.org/10.1186/s40854-016-0034-9

Haleem, A., Luthra, S., Mannan, B., Khurana, S., Kumar, S. & Ahmad, S. (2016). Critical factors for the successful usage of fly ash in roads & bridges and embankments: Analyzing Indian perspective. *Resources Policy*, *49*, 334-348. https://doi.org/10.1016/j.resourpol.2016.07.002

Holotiuk, F., Pisani, F. & Moormann, J. (2019). Radicalness of blockchain: an assessment based on its impact on the payments industry. *Technology Analysis & Strategic Management*, *31*(8), 915-928. https://doi.org/10.1080/09537325.2019.1574341

Ji, F. & Tia, A. (2021). The effect of blockchain on business intelligence efficiency of banks. *Kybernetes, 51*(8), 2652–2668. https://doi.org/10.1108/K-10-2020-0668

Jonsson, J.R. (2018). *Perceived affordance and socio-technical transition: blockchain for the swedish public sector*. KTH Royal Institute of Technology School of Industrial Engineering and Management.

Juho Lindman, Virpi Kristiina Tuunainen, & Matti Rossi. (2017). Opportunities and Risks of Blockchain Technologies – A Research Agenda. *Hawaii International Conference on System Sciences (HICSS)*.

Kabir, M.R., (2021). Behavioural intention to adopt blockchain for a transparent and effective taxing system. *Journal of Global Operations and Strategic Sourcing, 14*(1), 170–201. https://doi.org/10.1108/JGOSS-08-2020-0050

Kamel Boulos, M.N., Wilson, J.T. & Clauson, K.A. (2018). Geospatial blockchain: promises, challenges, and scenarios in health and healthcare. *International Journal of Health Geographics*, *17*(1), 25. https://doi.org/10.1186/s12942-018-0144-x

Lee, D.K.C. & Teo, E.G. (2015). Emergence of Fintech and the Lasic Principles. *SSRN Electronic Journal.* https://doi.org/10.2139/ssrn.2668049

Liu, N. & Ye, Z. (2021). Empirical research on the blockchain adoption–based on TAM. *Applied Economics*, *53*(37), 4263–4275. https://doi.org/10.1080/00036846.2021.1898535

Malin Rosenberg. (2018). *Blockchain for the Swedish Fund Market: A comparative study of the blockchain platforms Chain Core and Hyperledger Fabric*. Uppsala Universitet.

Marco Iansiti, & Karim R. Lakhani. (2017). *The Truth About Blockchain*.

Mary C. Lacity. (2018). *A Manager's Guide to Blockchains for Business: From Knowing What to Knowing How*.

Mathiyazhagan, K., Govindan, K., NoorulHaq, A., & Geng, Y. (2013). An ISM approach for the barrier analysis in implementing green supply chain management. *Journal of Cleaner Production, 47*, 283–297. https://doi.org/10.1016/j.jclepro.2012.10.042

Mills, D., Wang, K., Malone, B., Ravi, A., Marquardt, J., Chen, C., Badev, A., Brezinski, T., Fahy, L., Liao, K., Kargenian, V., Ellithorpe, M., Ng, W., & Baird, M. (2016). Distributed Ledger Technology in Payments, Clearing, and Settlement. *Finance and Economics Discussion Series, 95*. https://doi.org/10.17016/FEDS.2016.095

Mishra, L., & Kaushik, V. (2021). Application of blockchain in dealing with sustainability issues and challenges of financial sector. *Journal of Sustainable Finance & Investment*, 1–16. https://doi.org/10.1080/20430795.2021.1940805

Niranjanamurthy, M., Nithya, B. N., & Jagannatha, S. (2019). Analysis of Blockchain technology: pros, cons and SWOT. *Cluster Computing*, 22(6), 14743–14757. https://doi.org/10.1007/s10586-018-2387-5

Pereira, J., Tavalaei, M. M., & Ozalp, H. (2019). Blockchain-based platforms: Decentralized infrastructures and its boundary conditions. *Technological Forecasting and Social Change*, *146*, 94–102. https://doi.org/10.1016/j.techfore.2019.04.030

Prasad, S., Shankar, R., Gupta, R., & Roy, S. (2018). A TISM modeling of critical success factors of blockchain based cloud services. *Journal of Advances in Management Research, 15*(4), 434–456. https://doi.org/10.1108/JAMR-03-2018-0027

Rana, N. P., Dwivedi, Y. K., & Hughes, D. L. (2022). Analysis of challenges for blockchain adoption within the Indian public sector: an interpretive structural modelling approach. *Information Technology & People, 35*(2), 548–576. https://doi.org/10.1108/ITP-07-2020-0460

Reyna, A., Martín, C., Chen, J., Soler, E., & Díaz, M. (2018). On blockchain and its integration with IoT. Challenges and opportunities. *Future Generation Computer Systems, 88*, 173–190. https://doi.org/10.1016/j.future.2018.05.046

Samad, T. A., Sharma, R., Ganguly, K. K., Wamba, S. F., & Jain, G. (2022). Enablers to the adoption of blockchain technology in logistics supply chains: evidence from an emerging economy. *Annals of Operations Research*. https://doi.org/10.1007/s10479-022-04546-1

Sawant. B. S. (2011). Technological Developments in Indian Banking Sector. *Indian Streams Research Journal*.

Shardeo, V., Patil, A., & Madaan, J. (2020). Critical Success Factors for Blockchain Technology Adoption in Freight Transportation Using Fuzzy ANP-Modified TISM Approach. *International Journal of Information Technology and Decision Making*, *19*(6), 1549–1580. https://doi.org/10.1142/S0219622020500376

Sharma, M., Sehrawat, R., Daim, T., & Shaygan, A. (2021). Technology assessment: Enabling Blockchain in hospitality and tourism sectors. *Technological Forecasting and Social Change, 169*, 120810. https://doi.org/10.1016/j.techfore.2021.120810

Shin, D. D. H. (2019). Blockchain: The emerging technology of digital trust. *Telematics and Informatics, 45*, 101278. https://doi.org/10.1016/j.tele.2019.101278

Stewart, H., & Jürjens, J. (2017). Information security management and the human aspect in organizations. *Information & Computer Security, 25*(5), 494–534. https://doi.org/10.1108/ICS-07-2016-0054

Tang, C. S., & Veelenturf, L. P. (2019). The strategic role of logistics in the industry 4.0 era. T*ransportation Research Part E: Logistics and Transportation Review, 129*, 1–11. https://doi.org/10.1016/j.tre.2019.06.004

Tapscott, D., & Tapscott, A. (2017). How Blockchain Will Change Organizations. *MIT Sloan Management Review, 58*(2), 10–13. https://www.proquest.com/scholarly-journals/how-blockchain-will-change-organizations/docview/1875399260/se-2?accountid=27544

Till, B. M., Peters, A. W., Afshar, S., & Meara, J. G. (2017). From blockchain technology to global health equity: can cryptocurrencies finance universal health coverage? *BMJ Global Health, 2*(4), e000570. https://doi.org/10.1136/bmjgh-2017-000570

Toufaily, E., Zalan, T., & Dhaou, S. ben. (2021). A framework of blockchain technology adoption: An investigation of challenges and expected value. *Information & Management, 58*(3), 103444. https://doi.org/10.1016/j.im.2021.103444

Truby, J. (2018). Decarbonizing Bitcoin: Law and policy choices for reducing the energy consumption of Blockchain technologies and digital currencies. *Energy Research & Social Science, 44*, 399–410. https://doi.org/10.1016/j.erss.2018.06.009

Xu, Y., Chong, H.-Y., & Chi, M. (2021). Modelling the blockchain adoption barriers in the AEC industry. *Engineering, Construction and Architectural Management*. https://doi.org/10.1108/ECAM-04-2021-0335

Yadav, A. S., Agrawal, S., & Kushwaha, D. S. (2022). Distributed Ledger Technology-based land transaction system with trusted nodes consensus mechanism. *Journal of King Saud University - Computer and Information Sciences, 34*(8), 6414–6424. https://doi.org/10.1016/j.jksuci.2021.02.002

Yadav, A. S., & Kushwaha, D. S. (2021). Blockchain-based digitization of land record through trust value-based consensus algorithm. *Peer-to-Peer Networking and Applications, 14*(6), 3540–3558. https://doi.org/10.1007/s12083-021-01207-1

Yadav, A. S., & Kushwaha, D. S. (2022). Digitization of Land Record Through Blockchain-based Consensus Algorithm. *IETE Technical Review, 39*(4), 799–816. https://doi.org/10.1080/02564602.2021.1908859

Yadav, A. S., Singh, N., & Kushwaha, D. S. (2022). A scalable trust based consensus mechanism for secure and tamper free property transaction mechanism using DLT. *International Journal of System Assurance Engineering and Management, 13*(2), 735–751. https://doi.org/10.1007/s13198-021-01335-0

Yadav, A. S., Singh, N., & Kushwaha, D. S. (2022). Sidechain: storage land registry data using blockchain improve performance of search records. *Cluster Computing, 25*(2), 1475–1495. https://doi.org/10.1007/s10586-022-03535-0

Zheng, Z., Xie, S., Dai, H. N., Chen, X., & Wang, H. (2018). Blockchain challenges and opportunities: a survey. *International Journal of Web and Grid Services, 14*(4), 352. https://doi.org/10.1504/IJWGS.2018.095647

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 11 N. 4 (2022), 475-488
eISSN: 2255-2863
DOI: https://doi.org/10.14201/adcaij.28558

Performance Evaluation of Efficient Low Power 1-bit Hybrid Full Adder

Rahul Mani Upadhyay[a], R. K. Chauhan[a] and Manish Kumar[a]

[a]Madan Mohan Malaviya University of Technology, Gorakhpur, Uttar Pradesh, India
rahulmaniupadhyay@gmail.com, rkchauhan27@gmail.com, mkece@mmmut.ac.in

KEYWORDS

low power, hybrid full adder, IOT applications, power dissipation, XOR-XNOR circuit

ABSTRACT

The need for a low power system on a chip for embedded systems has increased enormously for human to machine interaction. The primary constraint of such embedded system is to consume less power and improve the battery performance of the device. We propose energy efficient, low power hybrid 1-bit full adder circuit in this paper, which may be integrated on chip to improve the overall performance of embedded systems. The proposed 1-bit hybrid full adder circuit designed at 130 nm technology was simulated using Mentor Graphics EDA tool. Further, a comparison is made with the previously proposed full adders, using metrics such as power dissipation, propagation delay and power delay product. Comparative performance shows that the proposed 1-bit full adder shows average improvement in terms of power dissipation (31.62 nW and 20.84 nW) and average delay (5.07ns and 11.41ns) over the existing 1-bit hybrid and cell 3 full adder circuit.

1. Introduction

The demand for Internet of Things (IoT) technologies in Industry 4.0 has had tremendous growth in recent years [Chen et al. 2018]. One of the main application areas is the connectivity of IoT devices, which may include human-to-machine (H2M) and machine-to-machine (M2M) interaction [Giordani et al. 2019]. Thus, increasing demand for technologies with real time applications in Industry 4.0, i.e., which includes the artificial intelligence (AI) and machine learning algorithm, requires an embedded system which is highly reliable and fast [Giordani et al. 2019]. However, a major challenge to such technology being developed is the requirement of a low power and high speed circuit that has to be integrated with a system-on-chip (SOC) device [Giordani et al. 2019]. Hence, demand for the high processing unit is increasing day by day, that includes the basic arithmetic and logic unit operation, which fully depends on this battery-operated device.

There has been an exponential growth in the use of battery operated portable systems over the past few decades, which demands an energy efficient circuitry. To enhance the performance and integration of more functions into chip, the feature size of each transistor has to shrink more, which consequently increases the power density [Nadu et al. 2014]. Therefore, power reduction has become the most critical factor for the success of the microelectronics industry. Power dissipation in digital circuits is minimized either by reducing the power supply voltage or decreasing the operating frequency. However, it results in the rise in propagation delay and also degrades the driving capability of the designed circuits. In electronic systems, digital signal processing (DSP) units play a significant role. DSP based electronic processors perform operations such as filtering, discrete Fourier transform, convolution, Fast Fourier transform and video processing. They perform mathematical operations such as addition, subtraction, division and multiplication [Wairya et al. 2011]. Adder circuit is the core module in these systems to perform various mathematical computations. Hence, increasing the performance of an adder circuit greatly improves the performance of these systems.

Literature in the last few years demonstrates various circuit techniques used for designing adders. Authors have designed various adder circuits with different logics and different transistor sizes, resulting in differences in propagation delay and power dissipation. A conventional CMOS adder (C-CMOS) consists of pull-down and pull-up networks and is designed by using 28 transistors. This adder circuit is reliable and simple; however, due to the greater number of transistors, issues such as greater power dissipation, slower switching speed and larger area are observed. Hybrid full adder has several sub-modules. The primary advantage of using hybrid full adder is that each sub-module can be checked and can be replaced with another sub-module that has a better performance. Further, these sub-modules are connected together to obtain the complete full adder. [Wairya et al. 2011]–[Goel et al. 2006] proposed a hybrid full adder that provides full swing output and is energy efficient. However, the proposed hybrid adder circuit fails to perform at lower voltages. Among different reported structures of full adder, three structures for implementing a full adder are shown in Figure 1 and their logic equations are mentioned in Table 1. The outputs from these structures are H and which are equal to $A \oplus B$ and respectively. Figure 1(a) represents the block diagram for implementing a full adder that is energy efficient and also requires lesser area [Nadu et al. 2014]. However, this design has higher carry propagation delay when used in cascaded structure due to sharing of carry input in the two modules. [Wairya et al. 2011] presented an alternate design of adder circuit which is shown in Figure 1(b). The authors had proposed two different full swing output adder circuits by using swing-restored complementary pass-transistor logic and double pass-transistor logic. However, these designs have higher power dissipation and higher transistor count. Figure 1(c) shows a hybrid design of the adder circuit as proposed by [Tung et al. 2007]. This adder circuit is energy efficient, reliable and provides full-swing output.

In the proposed work, a 1-bit energy efficient hybrid full adder is designed. The circuit utilizes 14 MOS transistors. The proposed full adder circuit shows lesser power dissipation and lesser power-delay product as compared to some of the previous related works. The effect of temperature on power dissipation has also been assessed in the proposed adder circuit by varying the temperature.

Therefore, a low power application has been identified as a prime concern for a Very Large Scale Integration (VLSI) system, which consumes less power and provides better battery life [Wairya et al. 2011]–[Goel et al. 2006]. Full adder is the fundamental building block in a portable device, which can be addressed through different logic designs to improve the battery life and decrease the silicon area i.e., by reducing the number of transistors [Goel et al. 2006]–[Hassoune et al. 2010]. Nevertheless, different logic designs (i.e., static and dynamic approach), performance metrics (i.e., power dissipation and delay) and number of transistors used in the designs are some of the fundamental parameters that have been studied by the researcher at different voltage ranges and technology scale (nano-technology) [Bhattacharyya et al. 2015], [Tirumalasetty et al. 2019], [Abid et al. 2008].

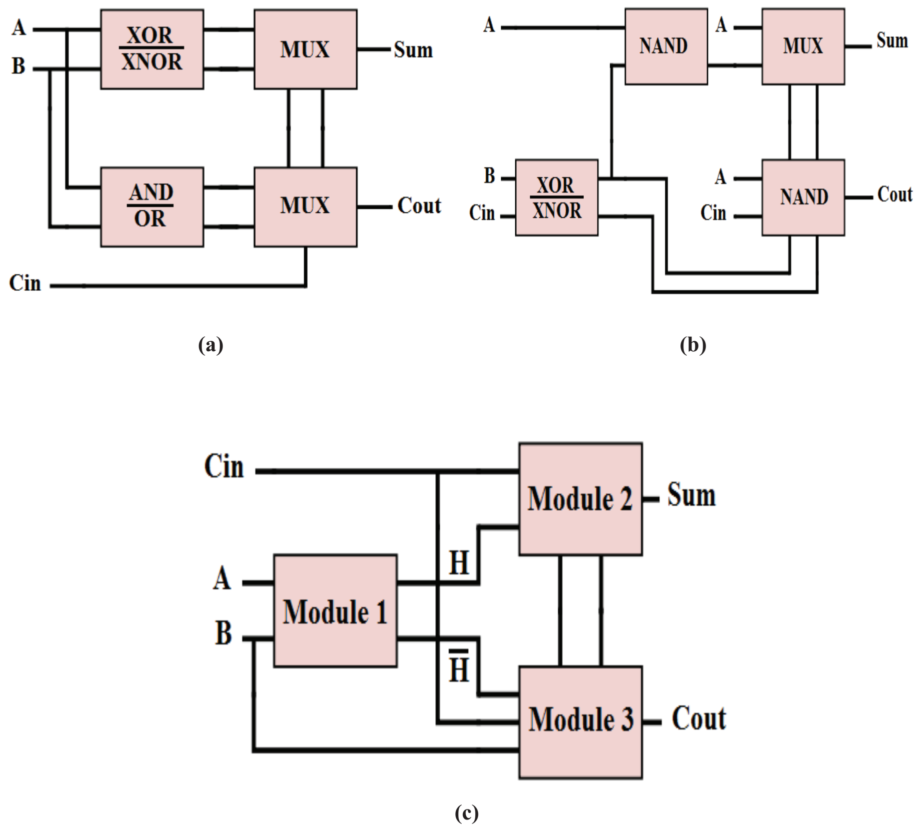


Figure 1. Different structures for implementing a full adder circuit: (a) Aguirre's model (b) Kumar's model (c) Hybrid model

Table 1. Logic equations for 1-bit full adder

Structure	Sum	Cout
Figure 1(a)	$H\bar{C}_{in} + \bar{H}C_{in} = (A \oplus B) \oplus C_{in}$	$\bar{C}_{in}(AB) + C_{in}(A+B)$
Figure 1(b)	$(B \oplus C_{in})\overline{(A(B \oplus C_{in}))} + A\overline{(B \oplus C_{in})}$	$A(B \oplus C_{in}) + C_{in}\overline{(B \oplus C_{in})}$
Figure 1(c)	$H\bar{C}_{in} + \bar{H}C_{in} = (A \oplus B) \oplus C_{in}$	$HCin + \bar{H}A$

On the basis of the study, mainly different logic styles are used to design the full adder such as; static and dynamic CMOS [Bhattacharyya et al. 2015], [Goel et al. 2006], [Amini-Valashan et al. 2018], [Sun et al. 2014], ratioed style [Wairya et al. 2012], complementary pass-transistor logic (CPL) [Bhattacharyya et al. 2015], [Hasan et al. 2019], [Aguirre-Hernandez et al. 2011], [Nadu et al. 2014], Transmisión Gate (TG)-CMOS [Goel et al. 2003], [Janwadkar et al. 2018], [Hassoune et al. 2010], [Aguirre-Hernandez et al. 2011], Static Energy Recovery Full (SERF) [Wairya et al. 2011], [Goel et al. 2006], [Venkatesan et al. 2019], [Yadav et al. 2018], carbon nanotube Field Effect Transistor (FET) [Dokania et al. 2018], Gate Diffusion Input (GDI) [Hasan et al. 2019], [Venkatesan et al. 2019], [Badry et al. 2018] and their hybrid arrangement [Wairya et al. 2011], [Bhattacharyya et al. 2015], [Kandpal et al. 2019], [Hasan et al. 2019], [Tirumalasetty et al. 2019], [Naseri et al. 2018], [Aguirre-Hernandez et al. 2011], [Nadu et al. 2014], [Yadav et al. 2018], [Dwivedi et al. 2016].

Every design has got their merits and demerits, for example, a static full adder is more reliable than dynamic full adder and consumes less power, whereas the dynamic full adder has faster switching speed in comparison to static full adder. Also, for N input logic, dynamic adder requires only (N+2) number of transistors whereas the static one requires 2N transistors. On the other hand, the voltage swing restoration of CPL is better in comparison to CMOS which contributes to low power consumption due to low voltage swing at the internal node. However, it suffers from static power consumption [Bhattacharyya et al. 2015], [Goel et al. 2006], [Hassoune et al. 2010], [Aguirre-Hernandez et al. 2011], whereas, voltage scaling [Dokania et al. 2018] of CMOS is better in comparison to CPL [Wairya et al. 2012], [Bhattacharyya et al. 2015], [Naseri et al. 2018]. Thus, the different logic styles have their own advantages and disadvantages, which is mainly dependent on power consumption, chip area and reliability constraint [Dokania et al. 2018], [Bhattacharyya et al. 2015], [Kandpal et al. 2019], [Naseri et al. 2018], [Amini-Valashan et al. 2018], [Abid et al. 2008], [Dwivedi et al. 2016]. Hence, finding the optimum solution to designing a full adder is considered an open research challenge for the researcher.

The notable work reported for 1-bit hybrid full adder circuit using GDI and CMOS technique at 120 nm technology simulated using Microwind VLSI CAD tool was described in [Nadu et al. 2014]. It was shown that, at supply voltage of 1.2 V, the proposed circuit provides better performance in terms of power dissipation and delay of 9 µW and 18 ps when compared with the conventional CMOS. Further, the silicon area size can be reduced due to the lesser transistor count (21T) when compared with the number of transistors (28T) in conventional CMOS [Nadu et al. 2014].

P. Bhattacharyya *et al.,* reported the work based on hybrid arrangement of CMOS logic and TGL (transmission gate logic) using 16 transistors (16T) for low power and high speed application [Bhattacharyya et al. 2015]. It was shown in [Bhattacharyya et al. 2015], the hybrid circuit has better delay response when compared with the other existing models. The simulation was carried out for both 180 nm and 90 nm technology. The best performance in terms of power dissipation and delay was found to be 112.79 µW and 5.57 ns at 1.8V supply voltage for 180 nm technology and 53.36 µW and 2.45 ns at 1.2V supply voltage respectively. This was achieved due to the efficient pairing of strong transmission gates with the CMOS inverters [Bhattacharyya et al. 2015].

On the other hand, A. K. Yadav *et al.,* presented 1-bit FA for low power application using 10T configuration designed at 45 nm CMOS technology using the HSIPCE software [Yadav et al. 2018]. It was reported that, the circuit presented in [Yadav et al. 2018], provides the maximum amount of improvement in PDP of 96.01% and 91.64% for GDI and SERF respectively. The hybrid combination of 1-bit adder using CPL, CMOS, and TG logic using 13T configuration for 90 nm technology is reported in [Kadu et al. 2018]. It was shown that considerable decrease in delay can be obtained by decreasing the transistor count i.e., delay of 5.18 ns and 20.03 ns is obtained using 13T and 16T full adder configuration.

Naseri and Timarchi presented the study of different type of logic for the implemented XOR/XNOR gates for 65-nm technology using CMOS [Naseri et al. 2018]. Simulation were carried out using HSPICE and Cadence software and showed that the best results were obtained using hybrid

FA-22T cell configuration, saving 43.5% energy delay product (EDP) and 23. 4% PDP when simulated over 0.65 to 1.5 V voltage range. Further, the study also presented the device sizing, which was obtained with the help of particle swarm optimization (PSO) algorithm [Naseri et al. 2018].

From the above discussion, it can be inferred that the 1-bit adder circuits design using the hybrid combination can provide an overall improvement in system performance. As different logic has different merits and demerits, therefore, in this we propose the new novel hybrid 1-bit adder using 14 T configurations, simulated at 130 nm technology using Mentor Graphics tool at 27 °C. The primary aim of this research work is to design the full adder circuit, which can provide the optimum solution in term of propagation delay and power dissipation. In Section 2 the schematic of the proposed hybrid 1-bit FA circuit and its operation is presented, as well as its comparison with other existing full adder. Section 3 deals with the results and analysis of the proposed 1-bit hybrid full adder and finally the conclusion is presented in Section 4.

2. Design and Simulation Setup of the Proposed 1-Bit Hybrid Full Adder

The schematic of the proposed 1-bit hybrid full adder circuit is shown in Figure 1, which is designed using fourteen (14) transistors. As observed from Figure 1, the hybrid arrangement of the proposed 1-bit hybrid full adder consists of three modules i.e., Module 1, Module 2 and Module 3 respectively (shown with color box). As shown in Figure 2, Module 1 arrangement has circuit of XOR-XNOR [Dokania et al. 2018], [Goel et al. 2006]–[Janwadkar et al. 2018], [Naseri et al. 2018], designed using six transistors based on the inverter configuration i.e., T_{n1}, T_{p1}, T_{n2}, T_{p2}, T_{n3}, T_{p3} respectively. The advantage of using the XOR-XNOR circuit is that it reduces the complexity of the circuit as only half the number of transistor is required as compared with the conventional CMOS XOR circuit [Wairya et al. 2011], [Tung et al. 2007], [Amini-Valashan et al. 2018]. Thus, it results in reducing the number of internal node, which further provides the reduction in power dissipation.

The output generated from Module 1 (using two inputs A and B) is coupled with Module 2 and Module 3. Input C is connected to Module 2 and Module 3 respectively. Module 2 consists of a pair of transistors (T_{n4}, T_{p4}, T_{n5}, T_{p5}) responsible for sum (SUM) output signal, whereas Module 3 consists of transmission gate logic (T_{n6}, T_{p6}, T_{n7}, T_{p7}) responsible for carry output (C_{out}) signal. Carry propagation delay is reduced by TGL which further provides reduction in the overall propagation delay.

As discussed earlier, module-1 is a XOR-XNOR circuit which further drives module-2 and module-3. Elaborated discussions on all these modules are presented below:

Module-1 in Figure 2 is XOR-XNOR logic cell which generates output signals and acts as a driving input source to successive modules. Module 1 is responsible for maximum power dissipation. Since module 1, which consists of six transistors (6T), will provide full swing, so for swing restoration no extra transistors are required. Due to no direct path between power supply and ground, short- circuit current has been reduced which provides good driving capability, thus providing the robustness to the circuit. The major advantage of this module is that for all input combinations, both outputs have perfect voltage level.

Module-2 of the proposed full adder is PTL and Transmission gate based XOR gate. The function of this module is to implement SUM output. The outputs of module-1 serve as the inputs for module-2 while another input signal, (Cin) is provided externally. There are in total 4 transistors required for implementing the SUM output. Both XOR and its compliment are generated from module-1. Therefore, additional transistors are not required for to design the inverter that would complement the input XOR in module-2. The major advantage of this module is that it provides full swing output for all inputs and has lesser transistor count thus, resulting in low power dissipation. **Module-3** is based on TG logic and is built by connecting nMOS and pMOS transistors in parallel and controlled by the complementary control signals. When turned on simultaneously, both transistors (nMOS and pMOS) provide the input logic "0" or "1" respectively. Hence no voltage drop problem whether the 0 or the 1 is passed through it.

Thus, the proposed one bit full adder is used for minimizing the power dissipation. Also, due to the lesser number of transistors (14 T configurations), the power dissipation at the internal node reduced which decreased the power dissipation. Also, the carry propagation delay was reduced due to the TG logic, which further improves the overall propagation delay.

The proposed 1-bit hybrid circuit of full adder designed on the Mentor graphic tool is shown in Figure 3 and the time domain waveform, obtained from proposed 1-bit hybrid circuit of full adder at V_{DD} =3V, is shown in Figure 4, this verifies the operation of the full adder circuits.

As a result, new complete adder architectures with low power dissipation and high speed are required. The 14T topology shown in Figure 2 is a full adder circuit. The adder chains integrated in the multiplier performed significantly better thanks to the compact circuit design and lower number of transistor counts. The low-power dissipation of adder topologies effectively minimizes the multiplier unit's overall power consumption and ensures energy-efficient low power computing.

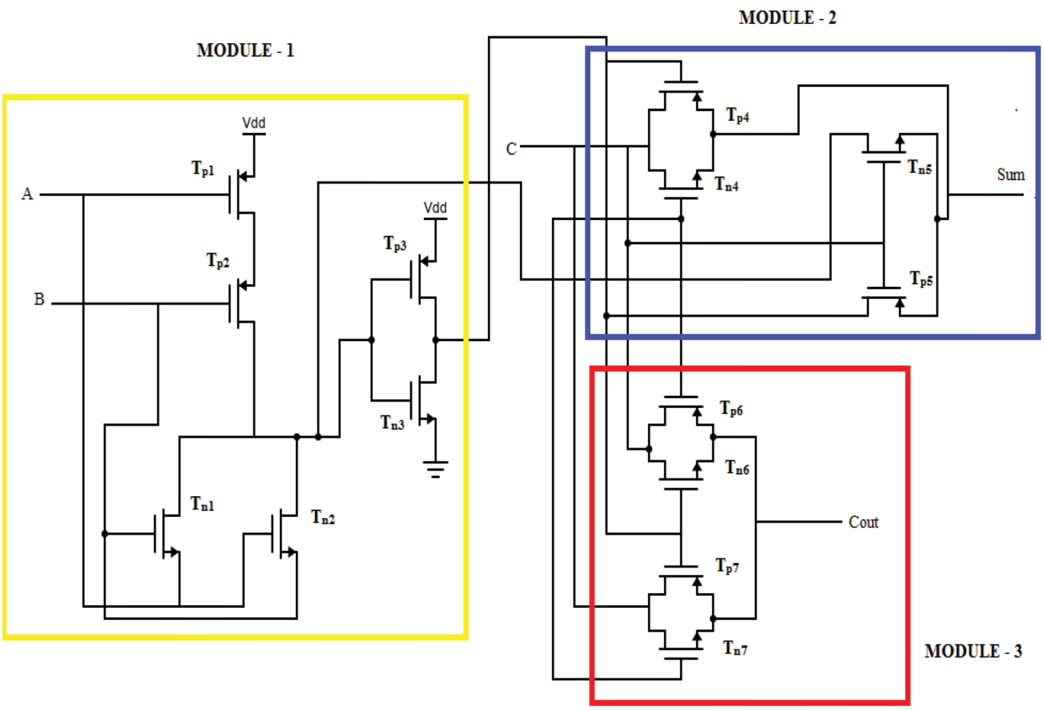


Figure 2. Design structure of the proposed 1-bit hybrid full adder for low power application (Different colors are used to distinguish between different modules).

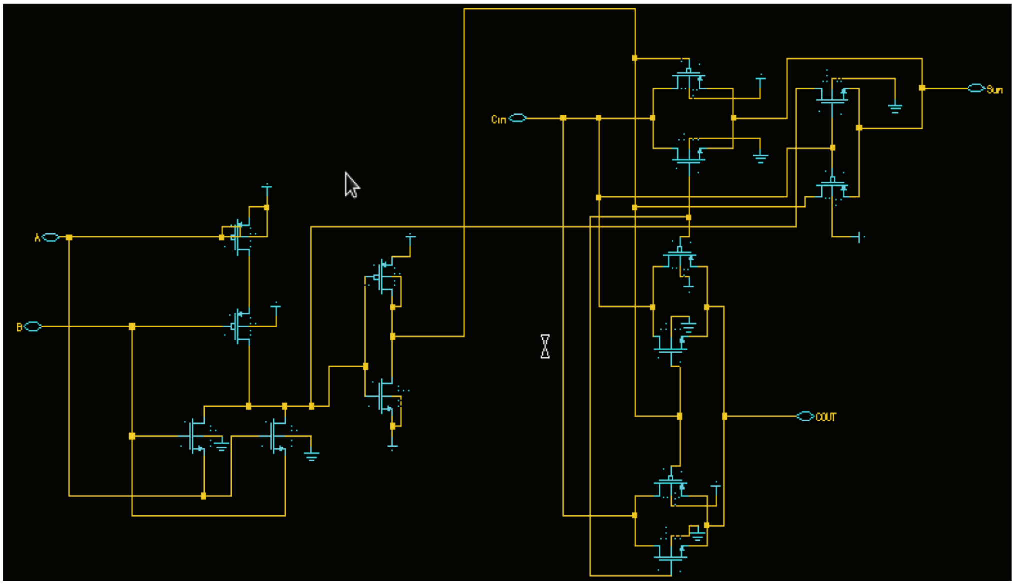


Figure 3. Simulated design of the proposed 1-bit hybrid full adder on Mentor graphic tool at 130 nm technologies

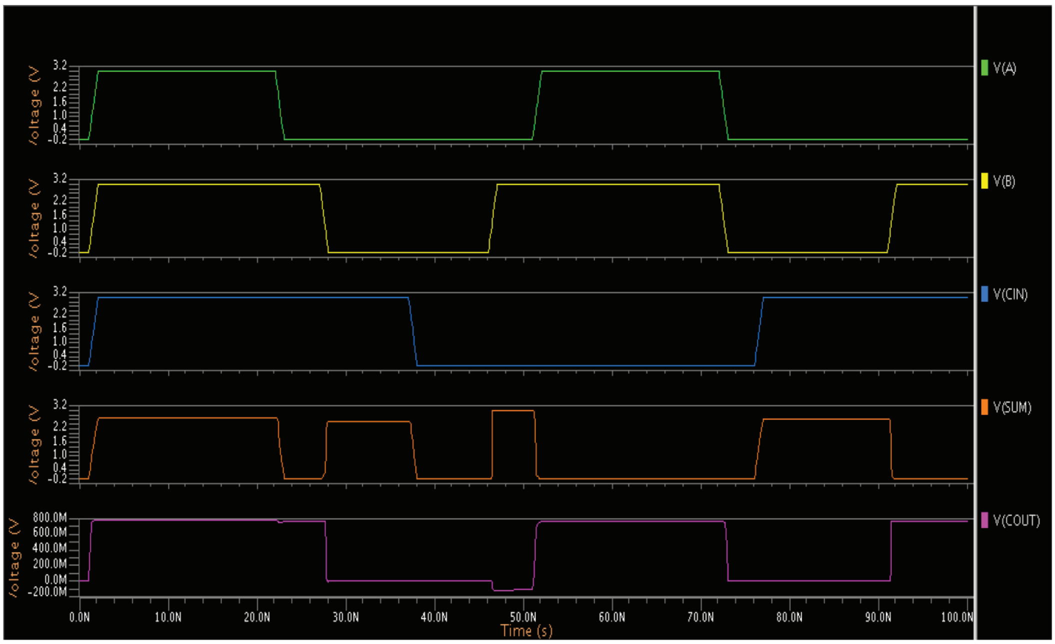
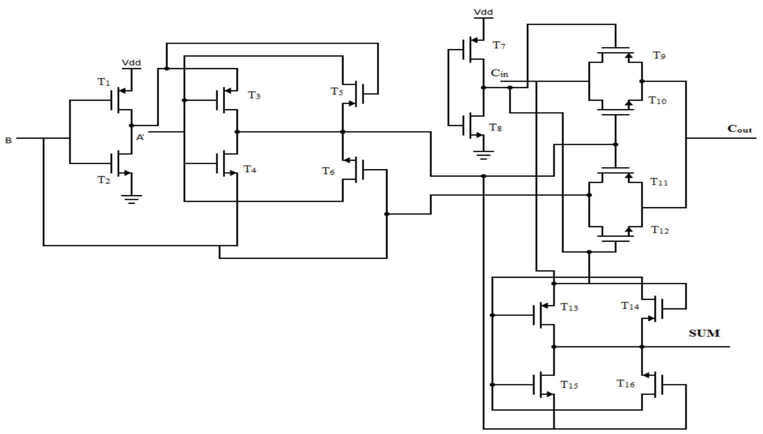
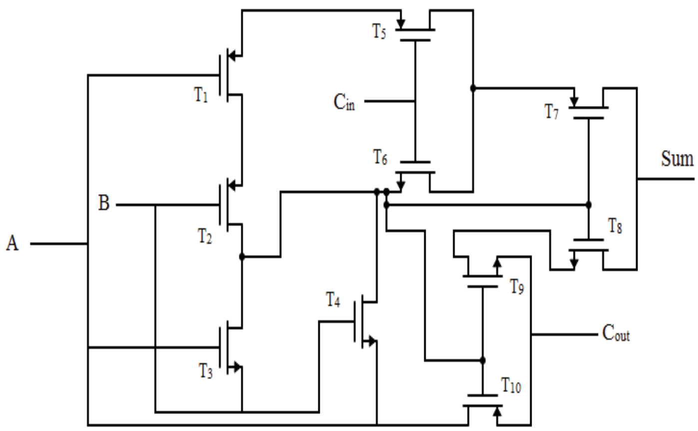


Figure 4. Time domain simulation waveform of the proposed 1 bit hybrid full adder at V_{DD} =3V.

Further, to make a comparison of performance, the proposed circuit of 1-bit hybrid full adder circuit was assessed along with the previous others 1-bit FA circuits reported in [Wairya et al. 2012], [Dokania et al. 2018] (shown in Figure 5 (a) and Figure 5 (b) respectively) are simulated at 130 nm technology design using Mentor Graphics Tools at 100 MHz frequency.



(a)



(b)

Figure 5. The schematic circuit design of 1-bit full adder
(a) Using Hybrid 1-bit full adder, (b) Cell 3 full adder

The hybrid full adder (16T) circuit depicted in Figure 5(a) was created using transmission gates and a low power XOR-XNOR architecture. It has an internal node threshold voltage loss issue that has spread to the output nodes. Figure 5(b) shows Cell 3 Full Adder. The designs have unfavorable output voltage levels and poor operation. Therefore, it may be said that Cell 3 Full Adder and Hybrid Full Adder either have unsatisfactory output levels or unfavorable output voltage level. The design structure of the proposed full adder in Figure 2 exhibits full swing output level.

The performance of digital circuits and systems is measured through parameters such as power dissipation, delay and power delay product. The power dissipated in digital circuits is classified into two categories: Static and dynamic power. The total power dissipated in any digital circuit is called average power dissipation (P_{avg}). Static power dissipation occurs due to leakage current in the circuit when it is in idle or standby mode. Theoretically, in standby mode a CMOS digital circuit does not dissipate power because either pull-up network or pull-down network is in OFF state, which prevents leakage current. In a practical situation, a certain amount of power passes through the transistors. Static power dissipation associated with each transistor due to leakage current is quite small; however, when millions of transistors are integrated together, their cumulative effects become significant. The other type of power dissipation is dynamic power, which occurs while the device is in active state and is further classified into short-circuit power and switching power dissipation. The power dissipated in the charging and discharging of load capacitances is termed as switching power. The portion of power dissipation due to the flow of current from supply voltage (V_{dd}) directly to the ground terminal is called short-circuit power.

To evaluate the performance, the comparison is carried out using three metric viz. power dissipation, propagation delay and PDP. Propagation delay ($T_{Propagation}$) depends on the chip area of the transistors (due to the scaling of transistors) and power dissipation and power delay product (PDP) in the circuit is given as:

$$P_{avg} = P_{static} + P_{dynamic} = I_{leakage} \times V_{dd} + f\, C_L \times V_{dd}^{\,2} + I_{SC} \times V_{dd} \tag{1}$$

where $I_{leakage}$, V_{dd}, f, C_L, I_{SC} represents leakage current, supply voltage, switching frequency, output capacitance, short circuit current:

$$PDP = T_{Propagation} \times P_{avg} \tag{2}$$

The propagation delay is the time needed for an input signal to reach to the output. It can be calculated by taking the time difference of the 50% transition points of the output and input waveforms.

The Power Delay Product (PDP) in any logic circuit is defined as the product of the power dissipation of the circuit and the propagation delay. Reduction in power dissipation and delay greatly improves the efficiency of digital circuits. An energy efficient digital circuit has lesser PDP.

3. Results and Discussion

The comparative performance parameters using the proposed 1-bit hybrid full adder, along with the existing other 1-bit full adder presented in [Wairya et al. 2012], [Dokania et al. 2018] for the voltage range of 1 V to 3 V (volts) at 130-nm technology using Mentor Graphics is shown in Table 2.

Table 2. Performance Parameters Comparison of Full Adder (1.0 V to 3.0 V)

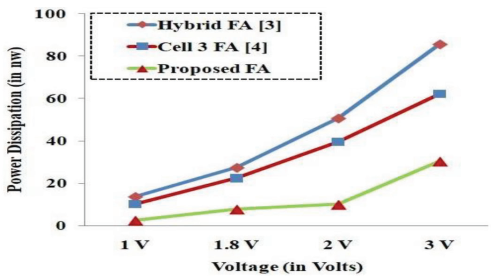
V_{dd} (V)	1.0	1.8	2.0	3.0
Power Dissipation (nW)				
Hybrid FA	12.80	27.60	52.58	84.10
Cell 3 FA	10.20	22.08	39.38	62.30
Proposed FA	1.10	7.18	9.20	33.12
Delay (nS)				
Hybrid FA	19.80	19.90	17.48	16.10
Cell 3 FA	28.20	26.17	24.32	19.95
Proposed FA	15.90	15.20	12.80	9.10
Power Delay Product (x 10^{-18}J)				
Hybrid FA	253.44	594.24	919.09	1354.01
Cell 3 FA	287.64	577.83	957.72	1242.88
Proposed FA	17.49	109.136	117.76	301.39

As shown in Figure 6 (a), the hybrid FA circuit presented in [Wairya et al. 2012] does not show any improvement in power consumption (shown in nW) when compared with Cell 3 FA [Dokania et al. 2018], whereas the proposed hybrid circuit outperforms both of the existing 1-bit full adder circuits. The average improvement in power dissipation of 31.62 nW and 20.84 nW is obtained using proposed 1-bit hybrid full adder when compared with the Hybrid FA [Wairya et al. 2012] and Cell 3 FA [Dokania et al. 2018] respectively. Similarly, the simulated results obtained in Figure 6(b), for delay (shown in ns) shows, that the cell 3 FA shows poor performance when compared with existing hybrid FA [Wairya et al. 2012]. However, our proposed 1-bit hybrid full adder shows better performance when compared with previously existing full adder. It provides an average improvement in propagation delay of 5.07 ns and 11.41 ns when compared with the Hybrid FA [Wairya et al. 2012] and Cell 3 FA [Dokania et al. 2018] respectively. It can be inferred from Figure 6(c), that the total PDP of both (previous existing adder) 1-bit full adder was almost similar, due to the fact that, one provides better power consumption while failing to provide any improvement in the propagation delay and vice versa.

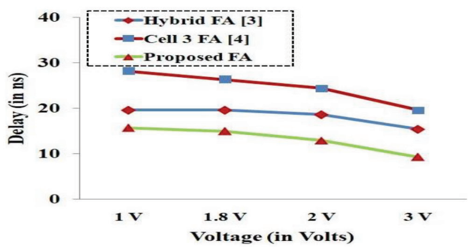
Thus, our proposed 1-bit hybrid full adder shows improvement in both power dissipation and propagation delay, therefore, improving the PDP of the proposed circuit as shown in Figure 6(c).

3.1 Temperature Analysis

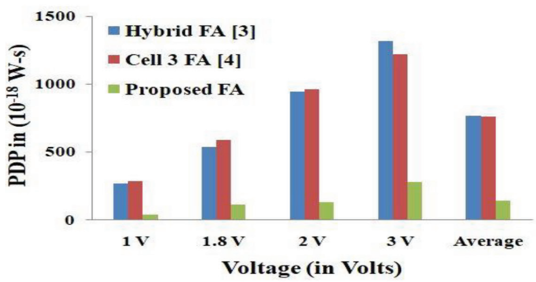
In today's era, where a large number of integrated circuits are fabricated on small area size die; a small temperature variation can significantly affect the overall performance. With the rise in temperature, the leakage current in any logic circuit also increases. To analyze the effect of temperature on the power dissipation of the proposed circuit, temperature ranges from -30^{0}C to 70^{0}C have been simulated. The power dissipation increases almost linearly with increase in temperature at lower temperature ranges and increases exponentially with further increase in temperature. Figure 7 shows the temperature variation effect on the power dissipation of the proposed adder circuit. The power dissipation increases at higher temperature ranges due to the increase in leakage current.



(a)



(b)



(c)

Figure 6. Comparative performance of the simulation results using proposed 1-bit hybrid adder circuit when compared with existing 1-bit adder in terms of, (a) Power dissipation (in nW), (b) Delay (in nS), (c) Power delay product (PDP in 10-18 W-s)

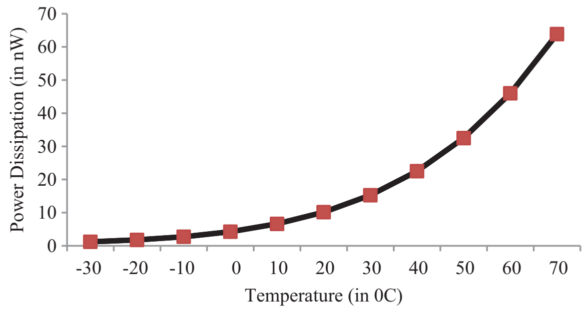


Figure 7. The effect of temperature variation on the power dissipation of the proposed circuit

4. Conclusion

The main objective of this research work was to design an efficient, low power, high speed adder circuit that can be integrated on a chip, making it a suitable and efficient choice for IOT applications. Energy efficient, low power 1-bit hybrid adder circuit is proposed in this paper, which provides better power dissipation and reduction in propagation delay, making it ideally suitable for low power application. The proposed 1-bit hybrid full adder provides the average power dissipation of 12.65 nW and average delay of 13.25 ns for voltage range between 1V to 3V volt. As the proposed circuit is designed using 14 T configuration, the area on the chip is also reduced. The proposed hybrid full adder circuit can be used in other different circuits such as multiplier, compressor and biomedical devices. It can detect faults in real time and can be used to design low power digital processing systems.

5. References

Chen S. L., C. H. Liao, T. K. Chi, T. L. Lin, And C. A. Chen, 2018 “Flexible Signals and Images Lossless Compression Chip Design for IoT and Industry 4.0,” in Proceedings of 14th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications, pp. 1–6. DOI: 10.1109/MESA.2018.8449205

Giordani, M., M. Polese, M. Mezzavilla, S. Rangan, And M. Zorzi, 2019 “Towards 6G Networks: Use Cases and Technologies,” IEEE Communications Magazine, vol. 58, no. 3, pp. 55–61, Mar. DOI: 10.1109/MCOM.001.1900411

Wairya, S., R. K. Nagaria, And S. Tiwari, 2012 “Comparative Performance Analysis of XOR- XNOR Function Based High-Speed CMOS Full Adder Circuits For Low Voltage VLSI Design,” International Journal of VLSI design & Communication Systems (VLSICS), vol. 3, no. 2, pp. 432–436, DOI: 10.5121/vlsic.2012.3219

Dokania, V., R. Verma, M. Guduri, And A. Islam, 2018. “Design of 10T full adder cell for ultralow-power applications,” Ain Shams Engineering Journal, vol. 9, no. 4, pp. 2363–2372, DOI: 10.1016/j.asej.2017.05.004

Wairya, S., G. Singh, Vishant, R. K. Nagaria, And S. Tiwari, 2011 “Design analysis of XOR (4T) based low voltage CMOS full adder circuit,” in Proceedings of Nirma University International Conference on Engineering, pp. 1–7. DOI: 10.1109/NUiConE.2011.6153275.

Bhattacharyya, P., B. Kundu, S. Ghosh, V. Kumar, And A. Dandapat, 2015. “Performance Analysis of a Low-Power High-Speed Hybrid 1-bit Full Adder Circuit,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 23, no. 10, pp. 2001–2008, DOI: 10.1109/TVLSI.2014.2357057

Goel, S., A. Kumar, And M. A. Bayoumi, 2006. “Design of robust, energy-efficient full adders for deep-submicrometer design using hybrid-CMOS logic style,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 14, no. 12, pp. 1309–1321, DOI: 10.1109/TVLSI.2006.887807

Goel, S., M. A. Elgamel, And M. A. Bayoumi, 2003 “Novel design methodology for high-performance XOR-XNOR circuit design,” in Proceedings of 16th Symposium on Integrated Circuits and Systems Design, pp. 71–76. DOI: 10.1109/SBCCI.2003.1232809

Janwadkar, S. And S. Das, 2018, “Design and Performance Evaluation of Hybrid Full Adder for Extensive PDP Reduction,” in Proceedings of 3rd International Conference for Convergence in Technology, pp. 1–6. DOI: 10.1109/I2CT.2018.8529780

Kandpal, J., A. Tomar, S. Adhikari, and V. Joshi, 2019, "Design of Low Power and High Speed XOR/XNOR Circuit using 90 nm CMOS Technology," in Proceedings of 2nd International Conference on Innovations in Electronics, Signal Processing and Communication, pp. 221–225. DOI: 10.1109/IESPC.2019.8902392

Hasan, M., U. K. Saha, A. Sorwar, M. A. Z. Dipto, M. S. Hossain, And H. U. Zaman, 2019, "A Novel Hybrid Full Adder Based on Gate Diffusion Input Technique, Transmission Gate and Static CMOS Logic," in Proceedings of 10th International Conference on Computing, Communication and Networking Technologies, pp. 1–6. DOI: 10.1109/ICCCNT45670.2019.8944888

Hassoune, I., D. Flandre, I. O'connor, And J. D. Legat, 2010. "ULPFA: A new efficient design of a power-aware full adder," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 57, no. 8, pp. 2066–2074, DOI: 10.1109/TCSI.2008.2001367

Tirumalasetty, V. R., And M. R. Machupalli, "Design and analysis of low power high-speed 1-bit full adder cells for VLSI applications," International Journal of Electronics, vol. 106, no. 4, pp. 521–536, 2019. DOI: 10.1080/00207217.2018.1545256

Naseri, H.,And S. Timarchi, 2018. "Low-Power and Fast Full Adder by Exploring New XOR and XNOR Gates," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 26, no. 8, pp. 1481–1493, DOI: 10.1109/TVLSI.2018.2820999

Tung, C. K., Y. C. Hung, S. H. Shieh, And G. S. Huang, 2007, "A low-power high-speed hybrid CMOS full adder for embedded system," in Proceedings of the 2007 IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems, DDECS, pp. 199–202. DOI: 10.1109/DDECS.2007.4295280

Amini-Valashani, M., M. Ayat, And S. Mirzakuchaki, 2017, "Design and analysis of a novel low-power and energy-efficient 18T hybrid full adder," Microelectronics Journal, vol. 74, pp. 49–59, 2018. DOI: 10.1016/j.mejo.2018.01.018

Abid, Z., H. El-Razouk, And D. A. El-Dib, 2008. "Low power multipliers based on new hybrid full adders," Microelectronics Journal, vol. 39, no. 12, pp. 1509–1515, DOI: 10.1016/j.mejo.2008.04.002

Sun, Y., And V. Kursun, 2014. "Carbon nanotubes blowing new life into NP dynamic CMOS circuits," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 61, no. 2, pp. 420–428, DOI: 10.1109/TCSI.2013.2268131

Aguirre-Hernandez, M. And M. Linares-Aranda, 2011 "CMOS Full-Adders for Energy-Efficient Arithmetic Applications," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 19, no. 4, pp. 718–721. DOI: 10.1109/TVLSI.2009.2038166

Nadu, T., S. Mohan, and N. Rangaswamy, 2014, "Performance Analysis of 1 bit Full Adder Using GDI Logic," in Proceedings of International Conference on Information Communication and Embedded Systems (ICICES2014), no. 978, pp. 1–4. DOI: 10.1109/ICICES.2014.7034029

Venkatesan, C., M. Thabsera Sulthana, M. G. Sumithra, And M. Suriya, 2019, "Analysis of 1- bit full adder using different techniques in Cadence 45nm Technology," in Proceedings of 5th International Conference on Advanced Computing and Communication Systems, pp. 179–184. DOI: 10.1109/ICACCS.2019.8728449

Yadav, A. K., B. P. Shrivatava, And A. K. Dadoriya, 2018, "Low power high speed 1-bit full adder circuit design at 45nm CMOS technology," in Proceedings of International Conference on Recent Innovations in Signal Processing and Embedded Systems, pp. 427–432. DOI: 10.1109/RISE.2017.8378203

Badry, O. Al And M. A. Abdelghany, 2018, "Low power 1-Bit full adder using Full-Swing gate diffusion input technique," in Proceedings of 2018 International Conference on Innovative Trends in Computer Engineering, vol. 2018-March, no. ITCE, pp. 205–208. DOI: 10.1109/ITCE.2018.8316625

Dwivedi, S., K. Khare, And A. K. Dadoria, 2016, "Low-Power High Speed 1-bit Full Adder Circuit Design," in Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies, pp. 1–5. DOI: 10.1145/2905055.2905321

Kadu C. P., And M. Sharma, 2018, "Area-Improved High-Speed Hybrid 1-bit Full Adder Circuit Using 3T-XNOR Gate," in Proceedings of International Conference on Computing, Communication, Control and Automation, pp. 1–5. DOI: 10.1109/ICCUBEA.2017.8463827.

Notes of Contributors

Rahul Mani Upadhyay received the B.Tech. degree in Electronics and Communication Engineering, the M. Tech. degree in Electronic Design and Technology from Uttar Pradesh Technical University, Lucknow, India in 2010 and 2013 respectively and currently pursuing Ph.D. degree in VLSI and Microelectronics from Madan Mohan Malaviya University of Technology, Gorakhpur, India. His current research interests include digital systems, VLSI design, and low-power and ultra low power digital circuits.

Dr. Rajeev Kumar Chauhan received the Ph.D degree in Electronics Engineering from IIT-BHU, Varanasi in 2002 and ME from MNNIT – Allahabad in 1993 and B.Tech from G.B.P.U.A.T –Pantnagar in 1989. He has joined erstwhile Madan Mohan Malaviya Engineering College, Gorakhpur in 1993 as a Lecturer, as an Assistant Professor from 2002, as an Associate Professor from 2006 and Professor since 2017. The college has been reconstituted as Madan Mohan Malaviya University of Technology by U.P. State Government in 2013. He also worked as Professor in Department of ECE, Faculty of Technology, Addis Ababa University, Ethiopia between 2003 to 2005. He is author of more than 200 research papers published in Journals and Conferences. His current research interests include emerging nanoscale devices and issues in in VLSI as well as circuits and systems.

Dr. Manish Kumar received Ph.D. degree in VLSI and Microelectronics from IIT (ISM) Dhanbad, India in 2014. He is Associate Professor in the Department of Electronics and Communication Engineering, Madan Mohan Malaviya University of Technology, Gorakhpur, India. His field of Specialization is VLSI design. He has received INSA-Visiting Fellowship by the Indian National Science Academy, New Delhi in the year 2012-13 and Selected as a UGC Dr. D.S. Kothari Post Doctoral Fellow (DSK-PDF) in the year 2014-15. He is a member of Indian Science Congress Association, International Association of Engineers, Institution of Engineers (India), Indian Society for Technical Education, Semiconductor Society of India and Institution of Electronics and Telecommunication Engineers.

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 11 N. 4 (2022), 489-505
eISSN: 2255-2863
DOI: https://doi.org/10.14201/adcaij.30128

Contextual Urdu Text Emotion Detection Corpus and Experiments using Deep Learning Approaches

Muhammad Hamayon Khan Vardag[a], Ali Saeed[b],
Umer Hayat[a], Muhammad Farhat Ullah[a], Naveed Hussain[b]

[a]Department of Software Engineering, University of Lahore, Lahore, Pakistan
[b]Department of Software Engineering, Faculty of Information Technology, University of Central Punjab, Lahore, Pakistan
humayonverdag98@gmail.com, ali.saeed@ucp.edu.pk, hayatumer575@gmail.com, farhat.ullah@se.uol.edu.pk, and dr.naveedhussain@ucp.edu.pk

KEYWORDS

emotion detection corpus; labeled corpus; deep learning approaches; supervised learning approaches

ABSTRACT

Textual emotion detection aims to discover human emotions from written text. Textual emotion detection is a significant challenge due to the unavailability of facial and voice expressions. Considerable research has been done to identify textual emotions in high-resource languages such as English, French, Chinese, and others. Despite having over 300 million speakers and large volumes of literature available online, Urdu has not been properly investigated for the textual emotion detection task. To address this gap, this study makes two contributions: (1) the creation of a novel dialog-based corpus for Urdu (Contextual Urdu Text Emotion Detection Corpus). CUTEC contains 30,160 training and 5,509 testing labelled dialogues, where each dialogue consists of three Urdu contextual sentences. In addition, all dialogues are labelled using four emotion classes, i.e., Happy, Sad, Angry, and Other. As a second contribution (2) five deep learning models, i.e., RNN, LSTM, Bi- LSTM, GRU, and Bi-GRU have been trained and tested using CUTEC with different parametric settings. The highest results (Accuracy = 87.28 and F_1 = 0.87) are attained using a GRU-based architecture.

1. Introduction

Textual emotion detection in computational linguistics is the process of recognizing discrete emotions indicated in a text (Chatterjee, et al. 2019). Researchers widely studied four to six emotions from the text, i.e., happy, sad, angry, love, fear, and others (Baali and Ghneim 2019) (Bashir, et al. 2022).

The “Others” class is assigned to those particular text dialogues which do not enclose any emotion. Recently, EmoContext used these four classes for the textual emotion detection task (Chatterjee, et al. 2019). Human readers are normally capable of extracting emotions from the written line or a dialogue; however, doing so is generally hard for machines.

Textual emotion detection has the potential to assist multiple fields of computer science and information technology, such as human-computer interaction (Kao, et al. 2009) (Rincon, et al. 2016), dialogue classification (Baali and Ghneim 2019) (Bashir, et al. 2022), comment classification (Nagwani 2015), e-learning review improvement (V. Chang 2016), brand review identification (Syed and others 2015), detection of emotions on political events (Chang and Masterson 2020), product refinements through user reviews (Druck and Pang 2012), monitoring hostile and criminal communications (Panko and Beh 2002).

Generally, emotions are expressed through face, voice, body language, and text. Detecting emotions from facial expressions (Rani and Garg 2014) and speech (Yu, et al. 2001) has been widely explored by researchers, and now researchers are focusing on the text (Chatterjee, et al. 2019) (Bashir, et al. 2022) (Vijay, et al. 2018). Context (previous sentences) changes the emotion perceived through a sentence (Chatterjee, et al. 2019). Context can also decrease the ambiguity among emotions contained by the text, e.g., “مجھے رونا پسند ہے” often classified as Sad, but a dialogue User-1: “مجھے رونا پسند ہے” User- 2: “تم کیوں رو رہی ہو؟” User-1: “یہایک مذاق تھا” can be categorized as Happy. Therefore, it is clear from the above example of three Urdu sentences, that context has a significant impact on emotion detection and has the potential to change the class. The focus of this research work is on contextual emotion detection task for the Urdu language. Urdu is an important South Asian language with 300 million users (Hussain 2008) (Khan, et al. 2016) (Riaz 2010). There are millions of L1 (151 million) and L2 (149 million) speakers of Urdu (Rahman 2004). Urdu is Pakistan’s official national language with 11 million L1 speakers (Rahman 2004) (Naseer and Hussain 2009). Other countries with a large community of Urdu speakers include India, Bangladesh, the USA, Canada, and Europe (Hussain 2008) (Khan, et al. 2016). Urdu is considered morphologically rich, which means, many of its words can have more than 40 interpretations. Consequently, it is difficult to process as compared to morphologically poor languages (such as English) (Hussain 2008) (Naseer and Hussain 2009).

Benchmarked labeled corpora are required to construct, analyze and evaluate textual emotion detection systems. A range of corpora constructed for Textual Emotion Detection tasks is for English and other popular international languages (Chatterjee, et al. 2019) (Baali and Ghneim 2019) (Bashir, et al. 2022) (Syed and others 2015) (Syed, et al. 2010). Whereas, there is a need for standard benchmarked labeled corpora for Urdu (a poorly resourced language). This research work represents a novel benchmark corpus and designed various deep learning models for contextual Textual Emotion Detection task for Urdu.

We chose deep neural networks because of their ability to comprehend data's internal representation with multiple hidden layers that do not require handcrafted feature engineering (Baali and Ghneim 2019) (ayir, Yenidouan and Da 2018).

Our Contextual Urdu Text Emotion Detection Corpus (CUTEC) contains 35,509 labeled dialogues. Following the standard practice of EmoContext- SemEval-2019 Task 3, a single dialogue is composed of three contextual sentences. EmoContext corpus has been used as a source corpus and all instances were automatically translated using Google Translator. Furthermore, the quality of each dialogue has been ensured by three human language experts. Furthermore, five deep learning-based approaches have been designed and evaluated against CUTEC to clarify how our proposed dataset can be utilized for the construction and deployment of an automatic contextual emotion detection system for Urdu.

Overall, this research presents two contributions: (1) a novel Urdu dialog-based corpus (Urdu Text Emotion Detection Corpus). Every dialogue in CUTEC has three contextual Urdu sentences. There are 30,160 training dialogues and 5,509 testing dialogues. All dialogues are also labelled with one of four mood categories: happy, sad, angry, or other. (2) The second contribution is the training and testing of five deep learning models using CUTEC, including RNN, LSTM, Bi-LSTM, GRU, and Bi-GRU. The most accurate results were obtained using a GRU-based architecture (Accuracy = 87.28 and F_1 = 0.87).

The rest of this paper is organized as follows: Section 2 presents a review of existing contributions on the topic of emotion detection in various languages. Section 3 describes the generation process of the proposed corpus. Section 4 details the deep learning models applied to our proposed corpus. Section 5 shows the results and their analysis. Finally, Section 6 concludes the research presented in the article.

2. Related Work

Research on emotion detection started in the last few decades (Canales and Martinez 2014). The majority of work for emotion detection concentrated on English and other international languages (Chatterjee, et al. 2019) (Baali and Ghneim 2019) (Syed and others 2015), and a few attempts for Urdu (Bashir, et al. 2022), Arabic (Baali and Ghneim 2019), and Indonesian (Arifin, et al. 2014) languages. This section provides a review of principal literature on emotion detection tasks for different languages, including Urdu.

EmoContext presented in SemEval-19 task 3 is a significant contribution to contextual emotion detection for the English language (Chatterjee, et al. 2019).[1] A public dataset was developed containing 38k labeled dialogues distributed over 115k sentences. All dialogues were classified into four emotion classes, i.e., sad, happy, angry, and others. Although diverse approaches had been designed and experimented on this corpus, Bidirectional LSTM based models showed the highest result. The highest achieved micro-averaged F_1 score was 79.59.

Another prominent contribution to English emotion detection is (Ghosh, et al. 2020). A dataset of 18,921 labeled tweets distributed over six basic emotions, namely, happiness, anger, sadness, disgust, surprise, and fear. In the conducted experiments, the authors split training, validation, and testing sets containing 70%, 20%, and 10% tweets respectively. On this dataset, CNN, Bi-GRU, Bi-LSTM, and HAtED models were applied. By implementing the HAtED model, an accuracy of 81% was achieved.

There has been a proposal for emotion detection in the Chinese language (Lai, et al. 2020). A dataset consisting of 15,664 microblogs was prepared and divided evenly over seven emotion classes i.e., happiness, disgust, like, anger, fear, sadness, and surprise. Moreover, the authors applied a syntax-based graph convolution network (GCN) model on a given dataset. Overall, an 82.32% F_1 score was achieved.

Regarding Arabic emotion detection, an important research contribution is presented in (Baali and Ghneim 2019).[2] A dataset containing 5,600 tweets was split into two, with 5,064 tweets as the training dataset and 561 tweets as the testing dataset. The emotions were distributed over four classes sadness, fear, anger, and joy, where there were 1,400 tweets for each class. SVM, Naïve Bayes, Multi-Layer Perceptron, and CNN-based models were developed, and trained and tested on the dataset. The highest accuracy was of 99.82%, achieved for the CNN-DNN-based model.

---

1 https://aka.ms/emocontextdata Last visited: 19-Oct-2022

2 https://competitions.codalab.org/competitions/17751#learn_the_details-datasets Last visited: 19-Oct-2022

Hindi-language centering emotion detection effort is made in (Vijay, et al. 2018). This work contributed to Hindi-English parallel languages. A corpus containing 2,866 sentences, categorized into fear, disgust, surprise, happiness, sadness, and anger, containing 85, 291, 182, 595, 878, and 667 sentences respectively. In this dataset, 168 sentences have multiple emotion classes. The highest accuracy 58.2 is reported and is achieved using the SVM classifier.

For the Urdu language in literature, we found two significant contributions (1) sentiment-annotated lexicon (Syed, et al. 2010) and (2) RUED corpus (Arshad, et al. 2019). In the former contribution, researchers constructed a corpus of 753 processed reviews, containing 361 and 392 positive and negative reviews respectively, designed the SentiUnits-based method, and reported a maximum of 78% accuracy. While in the latter work, a corpus named "RUED" containing 10,000 Roman-Urdu-based sentences was developed. It was annotated using four classes, Happy, Sad, Angry, and Neutral, however, no experimental results were reported.

Recent advancement in the Urdu Emotion detection task were presented in (Bashir, et al. 2022). The authors created a corpus called the Urdu Nastalique Emotions Dataset (UNED), which contains 52k labeled sentences. Furthermore, six emotion classes were used for tagging: happy, sad, in love, angry, afraid, and neutral. The authors used word embedding, BOW, and TF-IDF approaches for feature extraction. Finally deep learning models were applied to this dataset for classification using LSTM and achieved 85% accuracy.

Overall, as far as we observed, the first limitation of existing research is that there is a lack of contextual text emotion detection corpus and techniques for Urdu. Furthermore, the research community largely focused on designing traditional machine learning approaches (Naïve Bayes, SVM, Multi-layered Perceptron) and there have been few attempts at crafting deep learning approaches for Urdu language processing tasks.

3. Corpus Generation Process

a. Source Corpus

Our proposed CUTEC corpus was created using EmoContext: Contextual Emotion detection corpus. EmoContext was originally constructed by Microsoft for an English emotion detection task and was presented in SemEval-2019 Task 3 (Chatterjee, et al. 2019). Overall, the corpus contained 38,424 dialogues and 115,272 sentences. The corpus was released with single training and two testing sets (Test 1 and Test 2). Test 1 carried 2,755 dialogues, and Test 2 contained 5,509 dialogues. Moreover, in Test 2 all dialogues were labeled, while in Test 1 all dialogues were unlabeled. Human experts assigned a label to the entire training and testing sets. Fleiss' Kappa score of the source corpus was 0.58 on training and 0.59 on testing sets, respectively. Each instance (dialogue) is composed of three sentences (turn-1, 2, and 3) to capture contextual information. Turn-1 included the utterance of a user, turn-2 carried the responsibility of a bot, and turn-3 again included the utterance of the user. The outputs were characterized into four labels, i.e., happy, angry, sad, and others. To ensure the quality of the labels of the corpus, formally, 50 judges were trained. Later, 7 judges assigned the final class. The emotionless sentences in the corpus were tagged as "others". The corpus was cleaned by following some important steps: (1) the personally identifiable Information (PII) such as (name, emails, and phone numbers) was removed from the corpus, (2) non-English conversations were removed, (3) offensive conversations were filtered, such as ethnic-religious content, reference to violence, crime or illegal activities. The corpus is freely available for research purposes under Creative Commons Attribution 4.0.[3]

3 https://aka.ms/emocontextdata Last visited: 19-Oct-2022

b. Translation via Google Translator

To develop our proposed Contextual Urdu Textual Emotion Corpus (CUTEC), the entire English text (omitting Test 1) of the source corpus was translated into Urdu using a world-widely recognized translator, i.e., Google Translator. Google Translate is a free multilingual neural machine translation service developed by Google, to translate text and websites from one language into another.[4] The reason for the suspension of Test 1 is that the instances were not labeled. For manual translation, each sentence was chosen from the dialogue (each dialogue was composed of three sentences) and placed as input text in Google Translator and the output was noted in a plain text file. As a whole, 106,527 sentences were manually translated into Urdu.

c. Validation of Translation

i. Sentence-wise invalid translation

Various sentences in the EmoContext corpus contained shorthand words, e.g., “plz”, “enaf”, “M8”. In one scenario “Enough” was written as “enaf”. A human can easily consider these two words alike, however, Google Translator was not able to recognize and translate them. To address this problem, (1) we manually added the correct translation and replaced shorthand words with their full form. For instance, by replacing “plz” with “Please” Google Translator produced the correct translation. Table 1. contains some examples chosen from the CUTEC corpus regarding sentence-wise invalid translation. This table holds the unique number for each instance in CUTEC, the turn of the sentence, the sentence as input to the translator, invalid output by the translator, and the correct translation of input.

Table 1. Sentence-wise Invalid Translations

Id	Turn	Sentence	Translator’s output	Correct translation
111	Turn 1	Plz darling 😭😭😭😭😭	Plz darling😭😭😭😭😭	براہ کرم پیاری 😭😭😭
190	Turn 1	That's enaf for me	یہ میرے لئے enaf ہے	یہ میرے لئے کافی ہے
151	Turn 2	Looks like a ficus.	ایک ficus کی طرح لگتا ہے.	انجیر کے درخت کی طرح لگتا ہے۔
168	Turn 2	I'll just stick with my M8. I see no benefit to upgrading.	میں صرف اپنے M8 کے ساتھ رہوں گا۔ مجھے اپ گریڈ کرنے میں کوئی فائدہ نہیں ہے۔	میں صرف اپنے میٹ کے ساتھ رہوں گا۔ مجھے اپ گریڈ کرنے میں کوئی فائدہ نہیں ہے۔
124	Turn 3	I'm not fine	میں &apos؛ ٹھیک نہیں ہوں	میں ٹھیک نہیں ہوں
174	Turn 3	I wish WE could hangout.	کاش ہم hangout کر سکتے۔	کاش ہم مل کر گھوم سکتے ہیں۔

4 https://translate.google.com/ Last visited: 19-Oct-2022

ii. Context-wise invalid translation

Another problem that occurred during the development of the dataset was contextually invalid translation. Google Translator, converted sentences based on words only, rarely focusing on context. For example, the sentence "gives you a patient smile I know, love." was translated into "مجھے معلوم ہے ، پیار ، آپ کو مریض کی مسکراہٹ دیتا ہے۔" if re-translate the Urdu sentence into English, "I know that, Love, gives you patient (medical patient, medical case) smile." which are literal meanings but not contextual. Google Translator considered the word "patient" as "medical case (medical patient)" but not as "calm, serene, or tolerant". In this scenario, the English sentences were correct, but the Urdu translation was contextually wrong. Linguistic experts solved these cases by writing the correct translation, later that translation was added to CUTEC. Table 2 contains further examples of such cases.

Table 2. Context-wise Invalid Translation

Id	Turn	Sentence	Translator's output	Correct translation
229	Turn 2	gives you a patient smile I know, love.	مجھے معلوم ہے ، پیار ، آپ کو مریض کی مسکراہٹ دیتا ہے۔	میں جانتا ہوں کہ آپ کو ایک خوش کن مسکراہٹ ملتی ہے۔ محبت.
252	Turn 1	Really of course	واقعی	واقعی ، یقینا
1089	Turn 3	Open your arms to cuddle.	اپنے بازوؤں کو کھونے کے ل. کھولیں۔	گلے ملنے کے لئے اپنے بازو کھولیں۔

iii. Emoji Validation

Generally, sentences containing emojis are more emotionally representational and bolder. During the development of the dataset, sentences containing emojis are sometimes translated incorrectly, or emojis are not present in translation. As an example, the translation of "please darling😭😭😭😭😭 " was given as "براہ کرم پیاری", neglecting emojis. However, "😭😭😭 براہ کرم پیاری" is the correct translation. This problem is solved using a space as a separator between English sentences and emojis, after space Google Translator identified emojis, and placed them accurately in translation.

iv. Label Validation

Class labeling is a supreme activity in the development of supervised learning methods. We selected EmoContext as an input corpus in which all instances had been labeled by 7 human judges. An interesting point observed in this study is that when we translate a sentence from English to Urdu, the emotions enclosed in the message remain the same. For instance, the English sentence "My friend is dead" is translated into Urdu as "میرا دوست فوت ہوگیا". However, both messages consolidate the same emotion class, that is, "sad". Thus, we translated all sentences, one by one, from EmoContext to CUTEC and class labels are the same in both corpora.

EmoContext corpus is translated with context and the sequence of turns of instances, so the validity of labels remains the same. Nevertheless, labels were reconfirmed by three linguistic experts. Both English and Urdu instances were provided to experts, and they identified the context of dialogues identically and applied the same label to Urdu dialogues.

d. Standardization of Corpus

CUTEC has been developed as a standard corpus, the corpus is available in a plain text file.[5] Each instance contains 3 sentences to cover contextual information, following the same standards used in the source corpus. The corpus will be released publicly after the publication of the paper under the license of Creative Commons Attribution 4.0 International license.

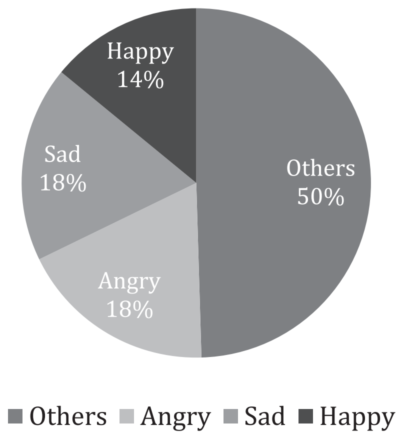


Figure 1. Class-wise Division of Training Dataset in CUTEC

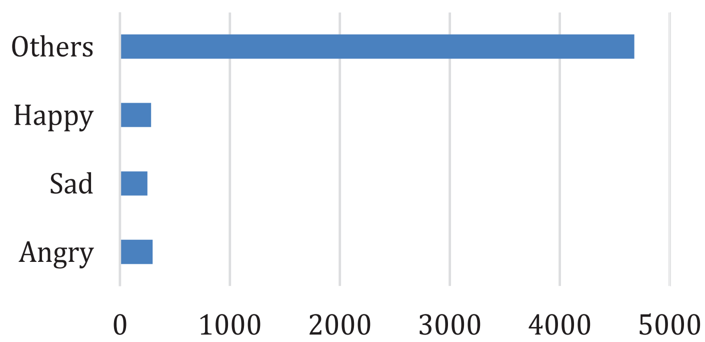


Figure 2. Division of Classes in Testing Dataset

e. Characteristics of Corpus

CUTEC (Contextual Urdu Textual Emotion Corpus) contains 606,548 words in total and 35, 509 labeled dialogues. It is divided into 30,000 instances (90,000 sentences), leading to 530, 381 words as the Training dataset. While the Testing dataset contains 5,509 instances (16,524 sentences), which

5 https://comsatsnlpgroup.wordpress.com/ Last Visited 11-Nov-2022

amounts to 76,167 words. The credibility of the dataset remains as the source corpus because translation only converts words whereas context remains the same. Figure 1 illustrates the distribution of classes in the training dataset. Overall, 50% of instances belong to the class others, 18% to the class Anger, 18% to Sad, and 14% to Happy. Figure 2 contains the distribution of classes in the test dataset, where "Others" is labeled as 85% of the testing dataset.

4. Experimental Setup

a. Most Frequent Class

Most Frequent Class also referred to as Majority Class classification is a simple approach used by a large number of researchers as a baseline approach to an emotion detection system (Inkpen, Keshtkar and Ghazi 2009) (Krcadinac, et al. 2013). This approach works on a simple phenomenon that assigns the majority class to all instances in the testing set and calculates the percentage of accuracy. Table 5 contains the result of this approach, that is, 84.89%.

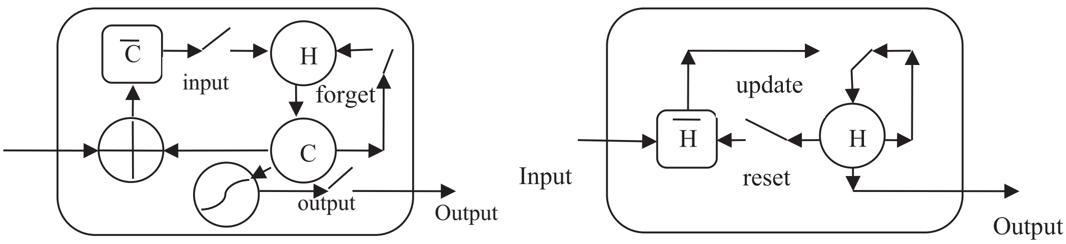
b. Model Description

Figure 4 describes the deep neural network-based architecture used for the Urdu Emotion Detection task. As the figure shows, our model inputs labeled instances from a newly developed corpus (CUTEC). A single instance is composed of three sentences. Further, this model concatenated all three sentences as a single sentence and used it with the class label to train and test deep learning models. Averaged results in terms of Accuracy, Precision, Recall, and F_1 are presented in Table 5.

Furthermore, all concatenated instances (a combination of three sentences) are used as input to the Embedding layer[6]. The purpose of the Embedding layer is to turn the input (words) into low-dimensional vector representations (Fang, et al. 2016) (Hochreiter and Shmidhuber 1997). These Embeddings can conveniently be used to develop embedding-based features. This study used embedding-based features to develop various deep neural network approaches.

Overall, we designed three-layered based models, (1) a single hidden layer, (2) two hidden layers, and (3) three hidden layers. The single hidden layer used either one type of processing unit at one time, i.e., Simple Recurrent Neural Networks (SRNNs) (Bullinaria 2013), Long Short-Term Memory (LSTM) (Hochreiter and Shmidhuber 1997) (Figure 3. (a) Shows an LSTM unit), Bidirectional Long Short-Term Memory (Bi-LSTM) (Zhang, et al. 2015), Gated Recurrent Units (GRUs) (Abdul, Mageed and Ungar 2017) (Figure 3. (b) Shows GRU Unit) and Bidirectional Gated Recurrent Units (Bi-GRUs) (Yu, Zhao and Wang 2019). The reason for using Recurrent Neural Networks based approaches is well suited for sequence learning and text processing tasks (Baali and Ghneim 2019) (Bashir, et al. 2022). These models return the state-of-the-art performance for different NLP tasks (Howard and Ruder 2018).

6 We used Keras (https://keras.io/) to implement Deep Learning models. According to the specifications of Keras, the embedding layer can only be used as a first layer of the model as we did in this study.



(a) Long-Short-Term Memory Unit (b) Gated-Recurrent-Unit

Figure 3. The Architecture of the Basic LSTM Unit (obtained from (Hochreiter and Shmidhuber 1997)) and GRU Unit (obtained from (Zhao, et al. 2019))

Figure 3 (a) illustrates that LSTM is composed of three gates, namely, input, output, and forget. Figure 3 (b) GRU is comprised of a comparatively simple architecture with two gates rest and update. C and C illustrate previous and upcoming memory cell contents. Whereas H and H are the candidate activation functions.

After using different types of processing units added on the hidden layer(s), we added a dropout layer to the model. Dropout is a regularization technique used to reduce the overfitting of the model. A dropout value of 0.2 (20%) is widely used to avoid overfitting (Abdullah, Hadzikadicy and Shaikhz 2018) (Krishna and Patil 2020). Lastly, on the output layer, Sigmoid function is used to classify each dialogue concerning its emotion class (Wang and others 2020) (a numeric representation of each class is available in Table 3).

Table 3. Corpus Classes and Respective Numeric Label

Class	**Angry**	**Sad**	**Happy**	**Others**
Numeric Label	0	1	2	3

Table 4 represents the parametric values used by deep neural networks for the textual emotion detection task. Experiments were conducted using three different hidden layers (from 1 to 3), a fixed number of 100 neurons in each hidden layer, Tanh as activation function, applied a constant 0.001 learning rate, batch size of 32, validation split as 20% and persistent dropout value of 0.2 (20%). The output layer used four specified neurons with a Sigmoid activation function. This study used an optimal value of epochs, whereas a large number of epochs can cause an overfitting problem, however, small numbers may lead to an under-fit model (Zhang, Zhang and Jiang 2019). During experiments, observed different values of the number of epochs (ranges from 5 to 40), and found 20 as the most optimal value.

Table 4. Parametric Configurations for All Experiments

Learning Rate	**Default**
Number of Epochs	20
Batch Size	32
Activation Function in Hidden Layers	Tanh
Activation Function in Output Layer	Sigmoid

(continued)

Table 4. Parametric Configurations for All Experiments (continued)

Learning Rate	**Default**
Embedding	1000
Number of Neurons in Hidden Layer	100
Number of Neurons in Output Layer	4
Validation Split	0.2
Dropout	0.2

c. Evaluation Methodology

In this study, the contextual emotion detection task has been studied as multi-class classification. The model has to distinguish four classes, i.e., happy, sad, angry, and others. CUTEC has been used for training and testing deep neural network-based models. In CUTEC, data had already been split into two sets, that is, there were a total of 30,000 training (Figure 1 shows the class-wise detail of training instances) and 5,509 testing instances (Figure 2 shows the class-wise detail of testing instances). This study used the entire data for experimentations and reported results in Table 5. Furthermore, this study used the Keras embedding layer for LSTM, GRU, Bi-LSTM, and Bi-GRU models in the input layer.

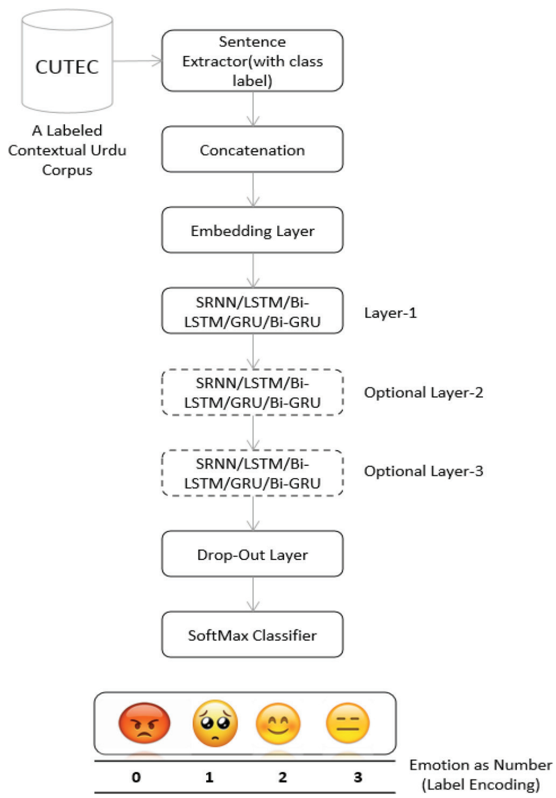


Figure 4. The Architecture of the Deep Learning Model Used for the Urdu Emotion Detection Task.

d. Evaluation Measure

In this study, the evaluation has been carried out using the widely applied evaluation measures i.e. Accuracy, Precision, Recall, and F_1 (Bashir, et al. 2022) (Abdullah, Hadzikadicy and Shaikhz 2018) (Acheampong, Wenyu and Nunoo-Mensah 2020). These measures are largely used by the research community to evaluate the performance of emotion detection systems (Acheampong, Wenyu and Nunoo-Mensah 2020). The Following equations have been used to calculate the measures, which are obtained from (Saeed, et al. 2019) (Al-Saqqa, Abdel-Nabi and Awajan 2018).

The Accuracy of a system is defined as the total number of correct predictions.

$$Accuracy = \frac{CorrectCases}{Allcases} \times 100 \quad (1)$$

The precision of a system is the ability of the classifier not to call a negative sample positive.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \quad (2)$$

The Recall of a system is the ability of a classifier to find all the positive samples.

$$Recall = \frac{TruePositive}{TruePpsitive + FalsNegative} \quad (3)$$

F_1 measure is a specific relationship (harmonic mean) between precision (Pre) and recall (Rec).

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (4)$$

Accuracy, Precision, Recall, and F_1 scores have been computed using all the 5.509k testing dataset and reported in this study.

5. Results and Analysis

Table 5 shows the results in terms of Accuracy, Precision, Recall, and F_1 obtained using five deep learning methods, i.e., LSTM, Bi-LSTM, GRU, Bi-GRU, and Simple RNN with three-layer based approaches, i.e., 1, 2 and 3 hidden layers.

Overall, on our proposed corpus CUTEC, the highest results (Accuracy = 87.28% and F_1= 0.87) have been achieved using the GRU based model, and the lowest results (Accuracy = 66.62% and F_1= 0.72) have been attained using the LSTM-based model. The results highlight the fact that the models that used two hidden layers are the most appropriate for Urdu contextual emotion detection task. The possible reason is that the training data that have been used for experimentation were of a moderate size (approximately 30K instances). To train the neural model with three or more layers, more training data is required which is unfortunately not available for the Urdu language. An important conclusion can be drawn from the results, namely, that a large number of deep learning methods showed results higher than the baseline approach (Accuracy = 84.89%). Further, this study presents a more accurate model (2-layers GRU architecture) in comparison to the previous studies which had carried out the same task (Bashir, et al. 2022) (Rehman and Bajwa 2016).

Table 6 compares the features of our proposed corpus with existing corpora for the textual emotion detection task. As can be noted from the above table our proposed corpus (CUTEC) has more tagged sentences (106, 527 sentences) compare to the existing corpora. Furthermore, we used 4 classes, which are widely used for textual emotion detection task (Arshad, et al. 2019).

Table 5. Results obtained using different deep learning modes for contextual Urdu emotion detection task

No of Layers	Corpus	Model	Accuracy	F_1-Score	Precision	Recall
1	CUTEC	LSTM	66.62%	0.72	0.72	0.67
		Bi-LSTM	82.12%	0.83	0.83	0.82
		GRU	85.55%	0.86	0.86	0.86
		Bi-GRU	86.60%	0.87	0.87	0.87
		Simple RNN	67.51%	0.72	0.72	0.68
2	CUTEC	LSTM	68.22%	0.73	0.79	0.74
		Bi-LSTM	67.53%	0.72	0.80	0.68
		GRU	**87.28%**	**0.87**	**0.88**	**0.87**
		Bi-GRU	86.15%	0.86	0.87	0.86
		Simple RNN	73.95%	0.76	0.80	0.74
3	CUTEC	LSTM	79.31%	0.81	0.85	0.79
		Bi-LSTM	81.21%	0.83	0.85	0.81
		GRU	85.02%	0.86	0.87	0.85
		Bi-GRU	83.92%	0.85	0.87	0.84
		Simple RNN	84.90%	0.78	0.72	0.85
(Bashir, et al. 2022)	UNED	Bi-LSTM	85.30%	0.84	0.87	0.85
(Rehman and Bajwa 2016)	Urdu-Lexicon	Text Similarity	66.00%	0.73	0.69	0.79
MFC (Baseline)		-	84.89%	-	-	-

Table 6. Comparison of CUTEC with Existing Corpora

Corpus	Language	Emotion Classes	Labeled Sentences
UNED (Bashir, et al. 2022)	Urdu	6	52,000 sentences and 2,000 paragraphs
RUED (Arshad, et al. 2019)	Roman Urdu	4	10,000 sentences
Urdu-Lexicon (Rehman and Bajwa 2016)	Urdu	3	-
CUTEC (current study)	**Urdu**	**4**	**35, 509 x 3 = 106, 527 sentences**

Table 7 describes the class-wise detail of F_1 achieved using five deep-learning models. The model that shows the top results, i.e., GRU with two hidden layers is stronger on “Angry and Sad” than “Sad and Happy” in terms of F_1 score. Further, LSTM and Simple are not well suited for Urdu contextual emotion detection task.

Table 7. Class-wise Detail of F_1-score Results

No of Layers	Model	Angry	Sad	Happy	Others
1	LSTM	0.28	0.24	0.23	0.80
	Bi-LSTM	0.54	0.46	0.41	0.90
	GRU	0.58	0.52	0.53	0.92
	Bi-GRU	0.59	0.53	0.53	0.92
	Simple RNN	0.38	0.22	0.16	0.80
2	LSTM	0.27	0.26	0.16	0.81
	Bi-LSTM	0.35	0.20	0.25	0.80
	GRU	**0.61**	**0.54**	**0.56**	**0.93**
	Bi-GRU	0.58	0.54	0.48	0.92
	Simple RNN	0.33	0.26	0.18	0.85
3	LSTM	0.47	0.46	0.42	0.88
	Bi-LSTM	0.54	0.48	0.40	0.89
	GRU	0.59	0.53	0.51	0.91
	Bi-GRU	0.58	0.50	0.51	0.91
	Simple RNN	0.00	0.00	0.00	0.92

Table 8 presents class-wise detail of Precision computed using different deep-learning models. The results show that in terms of Precision, for the class "Others" the most appropriate methods are that usually use three hidden layers.

Table 8. Class-wise Detail of Precision Results

No of Layers	Model	Angry	Sad	Happy	Others
1	LSTM	0.20	0.17	0.19	0.90
	Bi-LSTM	0.45	0.39	0.36	0.93
	GRU	0.51	0.45	0.55	**0.94**
	Bi-GRU	0.56	**0.51**	0.52	0.93
	Simple RNN	0.29	0.15	0.13	0.90
2	LSTM	0.28	0.14	0.19	0.90
	Bi-LSTM	0.35	0.20	0.25	0.80
	GRU	**0.58**	0.48	**0.63**	0.93
	Bi-GRU	0.52	0.48	0.58	0.93
	Simple RNN	0.23	0.20	0.30	0.90
3	LSTM	0.35	0.38	0.35	**0.94**
	Bi-LSTM	0.44	0.39	0.34	0.93
	GRU	0.52	0.47	0.45	**0.94**
	Bi-GRU	0.51	0.40	0.47	**0.94**
	Simple RNN	0.00	0.00	0.00	0.85

For classes “Happy” and “Angry” the most suitable method is GRU with two hidden layers in terms of Precision. For class “Sad” Bi-GRU shows the highest results in term of Precision.

Table 9. illustrates the Recall of models. Where the highest recall over the “Angry” class is achieved by LSTM-3. Bi-GRU-3 performed highest over the “Sad” Class. The “Happy” class’s maximum recall of 0.59 was achieved by GRU-3, and the highest recall over “Others” was achieved by SimpleRNN-3 which was 1.0. if neglect SimpleRNN-3 then by Bi-GRU-1, Bi-GRU-2, GRU-2 is 0.92.

Table 9. Class-wise Detail of Recall Results

No of Layers	Model	Angry	Sad	Happy	Others
1	LSTM	0.48	0.42	0.29	0.72
	Bi-LSTM	0.67	0.56	0.47	0.87
	GRU	0.69	0.61	0.50	0.90
	Bi-GRU	0.63	0.56	0.54	0.92
	Simple RNN	0.56	0.42	0.22	0.72
2	LSTM	0.43	0.42	0.21	0.27
	Bi-LSTM	0.48	0.40	0.35	0.72
	GRU	0.63	0.63	0.51	0.92
	Bi-GRU	0.64	0.63	0.40	0.92
	Simple RNN	0.58	0.36	0.13	0.81
3	LSTM	**0.71**	0.58	0.51	0.83
	Bi-LSTM	0.69	0.62	0.47	0.84
	GRU	0.70	0.61	**0.59**	0.89
	Bi-GRU	0.67	**0.69**	0.56	0.87
	Simple RNN	0.00	0.00	0.00	**1.00**

6. Conclusion

Urdu is an under-resourced language with a significant number of speakers. Urdu requires labeled corpora to enhance different NLP tasks. As presented in the paper, the contribution of this study is a newly constructed and publicly available benchmark corpus called CUTEC. The corpus contains 35,509 labeled dialogues, where each dialogue is composed of three Urdu sentences. In addition to the construction of the corpus, this study experimented with five widely used deep learning models to check their stability. The results of our contextual emotion detection task show that GRU achieves the best performance. In the future, we are interested in studying transfer learning, ensemble learning, and generative adversarial networks for the task of emotion detection in Urdu.

References

Abdul, M., M., and Lyle, U., 2017. Emonet: Fine-grained emotion detection with gated recurrent neural networks. Proceedings of the 55th annual meeting of the association for computational linguistics, Vancouvre, Canada, ACL, 1, 718-728.

Abdullah, M., Mirsad, H., and Samira, S., 2018. SEDAT: sentiment and emotion detection in Arabic text using CNN-LSTM deep learning. 17th IEEE international conference on machine learning and applications (ICMLA), Florida, USA, IEEE, 5-840.

Acheampong, F. A., Chen, W., and Henry N. M., 2020. Text-based emotion detection: Advances, challenges, and opportunities. Engineering Reports, 2(7), e12189.

Al-Saqqa, S., Heba, A., N., and Arafat, A., 2018. A survey of textual emotion detection. 8th International Conference on Computer Science and Information Technology (CSIT). Amman, Jordan, 136-142.

Arifin, A., Z., Yuita A., S., Evy K., R., and Siti M., 2014. Emotion Detecion of Tweets in Indonesian Language using Non-Negative Matrix Factorization. International Journal of Intelligent Systems and Applications 6(9), 54.

Arshad, M., U., Muhammad, F., B., Adil, M., Waseem, S., and Mirza, O., Beg., 2019. Corpus for emotion detection on roman urdu. 22nd International Multitopic Conference (INMIC). Islamabad, Pakistan, 1-6.

Ayir, A., Iil, Y., and Hasan, D., 2018. Feature extraction based on deep learning for some traditional machine learning methods. 3rd International Conference on Computer Science and Engineering (UBMK), USA, 494-497.

Baali, M., and Nada, G., 2019. Emotion analysis of Arabic tweets using deep learning approach. Journal of Big Data. 6(1), 1-12.

Bashir, M., F., Abdul R., J., Muhammad U., A., Thippa R., G., Waseem S., and Mirza O., B., 2022. Context aware emotion detection from low resource urdu language using deep neural network. Transactions on Asian and Low-Resource Language Information Processing, 2022.

Bullinaria, J., A., 2013. Recurrent neural networks. Neural Computation: Lecture 12.

Canales, L., and Barco, P., M., 2014. Emotion detection from text: A survey. Proceedings of the workshop on natural language processing in the 5th information systems research working days (JISIC), Quito, Ecuador, 37-43.

Chang, C., and Michael, M. 2020. Using word order in political text classification with long short-term memory models. Political Analysis 28(3), 395--411.

Chang, V., 2016. Review and discussion: E-learning for academia and industry. International Journal of Information Management 36(3), 476--485.

Chatterjee, A., Kedhar, N., N., Meghana, J., and Puneet, A., 2019. SemEval-2019 Task 3: EmoContextContextual Emotion Detection in Text. International Workshop on Semantic Evaluation. Minneapolis: MIT press, 39-48.

Druck, G., and Bo, P., 2012. Spice it up? Mining refinements to online instructions from user generated content. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Jejo Island, Korea, 545-535.

Fang, W., Jianwen, Z., Dilin, W., Zheng, C., and Ming, L., 2016. Entity disambiguation by knowledge and text jointly embedding. Proceedings of the 20th SIGNLL conference on computational natural language learning. Berlin, Germany, 260-269.

Ghosh, Soumitra, et al. 2020. Annotated Corpus of Tweets in English from Various Domains for Emotion Detection. Proceedings of the 17th International Conference on Natural Language Processing (ICON). Patna, India, 460-469.

Hochreiter, S., and Jurgen, S. 1997. Long Short-Term Memory. Neural Computation 9(8), 1735-1780.

Howard, J., and Sebastian, R., 2018. Universal Language Model Fine-tuning for Text Classification. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. Melbourne: Association for Computational Linguistics, 328-339.

Hussain, S., 2008. Resources for Urdu language processing. Proceedings of the 6th workshop on Asian Language Resources.

Inkpen, D., Fazel, K., and Diman, G., 2009. Analysis and generation of emotion in texts. KEPT, 3--13.

Kao, E., C., C., Chun-Chieh, L., Ting-Hao, Y., Chang-Tai, H., and Von-Wun, S., 2009. Towards text-based emotion detection a survey and possible improvements. International Conference on Information Management and Engineering, New Zeeland, IEEE, 70-74.

Khan, W., Ali D., Jamal, A N., and Tehmina, A., 2016. A survey on the state-of-the-art machine learning models in the context of NLP. Kuwait journal of Science. 43(4).

Krcadinac, U., Philippe, P., Jelena, J., and Vladan, D., 2013. Synesketch: An open source library for sentence-based emotion recognition. IEEE Transactions on Affective Computing. 4(13). 312-325.

Krishna, DN., and Ankita, P., 2020. Multimodal Emotion Recognition Using Cross-Modal Attention and 1D Convolutional Neural Networks. Interspeech. 4243-4247.

Lai, Y., Linfeng, Z., Donghong, H., Rui, Z., and Guoren, W., 2020. Fine-grained emotion classification of Chinese microblogs based on graph convolution networks. World Wide Web, 23(5), 2771--2787.

Nagwani, N., K., 2015. A comment on "a similarity measure for text classification and clustering". IEEE Transactions on Knowledge and Data Engineering, 27(9), 2589--2590.

Naseer, A., and Sarmad, H., 2009. Supervised word sense disambiguation for Urdu using Bayesian classification. Center for Research in Urdu Language Processing, Lahore, Pakistan, 2009.

Panko, R., R, and Hazel, G. B., 2002. Monitoring for pornography and sexual harassment. Communications of the ACM, 45(1), 84--87.

Rahman, T., 2004. Language policy and localization in Pakistan: Proposal for a paradigmatic shift. In SCALLA Conference on computational linguistics, Pakistan, 1-19.

Rani, J., and Kanwal G., 2014. Emotion detection using facial expressions-A review. International Journal of Advanced Research in Computer Science and Software Engineering, 4(4).

Rehman, Z., U., and Imran, S., B., 2016. Lexicon-based Sentiment Analysis for Urdu. Sixth international conference on innovative computing technology (INTECH). Dublin, IEEE,497-501.

Riaz, K., 2010. Rule-based named entity recognition in Urdu. Proceedings of the 2010 named entities workshop. Uppsala, Sweden, ACL, 126-135.

Rincon, J., Jose, L., P., Juan, L., P., Vicente, J., and Carlos, C., 2016. Adding real data to detect emotions by means of smart resource artifacts in MAS. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal, 5(4), 85.

Saeed, A., Rao, M., A., N., Mark, S., and Paul, R., 2019. A word sense disambiguation corpus for Urdu. Language Resources and Evaluation, 53(3), 397--418.

Syed, A. Z, and others. 2015. Applying sentiment and emotion analysis on brand tweets for digital marketing. IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), Jordan, IEEE, 1-6.

Syed, A., Z, Muhammad, A., Enriquez, M., and Maria, A., 2010. Lexicon Based Sentiment Analysis of Urdu Text Using SentiUnits." Mexican international conference on artificial intelligence. Maxico, IEEE, 32-43

Vijay, D., Aditya, B., Vinay, S., Syed, S., A., and Manish, S., 2018. Corpus creation and emotion prediction for Hindi-English code-mixed social media text. Proceedings of the 2018 conference of the North American chapter of the Association for Computational Linguistics: student research workshop, New Orleans, Lousiana, USA, ACL, 128-135.

Wang, Z., and others. 2020, Text emotion detection based on Bi-LSTM network. Academic Journal of Computing \& Information Science, 3(3).

Yu, F., Eric, C., Ying-Qing, X., and Heung-Yeung, S., 2001. Emotion detection from speech to enrich multimedia content. Pacific-Rim Conference on Multimedia, Bejing, China, 550--557.

Yu, Q., Hui, Z., and Zuohua, W., 2019. Attention-based bidirectional gated recurrent unit neural networks for sentiment analysis. Proceedings of the 2nd International Conference on Artificial Intelligence and Pattern Recognition, Bejing, China, 116-119.

Zhang, H., Lin, Z., and Yuan, J., 2019. Overfitting and underfitting analysis for deep learning based end-to-end communication systems. 11th International Conference on Wireless Communications and Signal Processing (WCSP), Xian, China, 1-6

Zhang, S., Dequan, Z., Xinchen, H., and Ming, Y., 2015. Bidirectional long short-term memory networks for relation classification. Proceedings of the 29th Pacific Asia conference on language, information and computation, Shanghai, China, 73-78.

Zhao, H., Zhongxin, C., Hao, J., Wenlong, J., Liang, S., and Min, F., 2019. Evaluation of three deep learning models for early crop classification using sentinel-1A imagery time series—A case study in Zhanjiang, China. Remote Sensing, 11(22), 2673.

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 11 N. 4 (2022), 507-518
eISSN: 2255-2863
DOI: https://doi.org/10.14201/adcaij.27349

Sentiments Analysis of Covid-19 Vaccine Tweets Using Machine Learning and Vader Lexicon Method

Vishakha Arya[1], Amit Kumar Mishra[2],
Alfonso González-Briones[3,4,5]

[1,2]School of Computing- Computer Science & Engineering, DIT University, Dehradun 248001, India
[3]Research Group on Agent-Based, Social and Interdisciplinary Applications (GRASIA), Complutense University of Madrid, 28040 Madrid, Spain;
[4]BISITE Research Group, University of Salamanca. Calle Espejo s/n. Edificio Multiusos I+D+i, 37007, Salamanca, Spain: alfonsogb@usal.es
[5]Air Institute, IoT Digital Innovation Hub, Calle Segunda 4, 37188, Salamanca, Spain
Vishakha.27arya@gmail.com, aec.amit@gmail.com, alfonsogb@ucm.es

KEYWORDS

sentiment analysis, VADER lexicon, Twitter, vaccine tweets, Covid-19, classification model

ABSTRACT

The novel Coronavirus disease of 2019 (COVID-19) has subsequently named Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2) have tormented the lives of millions of people worldwide. Effective and safe vaccination might curtail the pandemic. This study aims to apply the VADER lexicon, Text Blob, and machine learning approach: to analyse and detect the ongoing sentiments during the affliction of the Covid-19 pandemic on Twitter, to understand public reaction worldwide towards vaccine and concerns about the effectiveness of the vaccine. Over 200000 tweets vaccine-related using hashtags # Covid Vaccine # Vaccines # CornavirusVaccine were retrieved from 18 August 2020 to 20 July 2021. Data analysis conducted by VADER lexicon method to predict sentiments polarity, counts and sentiment distribution, Text Blob to determine the subjectivity and polarity, and compared with two other models such as Random Forest (RF) and Logistic Regression (LR). The results determine sentiments that public have a positive stance towards a vaccine follows by neutral and negative. Machine learning classification models performed better than the VADER lexicon method on vaccine Tweets. It is anticipated this study aims to help the government in long run, to make policies and a better environment for people suffering from negative thoughts during the ongoing pandemic.

1. Introduction

In Jan 2020, China confirmed the human transmission of novel coronavirus. The WHO declared the outbreak as a pandemic. Now, it has been more than a year since the corona virus has prevailed in the society (Brunier, A. et al., 2020). In Jan 2021, a new mutated variant of coronavirus identifies in the United Kingdom called B.1.1.7, B.1.351 in South Africa, P.1 in Brazil, B.1.617 in India, B.1.526 in New York and many other mutants reported worldwide (Centre for Disease control and Prevention, 2021). These new variants have a high tendency of spreading more swiftly leads to an increased number of Covid-19 cases means more hospitalization and probably more deaths (Terry, M. 2021). Recently, researchers have reported a unique combination of «three mutant variants» or «Bengal strain» in the SARS-CoV-2 virus detected in India (Cherian, S., Potdar, et al., 2021). It is continuously evolving to elude the human immune system. It is continuously evolving to elude the human immune system. To curtail the COVID-19 pandemic vaccine is very important. As the vaccination program roll-out, people started sharing their views on social media sites like Twitter etc (World Economic Forum, 2020). Analyzing these views or opinions is mandatory to under- stand the sentiments of people globally (Khan, H. et al., 2020). To control the false lies related to vaccines among the public to prevent the spread of the pandemic (Chancellor et al., 2020). Monitoring sentiments of the public on social media sites (Chilwal, B. & Mishra, A. K., 2021) helps the government and other institutes to understand the public behaviour towards the vaccine and make effective policies to roll out immunization programs to mitigate the coronavirus pandemic (Saladino et al., 2020).

Sentiment analysis is one of the ways to analyze the opinions shared by people on Twitter by text analysis (Yazdavar et al, 2018). Social media data is highly unstructured, EmotionAI or affective computing to detect emotion using Natural Language Processing, textual analysis, deep learning, and machine learning models to extract and quantify the posts from social media sites on subjective text (Subhani et al., 2017). By using data mining techniques, extract the data from large databases. By applying natural language processing (NLP), identifying the thoughts and opinions of the text is known as opinion mining or sentiment analysis (Baheti et al., 2019). In this paper, large dataset of covid-19 vaccine related tweets by users globally has been considered (Ritchie, H et al., 2020). To specify these sentiments TextBlob and VADER lexicon methods to determine the polarity, subjectivity, and compound score of the target text (Nguyen et al., 2014). Data analysis was done by the VADER lexicon method to predict sentiments polarity, counts and sentiment distribution, TextBlob to determine the subjectivity and polarity (Aldarwish et al., 2017). It classifies the text as positive, negative, or neutral; also, how much the text is positive or negative (Abd El-Jawad et al., 2018). It further classifies based on polarity as (positive, negative, neutral, compound score) and machine learning classification model such as random forest (RF) and logistic regression (LR) to attain precision, recall, F1-score (suitability), and accuracy to compare the performance with lexicon rule-based method (Raichur et al., 2017). Furthermore, section 2 elaborates the study sites and methodology, section 3 results and experiments, section 4 discussions, and sections 5 conclusions. This study sceptic helps the government and other organization to develop trust in immunization program and the goal of achieving herd immunity.

2. Material and Methods

2.1. Study sites

Data extracted from Twitter is one of the largest repositories of social media post. Tweets were posted in the English language globally from 18 August 2020 to 20 July 2021 using Twitter API. Then, Twitter Ids were hydrated or retrieved using Hydrator application to get tweets related to the covid vaccine like «CovidVaccine», «Vaccines», «CoronavirusVaccine», and «COVID19». Total tweets hydrated using related keywords were 207006 collected geospatially.

2.2. Sentiment Analysis

The filtered dataset of tweets related to the covid vaccine tweets pre-processed initially as raw data contains noise, missing values, re-tweets, and outliers (Deshpande et al., 2017). A hybrid model developed for subjective sentiment analysis. A rule-based lexicon Valance Aware Dictionary and Sentiment Reasoner (VADER) keyed to the expression expressed on social media. Text Blob analyses sentiments on subjectivity and polarity. Subjectivity refers to expression of a text given as [0, 1] whereas polarity of text indicated it is positive or negative [-1, 1]. Acquired subjectivity, polarity, counts, compound score, and sentiment distribution combined with the machine learning classification model: Random Forest, and Logistic regression (Shatte et al., 2019) (Rana, M. et al., 2022).

2.2.1. Text pre-processing

For building any machine learning model, data pre-processing is the primary task (Dubey, 2020). Text pre-processing involves removing noise, outliers, conversion to lower case, removing URL, removing punctuations, stemming, lemmatization, stopwords, and normalization. Text encoded into a numeric vector as machine learning understands the numeric format. N-grams, TF-IDF, and count vectorization are some encoding techniques to quantify the text into numeric (Desai et al., 2016). N-grams represent sequencing of n-words in each text or corpus. 1-gram means unigram, 2-gram means bigram, and 3-gram means trigram (Cavazos-Rehg et al., 2016). Figure 1 shows N-gram representation unigram, bigram, and trigram of vaccine tweets.

2.2.2. Lexicon Methods

Covid-19 vaccination tweets (n=2, 00,000) pre-processed using Natural Language Processing (NLP) with Natural Language Toolkit (NLTK) in python and «re» package. To normalize the text noise removal, outliers, text enrichment, and POS tagging improve the efficacy to analyses the sentiments (Bonta, V. et al., 2019).

VADER lexicon method is a tool used for labelling according to the semantic orientation of a text positive, negative, and neutral. It classifies the intensity of a text, how much positive or negative. It classified sentiment analysis into three steps: subjective, semantic, and polarity. Sentiment classification is done on subjective text rather than objective text. The subjective text defines the opinion, thoughts, emotions of a person help for semantic classification, and objective text carries relevant information about the topic. SentimentIntensityAnalyzer() creates an object sid.polarity_scores() provides dictionary gives polarity of a text classifies into positive, negative, neutral, and compound score. Sentiment scores were accredited to the text according to their semantic orientation if compound score > 0.5 assigns as «positive», compound score < -0.5 assigns as «negative», and compound score = 0 assigns as «neutral». Table 1 shows sentiment polarity of tweets assigning their scores and values relat- ed to semantic orientation. The compound score lexicon calculated by aggregating all sentiment scores positive and negative sentiments then normalized between -1(negative) to +1 (positive). Equation (1) intimates the «Y» sum of all polarity words. Alpha (α) is a constant word.

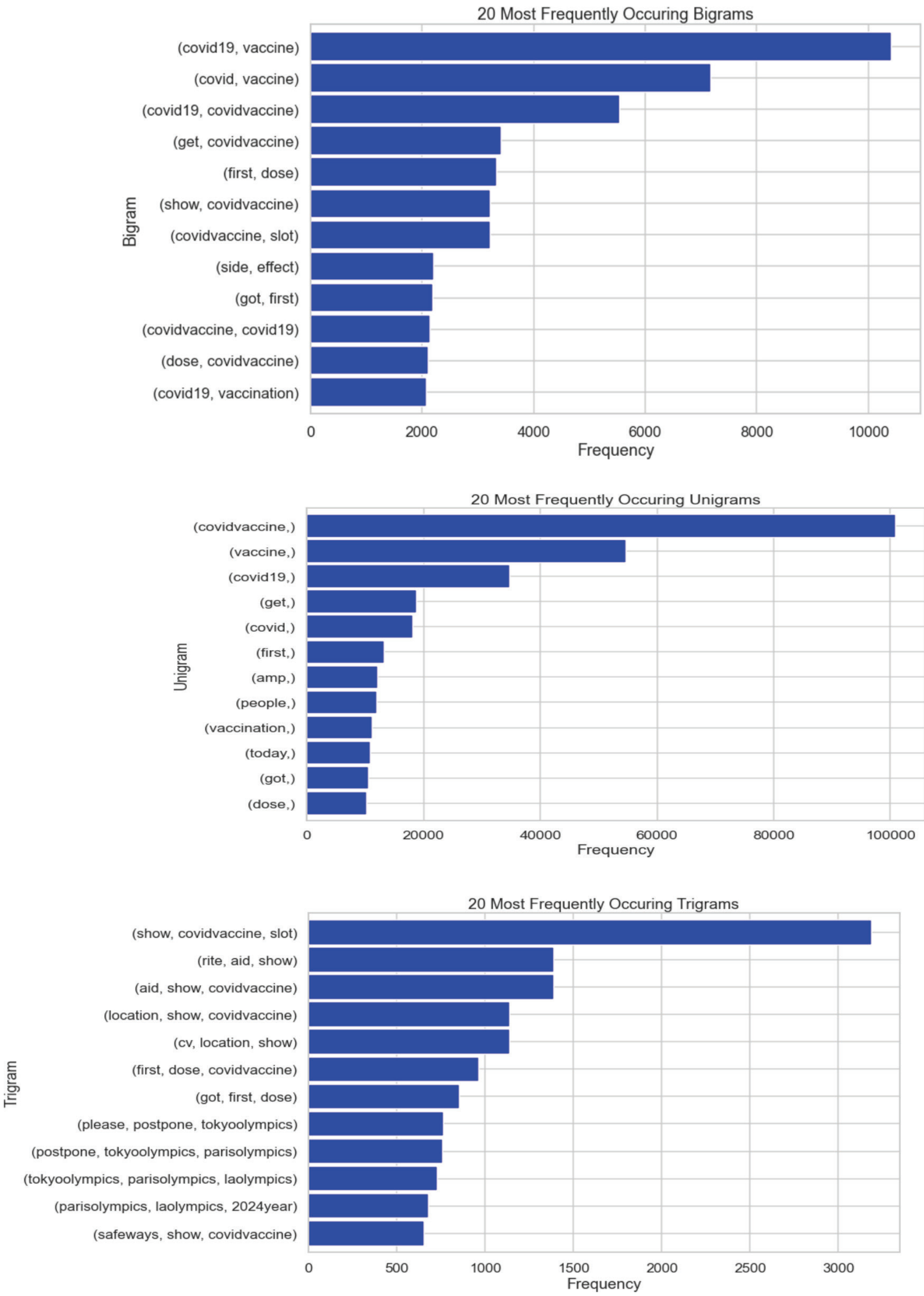


Figure 1. Unigram, Bigram, and Trigrams representation of covid-19 vaccine tweets

$$\text{Compound Score} = \frac{Y}{\sqrt{Y^2 + \alpha}} \tag{1}$$

By leveraging TextBlob (Gbashi S et al., 2021) library sentiment scores can be extracted from a text or document. Subjectivity and polarity of a text obtained with Text Blob. Polarity assigns positive and negative ranges from -1 to 1 (negative to positive). Subjectivity tells the intensity of a text thoughts, emotions, and expressions ranges from 0 to 1 (objective to subjective). Table 2 shows the subjective and polarity of vaccine tweets.

Table 1. Sentiment Polarity of COVID-19 vaccine tweets pos, neg, neu, compound score and its value

Text	Positive	Negative	Neutral	Compound	Value
australia manufacture vaccine give citizens	0.231	0.000	0.769	0.5106	Positive
Coronavirus vaccine coronavaccine covid vaccine	0.420	0.000	0.580	0.4404	Positive
deaths due affected countries red rpiryani sh...	0.000	0.211	0.789	-0.1531	Negative
michellegrattan conversationedu passes leaders	0.000	0.000	1.000	0.0000	Neutral
protection could last 6 months click read mode.	0.000	0.000	1.000	0.000	Neutral

Table 2. Subjectivity and polarity of vaccine tweets

Text	Subjective	Polarity
vaccines r wonderful protecting viruses exist	1.0	1.0
whipclyburn dearth warrant lack funding inovio	1.0	0.3
realdonaldtrump gonna go worst president time	-1.0	1.0
boring life its okay tono tbe okayep16 apple boy	-1.0	1.0
best luck russia hope vaccine works russianvac	1.0	0.3

Sentiment score or polarity of tweets represented using different graphs to improve the visualization and understanding. Sentiments counts of tweets depicts people have more positive attitude towards vaccine shown in the graph below Fig. 2(a). Correlation matrix represents the relationship between all the sentiment scores and compound score of vaccine tweets. This represents the linear relationship Fig. 2(b). Distribution of sentiment score positive, negative, and neutral tweets shown in Fig. 2(c).

Further, feature extraction using (Term frequency and Inverse document frequency) TF-IDF transform the text into a vector (Bania R. K., 2020). It reduces the occurrence of words in a corpus. Feature extraction improves the efficacy of the model. Count Vectorization converts or encodes text into vector format. It counts the number of occurrence of words in text or a document (Das et al., 2018).

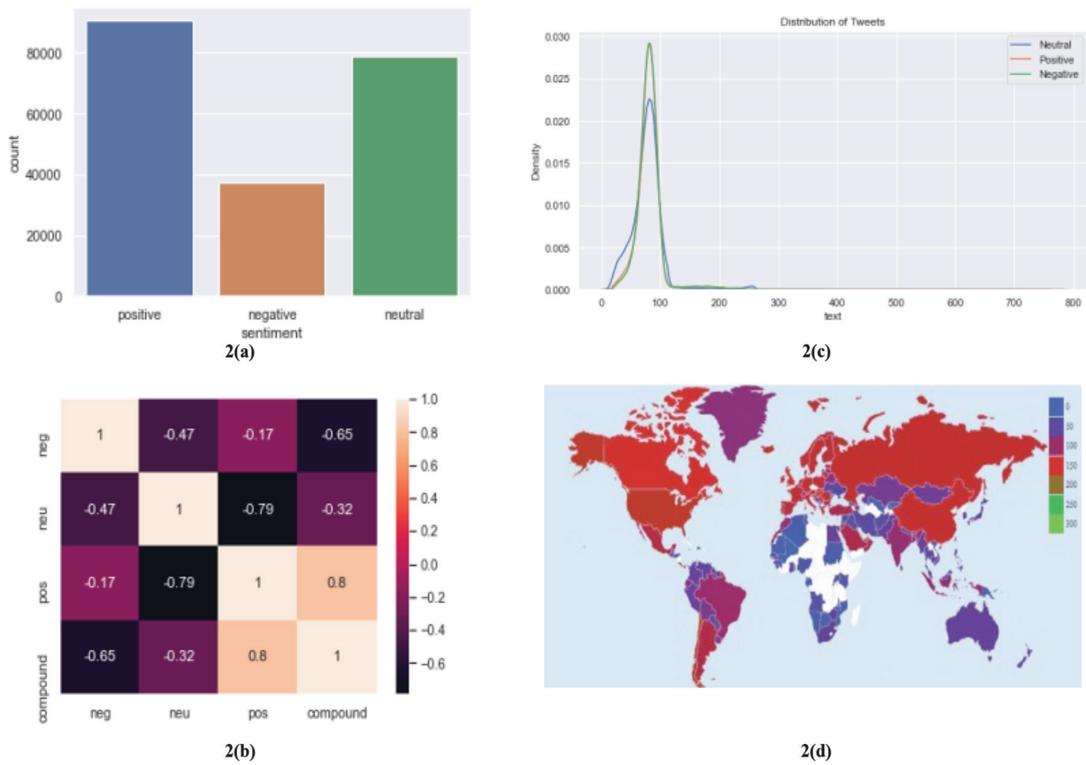


Figure 2. (a). Sentiment counts of vaccination tweets positive, negative, neutral. (b). Correlation matrix of sentiment score pos, neg, neu, and compound of tweets. (c). Sentiments distribution of tweets positive, negative, and neutral. (d). Geospatial mapping of Covid-19 vaccination doses per million of coronavirus disease globally

Source: https://ourworldindata.org/covid-vaccinations

2.3. Machine Learning Models

Building ML model for sentiment analysis after pre-processing is an important task (Khan et al., 2014). Machine Learning helps to extract insights for making accurate decision. Labelled datasets as positive, negative, or neutral fed into the model. To predict the sentiments of covid-19 vaccine tweets, Random Forest and Logistic Regression implemented to the dataset (Troussas et al., 2013) (Arya, V., & Mishra, A., 2021).

2.3.1. Random Forest

Random Forest also called "Random decision forest" ensembles learning aggregating multiple decision trees together to get more accurate and precise result. It is used for both classification and regression. It reduces over fitting as its hyper parameter return good results. Random forest takes prediction from each tree which enhances its accuracy rather than relying on one decision tree. It is a multiclass problem that performs well with both numeric and absolute features. Perform voting for each result with the highest vote as the result (Arya, V. et al., 2022).

2.3.2. Logistic Regression

LR is a probabilistic model used for binary or dichotomous classifications such as yes/no, malignant/benign, and pass/fail. Logistic function or sigmoid function used to map values between 0 and 1 provides a constant output. It is sub divided into three parts: binomial, multinomial or ordinal. In binomial LR, prediction based on dependent variable as 0 and 1. Multinomial LR based on multiclass prediction having more than one outcome, and lastly ordinal LR outcome or prediction is not in order.

$$\text{Sigmoid function (x)} = \frac{1}{1+e^{-n}} \tag{2}$$

3. Experiments and Results

The lexicon methods implemented using NLTK library in python, and machine learning classification models implemented using Scikit-learn library in python (M. Rathi et al., 2018).

VADER lexicon and Text Blob tools used for classifying sentiment of a text as positive, negative, neutral, compound, subjectivity, and polarity. Once installing vadersentiment call the object SentimentIntensityAnalyser(), then polarity obtained by using polarity_scores() as positive, negative, neutral, or compound. Compound polarity or score indicates the aggregate of all the polarities range from -1 to +1. If the compound score >0.5 assigns "pos", compound< 0.5 assigns "neg", and compound = 0 assigns "neu". Fig. 3 shows graphical representation of sentiment polarity of tweets pos, neg, neu, and compound. TextBlob is an NLP library used in python for processing text. Text Blob analyzes the polarity and subjectivity of a text which indicates the positive, negative, and intensity. Polarity assigns the text as positive and negative ranges from 1 to -1. Subjectivity assigns the text as subjective and objective ranges from 1 to 0. Fig.4 graph shows the subjectivity and polarity of tweets.

The machine learning classification models split dataset for train_test_validation in ratio 80:20 implies 80% retained for training and 20% for testing (Gyeongcheol Cho et al, 2019). Evaluation measures used for the classification models are precision, recall, f1-score, and accuracy. For multiclass classification, it categorizes into three: positive, negative, and neutral. True positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) are evaluation measures used for building models. The confusion matrix can be map for both binary classification and multi-class classification. Table 3 illustrates the confusion matrix of Random Forest, Logistic Regression, and VADER lexicon for pos, neg, and neu their precision, recall, F1-score, accuracy, macro average and weighted average. For qualitative evaluation of all measures as macro-average and weighted average precision, recall, and f1-score measured used to evaluate train models for multi-class classification. The confusion matrix of machine learning classification model Random Forest and Logistic regression shown in Fig. 5.

Macro average – It is the average of precision and recall of all classes. In this equation, precision noted as Pr, recall as Re and macro avg. as M.

$$\text{Macro avg. precision (MP)} = (\text{Pr of c1} + \text{Pr of c2})/2 \tag{3}$$

$$\text{Macro avg. Recall (MR)} = (\text{Re of c1} + \text{Re of c2})/2 \tag{4}$$

Macro-average F1-score –F1 score also define as "Suitability". Harmonic mean concatenates the precision and recall and state balance between both.

$$\text{F1- score} = 2 *(\text{Pr}*\text{Re})/(\text{Pr}+\text{Re}) \tag{5}$$

Weighted average- It is the aggregation of all the precisions scores. In this equation, precision denoted as Pr and weighted average as Wg.

$$\text{Weighted avg.} = \frac{(\text{TPr1} + \text{TPr2} + \text{TPr3} + \ldots\ldots + \text{Tprn})}{\text{Sum of all classes}} \tag{6}$$

Accuracy – It states the overall score of the classifier. Summation of true positive (TP) and true negative (TN) divides by sum of all evaluation measures (TP+TN+FP+FN).

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})} \tag{7}$$

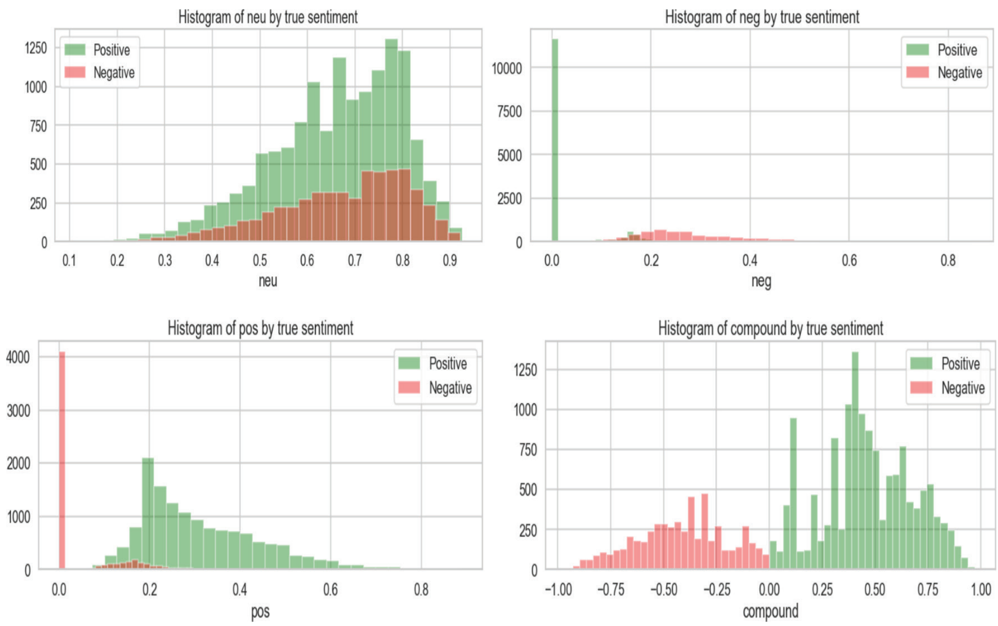


Figure 3. Sentiment polarity of tweets neutral, negative, positive, and compound of vaccine tweets

Figure 4. Subjectivity and Polarity of covid-19 vaccine tweets positive and negative of vaccine tweets

Random Forest and Logistic Regression achieved an accuracy of >90% shows supervised learning models performs better than unsupervised learning Vader Lexicon attain an accuracy of 64%. RF best suited for classification and regression. It can manage accuracy even over missing values and can handle complex and large dataset. LogisticRegression imported in python from sci-kit-learn moreover, input to the model in the two-dimensional array. Once training and testing data prepared with n_gram (1,1) transforming using countVectorizer create an instance of logistic regression. Parameters defined for the model random_state=0, max_iteration=100, n_estimator=100, keeping C=1. Multiclass classification model evaluates precision, recall, and f1-score for negative, positive, neutral, macro average, and weighted average. The model performed well obtained an accuracy of 92%.

In Random Forest, importing RandomForest from scitkit learn parameters random_state=0, n_estimator=100, test_size=0.20, bootstrap=True, keeping n_jobs=1.Removing noise, outliers and missing values enhance the performance of the model later split data into train and test. Evaluating all the measures precision, recall, f1-sore, macro avg, and weighted avg acquired an accuracy of 90%. Comparing to supervised machine learning models lexicon method performed low obtained an accuracy of 64%. Keeping, vader_polarity as compound, target as positive and negative, calculate the evaluation matrix and macro avg, and weighted avg. The VADER lexicon rule-based method performed least compared to other models' logistic regression (LR) and random forest (RF).

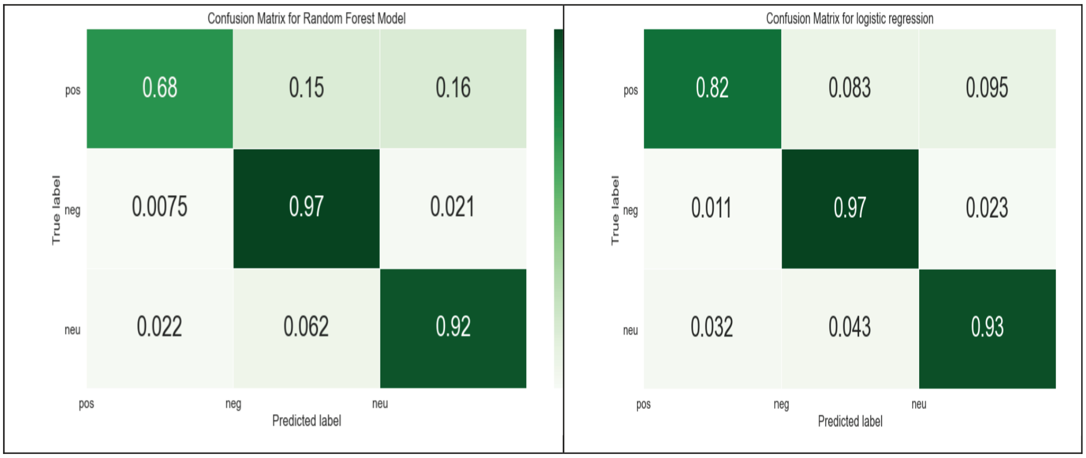


Figure 5. Confusion Matrix of Random Forest Classifier, and Logistic Regression of classification models

Table 3. Confusion Matrix of Random Forest, Logistic Regression, and VADER lexicon

Model		**Precision**	**Recall**	**F1-score**	**Accuracy**
Logistic Regression	Negative	0.89	0.82	0.85	
	Neutral	0.92	0.97	0.94	
	Positive	0.94	0.93	0.93	0.92
	Macro Avg.	0.92	0.90	0.91	
	Weight Avg.	0.92	0.92	0.92	

(continued)

Table 3. Confusion Matrix of Random Forest, Logistic Regression, and VADER lexicon (continued)

Model		**Precision**	**Recall**	**F1-score**	**Accuracy**
Random Forest	Negative	0.91	0.68	0.78	
	Neutral	0.87	0.97	0.92	
	Positive	0.92	0.92	0.92	0.90
	Macro Avg.	0.90	0.86	0.87	
	Weight Avg.	0.90	0.90	0.89	
VADER lexicon	Negative	1.00	1.00	1.00	
	Positive	0.34	1.00	0.50	0.64
	Macro Avg.	0.45	0.67	0.50	
	Weight Avg.	0.52	0.64	0.55	

4. Discussions

Most of the people globally have affected due to Covid-19 maybe it emotional, mental, social, and financial. The results obtained by the VADER lexicon method, are less than the machine learning classification models, including parameters precision, recall, and f1-score of tweets. In this analysis, Covid-19 vaccine tweets deduced from the Twitter database pre-processed with NLTK package using different Python libraries, the sentiment polarity of each text determined as pos, neg, neu, and compound score. Based on the compound score classified as positive, negative, and neutral. The stance of tweets after analysis observed to have positive over neutral and negative stance. The machine learning approach has outperformed lexicon methods Vader and TextBlob. The machine learning classification model implemented on the vectorized tweets and target. The Logistic Regression shows the highest accuracy in sentiment analysis of vaccine tweets with 92% compared to Random Forest acquired an accuracy of 90%. VADER lexicon rule-based method achieved an accuracy of 64%. RF and LR show better results on unseen tweets and classify them into three: positive, negative, and neutral. However, LR is also a good choice for text analysis.

5. Conclusion

With the proliferation of social media has become challenging to analyze the sentiments of data. This paper retrieved vaccination tweets from the Twitter database, pre-processed the tweets to remove noise and outliers, and classify them into positive, negative, or neutral. NLTK is a python library package used for the analysis of sentiments from the text. Machine learning classification models implemented to the pre-processed dataset employed Logistic Regression with an accuracy of 92%. Random Forest Classifier with TF-IDF vectorizer with an accuracy of 90% and VADER lexicon method achieved an accuracy of 64%. Logistic regression performed well over the vaccination tweets. The aim of this paper to evaluate the sentiments of the public vaccination of COVID-19 tweets, at an early phase helps the governments to make intervening policies to regulate immunization program. Evaluation of vaccination tweets provides an idea of public mindset and limiting fake news related to vaccination. This study sceptic helps the government and other organization to develop trust in immunization program and the goal of achieving herd immunity. Analyzing mental health during covid-19 impacting most of the countries, and the number of beds in hospitals worldwide. The future work, the dataset needs to be embedded in the different classification model to analyze the stance of sentiments. Another language should be incorporated while analyzing the sentiments from social media.

References

Abd El-Jawad, M. H., Hodhod, R., & Omar, Y. M. (2018, December). Sentiment analysis of social media networks using machine learning. In *2018 14th international computer engineering conference (ICENCO)* (pp. 174-176). IEEE.

Aldarwish, M. M., & Ahmad, H. F. (2017, March). Predicting depression levels using social media posts. In *2017 IEEE 13th international Symposium on Autonomous decentralized system (ISADS)* (pp. 277-280). IEEE.

Arya, V., & Mishra, A. (2021). Machine learning approaches to mental stress detection: a review. *Annals of Optimization Theory and Practice*, *4*(2), 55-67. https://doi.org/10.22121/aotp.2021.292083.1074

Arya, V., Mishra, A. K. M., & González-Briones, A. (2022). Analysis of sentiments on the onset of COVID-19 using machine learning techniques.

Baheti, R., & Kinariwala, S. (2019). Detection and analysis of stress using machine learning techniques. *Int. J. Eng. Adv. Technol*, *9*, 335-342.

Bania, R. K. (2020). COVID-19 public tweets sentiment analysis using TF-IDF and inductive learning models. *INFOCOMP Journal of Computer Science*, *19*(2), 23-41.

Bonta, V., & Janardhan, N. K. N. (2019). A comprehensive study on lexicon-based approaches for sentiment analysis. *Asian Journal of Computer Science and Technology*, *8*(S2), 1-6.

Brunier, A., & Drysdale, C. (2020). COVID-19 disrupting mental health services in most countries, WHO survey. *World Heal Organ*.

Cavazos-Rehg, P. A., Krauss, M. J., Sowles, S., Connolly, S., Rosas, C., Bharadwaj, M., & Bierut, L. J. (2016). A content analysis of depression-related tweets. *Computers in human behavior*, *54*, 351-357.

Centre for Disease control and Prevention, 2021. Variants of the virus that causes COVID-19. National Center for Immunization and Respiratory Diseases (NCIRD), Division of Viral Diseases

Chancellor, S., & De Choudhury, M. (2020). Methods in predictive techniques for mental health status on social media: a critical review. *NPJ digital medicine*, *3*(1), 1-11.

Cherian, S., Potdar, V., Jadhav, S., Yadav, P., Gupta, N., Das, M., ... & Panda, S. Convergent evolution of SARS-CoV-2 spike mutations, L452R, E484Q and P681R, in the second wave of COVID-19 in Maharashtra, India. bioRxiv 2021: 2021.04. 22.440932.

Chilwal, B., & Mishra, A. K. (2021). Extraction of Depression Symptoms From Social Networks. *The Smart Cyber Ecosystem for Sustainable Development*, 307-321.

Cho, G., Yim, J., Choi, Y., Ko, J., & Lee, S. H. (2019). Review of machine learning algorithms for diagnosing mental illness. *Psychiatry investigation*, *16*(4), 262.

Das, B., & Chakraborty, S. (2018). An improved text sentiment classification model using TF-IDF and next word negation. *arXiv preprint arXiv:1806.06407*.

Desai, M., & Mehta, M. A. (2016, April). Techniques for sentiment analysis of Twitter data: A comprehensive survey. In *2016 International Conference on Computing, Communication and Automation (ICCCA)* (pp. 149-154). IEEE.

Deshpande, M., & Rao, V. (2017, December). Depression detection using emotion artificial intelligence. In *2017 international conference on intelligent sustainable systems (iciss)* (pp. 858-862). IEEE.

Dubey, A. D. (2020). Twitter sentiment analysis during COVID-19 outbreak. *Available at SSRN 3572023*.

Gbashi, S., Adebo, O. A., Doorsamy, W., & Njobeh, P. B. (2021). Systematic delineation of media polarity on COVID-19 vaccines in Africa: computational linguistic modeling study. *JMIR medical informatics*, *9*(3), e22916.

Khan, H., Srivastav, A., & Mishra, A. K. (2020). Use of classification algorithms in health care. In *Big Data Analytics and Intelligence: A Perspective for Health Care*. Emerald Publishing Limited.

Khan, M., Rizvi, Z., Shaikh, M. Z., Kazmi, W., & Shaikh, A. (2014). Design and implementation of intelligent human stress monitoring system. *International Journal of Innovation and Scientific Research*, *10*(1), 179-190.

Nguyen, T., Phung, D., Dao, B., Venkatesh, S., & Berk, M. (2014). Affective and content analysis of online depression communities. *IEEE Transactions on Affective Computing*, *5*(3), 217-226.

Raichur, N., Lonakadi, N., & Mural, P. (2017). Detection of stress using image processing and machine learning techniques. *International journal of engineering and technology*, *9*(3), 1-8.

Ritchie, H., Mathieu, E., Rodés-Guirao, L., Appel, C., Giattino, C., Ortiz-Ospina, E., ... & Roser, M. (2020). Coronavirus pandemic (COVID-19). *Our world in data*.

Rana, M., Rehman, M. Z. U., & Jain, S. (2022, February). Comparative Study of Supervised Machine Learning Methods for Prediction of Heart Disease. In *2022 IEEE VLSI Device Circuit and System (VLSI DCS)* (pp. 295-299). IEEE.

Rathi, M., Malik, A., Varshney, D., Sharma, R., & Mendiratta, S. (2018, August). Sentiment analysis of tweets using machine learning approach. In *2018 Eleventh international conference on contemporary computing (IC3)* (pp. 1-3). IEEE.

Saladino, V., Algeri, D., & Auriemma, V. (2020). The psychological and social impact of Covid-19: new perspectives of well-being. *Frontiers in psychology*, 2550.

Shatte, A. B., Hutchinson, D. M., & Teague, S. J. (2019). Machine learning in mental health: a scoping review of methods and applications. *Psychological medicine*, *49*(9), 1426-1448.

Subhani, A. R., Mumtaz, W., Saad, M. N. B. M., Kamel, N., & Malik, A. S. (2017). Machine learning framework for the detection of mental stress at multiple levels. *IEEE Access*, *5*, 13545-13556.

Terry, M. (2021). UPDATED comparing COVID-19 vaccines: timelines, types, and prices. *BioSpace. April 23*.

Troussas, C., Virvou, M., Espinosa, K. J., Llaguno, K., & Caro, J. (2013, July). Sentiment analysis of Facebook statuses using Naive Bayes classifier for language learning. In *IISA 2013* (pp. 1-6). IEEE.

World Economic Forum, 2020. https://www.weforum.org/agenda/2020/08/covid-19-coronavirus-mental-health-well-being-countries/

Yazdavar, A. H., Mahdavinejad, M. S., Bajaj, G., Thirunarayan, K., Pathak, J., & Sheth, A. (2018, June). Mental health analysis via social media data. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)* (pp. 459-460). IEEE.

ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
eISSN: 2255-2863 - DOI: https://doi.org/10.14201/ADCAIJ2022114 - CDU: 004 -
IBIC: Computación e informática (U) - BIC: Computing & Information Technology (U) -
BISAC: Computers / General (COM000000)

Regular Issue, Vol. 11, N. 4 (2022)

INDEX

Ediciones Universidad Salamanca
RESEARCH GROUP BISITE .usal.es