ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

EMPLOYING DIGITAL TWIN TO LORA BASED FOREST FIRE MANAGEMENT SYSTEMS

M.Sc. THESIS

Buğra AYDIN

Department of Computer Engineering

Computer Engineering Programme

APRIL 2025

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

EMPLOYING DIGITAL TWIN TO LORA BASED FOREST FIRE MANAGEMENT SYSTEMS

M.Sc. THESIS

Buğra AYDIN
(504221505)

Department of Computer Engineering

Computer Engineering Programme

Thesis Advisor: Prof. Dr. Sema Fatma Oktuğ

APRIL 2025

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

LORA'YA DAYALI ORMAN YANGINI YÖNETİM SİSTEMLERİNE DİJİTAL İKİZ UYGULANMASI

YÜKSEK LİSANS TEZİ

Buğra AYDIN
(504221505)

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Sema Fatma Oktuğ

NİSAN 2025

Buğra AYDIN, a M.Sc. student of ITU Graduate School student ID 504221505 successfully defended the thesis entitled "EMPLOYING DIGITAL TWIN TO LORA BASED FOREST FIRE MANAGEMENT SYSTEMS", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Sema Fatma Oktuğ** Istanbul Technical University	.....
Jury Members :	**Prof. Dr. Fatih Alagöz** Boğaziçi University	.....
	Prof. Dr. Abdül Halim Zaim Istanbul Technical University	.....

Date of Submission : **11 April 2025**
Date of Defense : **21 April 2025**

To my family,

FOREWORD

I would like to express my sincere gratitude to my supervisor, Professor Dr. Sema Fatma OKTUĞ, for her invaluable support and guidance throughout the process of writing my M.Sc. thesis. I extend my thanks to the members of the jury, Prof. Dr. Fatih Alagöz and Prof. Dr. Abdül Halim Zaim, for their time, diligence, and thoughtful evaluation of this manuscript. I am also very grateful to Assoc. Prof. Dr. Yusuf Yaslan for the insightful discussions regarding the project, which proved instrumental in refining the core ideas explored in this work. I would also like to acknowledge the unwavering support of my family, whose continuous encouragement and assistance have been instrumental in my academic journey.

APRIL 2025 Buğra AYDIN

TABLE OF CONTENTS

ABBREVIATIONS

AI : Artificial Intelligence
ANN : Artificial Neural Network
CH : Cluster Head
CNN : Convolutional Neural Network
CSS : Chirp Spread Spectrum
CSV : Comma Seperated Values
DRL : Deep Reinforcement Learning
DT : Digital Twin
DTN : Digital Twin Network
FL : Federated Learning
GAT : Graph Attention Network
GLN : Graph Linformer Network
GNN : Graph Neural Network
GPRS : General Packet Radio Service
GRU : Gated Recurrent Unit
IoT : Internet of Things
LNS : LoRa Network Server
LPWAN : Low Powered Wide Area Network
LSTM : Long Short-Term Memory
MAC : Medium Access Control
MAPE : Mean Absolute Percentage Error
MLP : Multi Layer Perceptron
MSE : Mean Squeared Error
NB-IoT : Narrow Band Internet of Things
RNN : Recurrent Neural Network
SF : Spreading Factor
SLA : Service Level Agreement
SMOTE : Synthetic Minority Oversampling Technique
SVR : Support Vector Regressor
UAV : Unmanned Aerial Vehicle
Wi-Fi : Wireless Fidelity
WSN : Wireless Sensor Network

SYMBOLS

CO **:** Carbon Monoxide

LIST OF TABLES

LIST OF FIGURES

Page

EMPLOYING DIGITAL TWIN TO LORA BASED FOREST FIRE MANAGEMENT SYSTEMS

SUMMARY

Forest fires are one of the most critical risks threatening ecology. With the increasing average temperature of the world, the number and severity of forest fires have also increased. This has made early detection of forest fires essential. With developing technology, forest fires can be detected with the help of different techniques. Among these techniques, fire detection using sensor networks offers the most effective solutions in terms of both speed and cost.

Thanks to the Internet of Things, wireless sensor networks can be established that regularly monitor an environment and collect this data through a large number of affordable internet-connected devices. It is also possible to establish a network with sensor nodes that measure temperature, humidity, etc. for the detection of forest fires. However, these networks have different requirements for each forest. Therefore, managing the networks can be complicated.

Digital Twin technology is an innovative simulation alternative that facilitates the work to be done on this system by creating a real-time model of the physical system. It can be used in the optimization of systems, testing of different scenarios, and observation of the system. It can also provide significant benefits for the management of networks created for forest fires. However, it is quite difficult for such networks to provide the continuous and two-way communication demand of digital twins.

Forest conditions are also effective in determining the communication technologies of the wireless network to be designed. Short-range and high-power radio modulation technologies are generally not preferred because of the large areas covered by forests and the fact that the geographical location of the forests makes it inconvenient to frequently maintain the devices in the network. Communication technologies developed for low-power wide area networks are ideal communication techniques for forest fire detection management systems.

LoRa radio frequency modulation technology is one of the most widely used low-power wide-area network technologies. Nodes with LoRa modules are preferred in such networks due to their advantages of being able to transmit messages over distances of kilometers and providing years of battery life in battery-powered sensor devices. In addition, allowing unlicensed use and operating in unlicensed frequency ranges are also important reasons why it is often chosen in these networks.

Detection of forest fires using computer networks is a subject that has been studied extensively in the literature. In previous studies, sensors such as temperature, humidity, carbon monoxide, etc. were used for fire detection. Although long-distance technologies such as LoRa are preferred, short-distance data transmission techniques

such as Bluetooth and ZigBee have also been used. The topologies of the networks have been created with various techniques such as mesh, clustered, tree, and star. In addition, studies have been conducted using artificial intelligence to detect false alarms from sensors.

Digital twins have the ability to quickly adapt to changes in the system because they create a real-time model of the physical system. They can be used with optimization algorithms to take the necessary actions for the changing conditions of the system. This technique can be groundbreaking for the optimization of dynamic systems such as computer networks. However, some common artificial intelligence techniques used in the development of twins have difficulty making sense of the relational structure in these networks where neighborhoods are important.

For this reason, special digital twin technologies developed for computer networks are also called digital twin networks. These twins usually use graph neural networks to understand the relational structure of networks. Graph neural networks are a special data learning technique used to analyze data represented by graphs, such as protein structures, social networks, etc. Although it varies from model to model, the basic working principle of the technique is based on sharing the properties held in nodes and edges with neighboring nodes over several iterations to update their values.

Graph neural networks are used in computer network studies in the literature, in routing optimization, network slicing management, and real-time modeling of some metrics of the network. In these studies, while transmitting messages to the twin network models and implementing new configurations, classical optimization techniques work together with deep reinforcement learning techniques for optimization. It has also been studied to use federated learning techniques for training digital twin models.

This thesis study proposes adding a forecaster between the network and the twin to facilitate the application of digital twins to the Internet of Things networks. The problem that arises from the inconvenience of providing continuous data flow from these networks and the need for this flow, is overcome by the estimations it makes after the forecaster is trained with the data coming from the network.

In order to train the digital twin model, data from situations that prevent data flow such as many collisions and interference is needed. Due to the difficulty of collecting this data from physical networks and the fact that forest fire detection networks consist of hundreds of nodes and cover square kilometers of area, the network was simulated with a custom simulator developed in this study. A clustered network topology was designed, LoRa technology was used for intra-cluster communication, and GPRS technology was used for inter-cluster communication. At the end of the simulation, all packets formed in the simulation were reported.

The forecaster is a binary sequence prediction model that estimates whether a node sent a packet or not for each node and each time interval using the packets formed in the simulation. Since sensor nodes sleep most of the time, the undersampling technique was used to balance the dataset. Then, using these estimates, the overall state of the network was extracted for each time interval to be given to the digital twin.

Using a graph neural network-based digital twin, the total output of the network was estimated using the network state estimates of the forecaster. Using simulation data, the actual output values were compared with the twin's estimates.

The results are generally promising. Especially for small-scale networks, the system's R^2 performance is above 97 percent. The MSE values are well below 1. However, as the network scale increases, the system's performance decreases significantly. This is thought to be due to the cumulative accumulation of errors made by the forecaster in the estimation of each node. The system has also been tested with different digital twin and forecaster AI algorithms. Although the system performs similarly or slightly better with these models, it has been observed that this increase will not solve the scalability problem.

As a result, integrating digital twins with the Internet of Things is impractical due to the difficulty of meeting the demands of the twins. This study aims to facilitate this integration by proposing a forecaster model. For this purpose, a simulation was designed and the performance of the system was tested. According to the results, it was concluded that this method is suitable, especially for small-scale networks. However, it has also been shown that the system has a scalability problem. In future work, a forecaster module can be designed to make a more holistic prediction of the network, the simulation parameters can be improved to expand the environmental and traffic throughput, and the digital twin can be made to predict the arrival percentages based on the importance of incoming packets instead of the network throughput.

LORA'YA DAYALI ORMAN YANGINI YÖNETİM SİSTEMLERİNE DİJİTAL İKİZ UYGULANMASI

ÖZET

Orman yangınları, ekolojiyi tehdit eden en kritik risklerden bir tanesidir. Dünyanın artan ortalama sıcaklığı ile birlikte, orman yangınlarının sayısı ve şiddeti de artmıştır. Bu durum da orman yangınlarının erken tespitini elzem hale getirmiştir. Gelişen teknoloji ile birlikte farklı teknikler yardımıyla orman yangınlarının tespiti yapılabilmektedir. Bu teknikler arasından, sensör ağları kullanılarak yangın tespitinde bulunmak hem hız hem maliyet açısından en etkin çözümleri sunmaktadır.

Nesnelerin İnterneti sayesinde, erişilebilir maliyetli çok sayıdaki internete bağlı cihaz aracılığıyla, bir ortamın düzenli olarak gözlemlenmesi ve bu verilerin toplanmasını sağlayan kablosuz sensör ağları kurulabilmektedir. Orman yangınlarının tespiti için de sıcaklık, nem vb. değerleri ölçen sensör düğümleri ile bir ağ kurmak mümkündür. Fakat bu ağların, her orman için farklı isterleri olmaktadır. Bu nedenle de ağların yönetilmesi karmaşık olabilmektedir.

Dijital İkiz teknolojisi, fiziksel sistemin gerçek zamanlı modelini oluşturarak bu sistem üzerinde yapılacak çalışmaları kolaylaştıran yenilikçi bir simülasyon alternatifidir. Sistemlerin optimizasyonunda, farklı senaryoların testinde ve sistemin gözlemlenmesinde kullanılabilir. Orman yangını için oluşturulan ağların yönetimi için de önemli faydalar sağlayabilir. Fakat, dijital ikizlerin sürekli ve iki yönlü iletişim talebinin bu tarz ağlar tarafından sağlanması oldukça zahmetlidir.

Ormanların koşulları, tasarlanacak kablosuz ağın iletişim teknolojilerinin belirlenmesinde de etkilidir. Kısa mesafeli ve yüksek güç tüketimine sahip radyo modülasyon teknolojileri, ormanların kapladıkları geniş alanları ve ormanların coğrafi konumlarının ağdaki cihazların bakımlarının sıklıkla yapılmasını elverişsiz kılması nedeniyle genellikle tercih edilmemektedir. Düşük güçlü geniş alan ağları için geliştirilen iletişim teknolojileri orman yangını tespit yönetim sistemleri için biçilmiş iletişim tekniklerdir.

LoRa radyo frekansı modülasyon teknolojisi, en yaygın kullanılan düşük güçlü geniş alan ağı teknolojilerinden biridir. Kilometrelerce uzunlukta mesafelerde mesaj iletimi yapabilmesi ve batarya ile çalışan sönsör cihazlarında yıllarca pil ömrü sunabilmesi avantajlarından kaynaklı olarak bu tür ağlarda LoRa modülüne sahip düğümler tercih edilir. Ayrıca, lisanssız kullanıma izin vermesi ve yine lisanssız frekans aralıklarında çalışması da sıklıkla bu ağlarda seçilmesinin önemli nedenlerindendir.

Orman yangınlarının bilgisayar ağları kullanılarak tespiti literatürde yoğun çalışmılmış bir konudur. Daha önceki çalışmalarda yangın tespiti için sıcakık, nem, karbon monoksit vb. sensörler kullanılmıştır. LoRa gibi uzun mesafe teknolojileri tercih edilse

de Bluetooth, ZigBee gibi kısa mesafeli veri iletim teknikleri de kullanılmıştır. Ağların topolojileri örgü, kümelenmiş, ağaç ve yıldız gibi çeşitli tekniklerle oluşturulmuştur. Ayrıca sensörlerden gelen yanlış alarmların tespiti için de yapay zekadan yararlanılan çalışmalar da yürütülmüştür.

Dijital ikizler, fiziksel sistemin gerçek zamanlı modelini oluşturduğundan sistemdeki oluşan değişimlere hızlı adapte olma yeteneğine sahiptirler. Optimizasyon algoritmaları ile birlikte kullanılarak sistemin değişen şartları için gerekli aksiyonları alabilirler. Bilgisayar ağları gibi dinamik sistemlerin optimizasyonu için bu teknik çığır açıcı olabilir. Fakat ikizlerin geliştirilmesinde kullanılan bazı yaygın yapay zeka teknikleri, bu ağlardaki komşuluğun önemli olduğu ilişkisel yapıyı anlamlandırmakta zorlanmaktadır.

Bu nedenle bilgisayar ağları için geliştirilen özel dijital ikiz teknolojileri dijital ikiz ağları adıyla da anılmaktadır. Bu ikizler, ağların ilişkisel yapısını anlamak için genellikle grafik sinir ağlarını kullanır. Grafik sinir ağları, protein yapıları, sosyal ağlar vb. gibi grafiklerle temsil edilen verileri analiz etmek için kullanılan özel bir veri öğrenme tekniğidir. Modelden modele değişmekle birlikte, tekniğin temel çalışma prensibi, düğümlerde ve kenarlarda tutulan özelliklerin değerlerini güncellemek için birkaç yineleme boyunca komşu düğümlerle paylaşılmasına dayanır.

Grafik sinir ağları, literatürdeki bilgisayar ağları çalışmalarında, rotalama optimizasyonunda, ağ dilimleme yönetiminde ve ağın bazı değerlerinin gerçek zamanlı modellenmesinde kullanılır. Bu çalışmalarda ikiz ağı modeller ve yeni konfigürasyonların uygulanması mesajlarını iletirken optimizasyon için, klasik optimizasyon teknikleri, derin takviyeli öğrenme teknikleri ile birlikte çalışır. Ayrıca dijital ikiz modellerin eğitimi için birleşik öğrenme tekniğinin kullanılması da çalışılmıştır.

Bu tez çalışması ise dijital ikizlerin nesnelerin interneti ağlarına uygulanmasını kolaylaştırmak amacıyla ağ ile ikiz arasına ağın durumunu tahmin eden bir tahminci eklenmesini önermektedir. Bu ağlardan sürekli veri akışının sağlanmasının elverişiz olması ve dijital ikizlerin bu akışa ihtiyaç duymasından kaynaklı oluşan sorunu, tahminci ağdan gelen verilerle eğitildikten sonra yaptığı tahminlerle kapatır.

Dijital ikiz modelinin eğitilmesi için birçok çarpışma, parazit gibi veri akışını engelleyen durumların verisine ihtiyaç duyulmaktadır. Hem bu verilerin fiziksel ağlardan toplanmasının zor olması, hem de orman yangını tespit ağlarının yüzlerce düğümden ve kilometre karelerce alanı kaplamasından kaynaklı olarak bu çalışmada ağ geliştirilen özel simülatörle simüle edilmiştir. Kümelenmiş bir ağ topoljisi tasarlanmış, küme içi iletişim için LoRa, kümeler arası iletişim için GPRS teknolojisi kullanılmıştır. Simülasyonun sonunda, simülasyonda oluşan tüm paketler raporlanmıştır.

Tahminci, simülasyonda oluşan paketleri kullanarak, her düğüm ve her zaman aralığı için düğümün paket gönderip göndermediğini tahmin eden bir ikili sekans tahmin modelidir. Sönsör düğümleri zamanın büyük bir bölümünde uyuduğundan veri setini dengeli hale getirmek için alt örnekleme tekniğinden yararlanılmıştır. Daha sonra bu tahminler kullanılarak dijital ikize vermek üzere, her zaman aralığı için ağın genel durumu çıkartılmıştır.

Grafik sinir ağı tabanlı bir dijital ikiz ile tahmincinin ağ durumu tahminleri kullanılarak ağın toplam çıktısı tahmin edilmiştir. Simülasyon verileri kullanılarak gerçek çıktı değerleri ile ikizin tahminleri karşılaştırılmıştır.

Sonuçlar genel olarak umut vericidir. Özellikle küçük ölçekli ağlar için, sistemin R^2 performansı yüzde 97'nin üzerindedir. MSE değerleri ise 1'in oldukça altındadır. Ancak, ağın ölçeği büyüdükçe sistemin performansı kayda değer ölçüde azalmaktadır. Bunun, tahmincinin her düğümün tahmininde yaptığı hataların kümülatif bir şekilde toplanmasından kaynaklandığı düşünülmektedir. Ayrıca sistem farklı dijital ikiz ve tahminci yapay zeka algoritmaları ile de test edilmiştir. Sistem bu modellerle benzer veya biraz daha iyi performans gösterse de bu artışın ölçeklenebilirlik sorununu çözmeyeceği gözlenmiştir.

Sonuç olarak, dijital ikizlerin nesnelerin interneti ile entegre edilmesi, ikizlerin taleplerinin karşılanmasının zorluğundan dolayı elverişsizdir. Bu çalışmada bir tahminci modeli önerilerek bu entegrasyonun kolaylaştırılması hedeflenmiştir. Bunun için simülasyon tasarlanmış ve sistemin performansı test edilmişitir. Ortaya çıkan sonuçlara göre, özellikle küçük ölçekli ağlar için bu yöntemin uygun olduğu çıkarılmıştır. Fakat sistemin bir ölçeklenebilirlik problemi olduğu da gösterilmiştir.

Gelecekteki çalışmalarda, ağın daha bütüncül bir tahminini yapacak tahminci modülü tasarlanabilir, simülasyon parametreleri çevresel ve trafik üretimini genişletecek şekilde geliştirilebilir ve dijital ikizin ağın çıktısı yerine gelen paketlerin önemine göre varış yüzdelerini tahmin etmesi sağlanabilir.

1. INTRODUCTION

Forest fires are one of the most dangerous threats to the world's ecology. Rising average temperatures have also increased the frequency of forest fires. As in the rest of the world, the number of forest fires has increased in Turkey. More than 10 thousand hectares of forest area are destroyed in more than 2 thousand forest fires every year as in the graph in Figure 1.1 [1]. Although forest fires pose a threat to all forested areas, forests in some regions are at greater risk of fire due to climatic conditions. Forests in the Mediterranean and Aegean regions, especially in Muğla, Antalya, İzmir, and Adana, are frequently burned. As mentioned, the world's rising average temperature increases the risk of forest fires, which also increases the importance of forest fire management systems. Early and rapid-fire detection can significantly reduce the devastation of forest fires. Several different techniques are used for these detections. The most common methods include surveillance of forests with satellites, flying over forests with Unmanned Aerial Vehicles (UAVs), and regularly monitoring forest values such as temperature and humidity with wireless sensor networks. All of these systems have their advantages and disadvantages. However, the most cost-effective and fastest fire detection method is Wireless Sensor Network (WSN) solutions. [2].

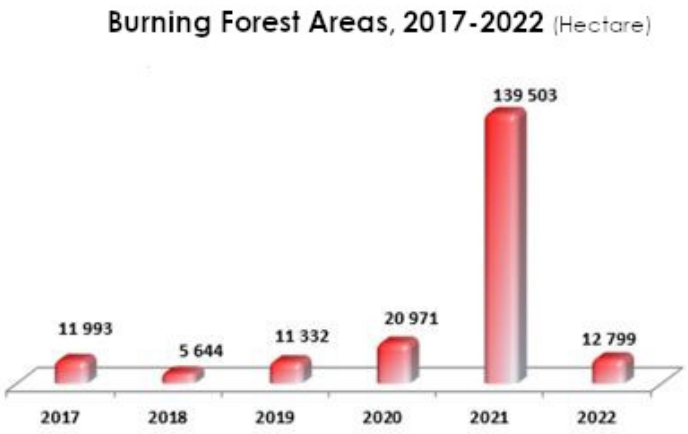

Figure 1.1 : Area of burned on every year [1].

Internet of Things (IoT) sensor networks are frequently used in these systems to detect forest fires early. However, forests differ from each other for many reasons, such as size, elevation difference, climatic conditions, tree density, and diversity. Therefore, the demands of forests and the structures of these networks vary widely. This requires specific decisions to be made in the management of each network. Also, due to the size of forests, managing these large networks can be difficult.

IoT devices are important in data collection. These electronic devices are generally affordable, internet-connected computers with modest computation capabilities. They can also use other communication protocols than the internet according to the needs of the problem and resource restrictions. Environments like forests require long-range and power-efficient communication technologies because, otherwise the networks would be difficult to maintain due to their scale. Therefore, Low-Powered Wide-Area Networks (LPWANs) are highly suitable and prevalent in such situations. LoRa is one of the most used and enhanced LPWAN technologies. IoT devices can send messages above a kilometer away for years with LoRa modules.

Once WSN networks are installed in forests, management is another problem to solve. In these sophisticated networks, it may be desirable to minimize packet loss or optimize energy consumption. Digital Twin (DT) technology can help network administrators in this area. DTs are widely used in computer networks for optimization and running test cases. Nonetheless, DTs are challenging to use in IoT networks due to their constant communication requirements. Therefore, this paper proposes a forecaster mechanism to facilitate this integration. The proposed model generates the data needed by DT by trying to predict the network's packets in advance.

In this thesis, in order to realize the applicability of digital twins in IoT networks, a digital twin application for forest fire detection IoT networks is studied. The integration of the digital twin into these networks is key to perceiving their complexity. It also enables effective and accurate testing of different strategies for network optimization. However, the need for continuous bidirectional data transfer of digital twins poses the main challenge in making this integration. Since IoT networks have limitations in terms of energy and performance, meeting these requirements is challenging for them.

In addition, some use cases can present interesting contradictions. For example, a digital twin modeling a network's packet delivery rate will need real-time lost packet data from the network. But predictably, it is impossible for the digital twin to be aware of a packet that has not reached the network's server. In networks such as the one studied in this thesis, where forest fires are specifically investigated, LPWAN communication technologies are widely preferred. The additional limitations of these technologies, such as two-way communication, can make the integration of DTs even more intractable.

Instead of providing real-time network data to the DT, designing a forecaster that predicts the future packets that the network will generate and providing the DT with the predictions it generates from historical network data can alleviate these problems and enable the integration of DTs into IoT networks. Based on this hypothesis, the aim of this thesis is to accurately determine the throughput value of the network in the simulation of the designed forest fire detection network with the help of a forecaster that detects the packets generated by this network and a DT that estimates the throughput of the network with the forecaster's output. Due to the high physical and hardware demands of the network, such as square kilometers of coverage and dozens of sensor devices, this study was conducted by designing the network in simulation. Also, since the models require lost packet data, the implementation of this work for real networks requires running the network in a simulator. Since a simulation was already used in the study, no extra simulation was performed. First, the simulation was run and the generated packets were obtained. Then, forecaster models were trained with the packets generated at the beginning of the simulation to generate forecasts for later packets. Then, these forecasts were forwarded to DT, and the throughput of the network over time was estimated by the DT model. To evaluate the performance of the system, the actual throughput values obtained from the simulation are compared with the predictions. The throughput of the network is affected by the packets generated and the packet losses in the network. The system needs to understand both of these components accurately in order to make successful predictions. Studies in the literature have shown that successful predictions can be made with techniques such as Recurrent Neural Network (RNN) based models in sequence data. In addition, Graph

Neural Network (GNN) based DT models in the research can successfully comprehend situations such as traffic and packet loss in networks. For this reason, it is thought that the results that will emerge with the cooperation of these models can be successful.

The results obtained in this study show that this system works promisingly, especially for small-scale networks. However, as the number of devices in the network increases, the system's performance decreases critically. This result is assumed to be due to the forecasting error of the whole network, which increases cumulatively with the increasing number of devices.

In this thesis, the study aims to apply DTs to IoT networks and calculate the throughput of the network with the proposed method using the data obtained from the simulation of the forest fire management system IoT network as an example scenario. In the remaining chapters of the thesis, we first review the literature on the detection and prevention of forest fires from a network perspective. Chapter 3 summarizes DT technology. Then, the methodology of the study is described in Our Work section and conclusions and evaluations are given in chapter 5. The last section concludes and discusses future work.

2. FOREST FIRE DETECTION AND PREVENTION: FROM NETWORK POINT OF VIEW

Detection and prevention of forest fires is a frequently studied topic in the literature. More than 250 studies have been conducted in this field in the last decade [3]. Within the scope of this study, a curated selection of papers from the literature on the frontiers of computer networks is reviewed in this section.

2.1 IoT

IoT targets a network of many small devices with specific purposes, communicating via the internet. Some of these devices provide data for decision-making processes by continuously measuring a physical phenomenon in their environment, while others can perform actions to put decisions into practice. Some use cases, such as forest fire detection, which is the purpose of this study, may only require monitoring. WSN solutions can be used for these applications. WSN is defined as a network of nodes that function collaboratively in order to perceive the environment and control the space that surrounds them. These nodes are interconnected by means of wireless media, facilitating communication between themselves [4]. Thanks to their modest equipment and constant monitoring, they make it possible to make and implement more precise and optimal decisions at very affordable costs. However, these features also present specific challenges such as security, scalability, maintenance, energy, and data reliability. It can be difficult to know whether the data obtained from the network is accurate, whether it came from the device in question, and how much of the measured data has arrived. Furthermore, it can be operationally labor-intensive to keep a large number of devices with limited batteries running continuously.

One of the most important decisions in the design of IoT networks is choosing the method of communication. Different methods may be more appropriate for different networks. The communication distance and power consumption requirements of the devices in the network can play a decisive role in communication technology, although

there are many other criteria. In situations where short-range communication is sufficient and high power consumption can be tolerated, technologies with high bit rates such as Bluetooth and Wireless Fidelity (Wi-Fi) can be used. However, if power consumption is an important limit, low-power technologies such as ZigBee may be preferred. These short-distance communication technologies can be used up to 100 meters, but long-distance communication technologies are needed for longer distances. In these cases, cellular connectivity can be used if power consumption can be tolerated. Yet, when both low power consumption and long distance are essential, the previously mentioned LPWAN technologies offer a solution. LPWAN technologies can send data at low speeds but over distances of many kilometers with low power consumption. Although they are not suitable for every application due to their low data transfer capacity, they can be used in applications such as smart garbage container management and smart street lights. Different technologies such as SigFox, LoRa, and Narrow Band Internet of Things (NB-IoT) are in this class. In this study, LoRa is simulated due to its popularity and license-free availability.

2.1.1 LoRa

LoRa is a radio frequency modulation technology for long-distance data transmission based on Chirp Spread Spectrum (CSS) technology [5]. It can communicate with 6 different spreading factors (SF) between 7 and 12. Since these SFs are orthogonal to each other, packets sent with different SFs do not collide with each other to a large extent. While low SFs can send more data over a relatively shorter distance, the opposite is true for high SFs. LoRa end devices are also divided into three classes, A, B, and C. Classes determine downlink ranges. Class A can only receive data for 2 specific intervals after sending a packet, while Class C devices have an open data reception period as long as they are not sending packets. Class B devices synchronize with the beacon from the gateway and listen for incoming packets at certain intervals.

To connect these devices to the internet, a gateway device, that has both internet and LoRa modules, is used. LoRa operates in the appropriate Sub-GHz ISM band, although this varies from region to region. Therefore the devices, gateways, and end devices, are subject to some duty cycle limits. The most preferred system at the Medium

Access Control (MAC) layers of LoRa networks is LoRaWAN. In LoRaWAN, end devices are directly connected to the gateway device as in star topology. Also, multiple gateways are supported in LoRaWAN networks. In this case, each gateway has its connected nodes creating multiple star topologies named star-of-stars topology. There are no strict assignments of nodes to gateway devices. The same packet of the node can be forwarded to the LoRa Network Server (LNS), which is the server where all gateways of the network connect, by multiple gateways. Although LoRaWAN is the most popular topology option for LoRa networks, its simple ALOHA-like structure can make it inefficient in crowded networks. Therefore, other topologies like mesh and clustered can also be constructed using LoRa modules and gateway devices. LoRaWAN is claimed to scale hundreds of end devices; nonetheless, in real-life applications, the packet delivery rate of the network rapidly decreases to unsatisfactory ratios with the increasing number of sensors. Some studies have shown that mesh topology scales better than LoRaWAN in terms of packet delivery ratio [6]. Thus, mesh and clustered networks are better options for crowded topologies.

2.2 Exemplary Forest Fire Detection Systems

Early detection of fires is critical to reducing their impact. However, detecting fires in minutes when they originate in vast forest areas is arduous. Satellites, UAVs, and sensors are utilized in forests to meet this need. Since it is not feasible to position satellites to continuously monitor the forest, and as UAVs have limited observation areas and need to be recharged for some time, IoT networks have the fastest fire detection capability of these methods.

IoT networks deployed for forest fire detection can be designed in many different ways. First, it is decided how to detect the fire. While fire detection can be done with affordable sensors such as temperature, CO, and humidity, it can also be done with the help of cameras.

In [7], data such as flame, humidity, and temperature were measured and if the designed algorithm detected a fire situation, alarm packets were sent from the nodes to the database server with location information using satellite communication via SAT-202 module.

Next, the topology of the network and communication technologies are decided. Clustered, and mesh topologies, such as star topology, are often preferred depending on the frequency of data sent and the network's scale. While fault tolerance is higher in mesh networks, clustered topologies provide scalability with a simpler network structure. In [8], a hierarchical network structure is designed. Two different types of nodes were preferred: central nodes and sensor nodes. Sensor nodes are connected to each other in a tree structure, with the root node being the central node. While Zigbee communication is provided in the sensor nodes, the central node also carries the cellular network module to send the data to the server. In another study [9], mesh topology is chosen as a network structure. While all nodes are interconnected with LoRa modules, a gateway device sends all generated packets over the internet to the database server. Thanks to Lora's long-distance communication, the authors stated that they could cover an area of 25 square kilometers for less than $5000 with 100 sensor nodes.

In addition, inexpensive options for sensor nodes can be implemented in networks with fixed Cluster Heads, while in mesh networks, all nodes usually have similar capabilities. As mentioned earlier, communication techniques typically used in wildfire detection are expected to support long-distance transmission. Nevertheless, for networks in a relatively small forest where the detection range of sensors is limited, technologies such as Bluetooth may be preferred. However, this technique would be both expensive and difficult to manage to cover large forested areas. For instance, in [10], ZigBee communication is chosen to transfer packets. Despite a fast 6-minute fire detection, a 560-acre park in the city was covered and a mesh topology was proposed for scenarios with more nodes. Once all decisions have been made, the network is deployed in the forest and the collected data is analyzed. Rule-based fire detection can be done, as well as smart systems that can detect false alarms with Artificial Intelligence (AI) techniques. Although fundamental algorithms that make decisions by checking certain threshold values for measured parameters are sufficient for fire detection, data-driven learning techniques are more popular for systems that can operate with high accuracy in every forest with minimal false alarms. In order to avoid false alarms, an unsupervised dataset was used to cluster alarms into false and

true using the k-means technique in the study [11]. Multiple linear regression models were then trained with these data. In [12], the decision whether there is a fire or not is made by RNN models. The model was trained with the data from the sensor nodes and then the incoming data was evaluated with this model and the fire was decided. Also in [13], a similar study is conducted with ANN models. In this study, instead of two classes such as the presence and absence of fire, different classes such as fire is about to start are also included. Predicting the area of fire spread can also be important to reduce its impact. In a study [14], wind sensors and artificial intelligence techniques were used to estimate the area of fire spread.

3. DIGITAL TWIN

3.1 Digital Twin & Digital Twin Networks

The digital twin that emerged in the early 21st century is a technique for modeling physical phenomena. Its research and use accelerated with the development of data collection and data analysis technologies. Prior to digital twins, modeling a system was needed for purposes such as testing extreme scenarios. Computer-aided simulations could be used to meet these demands. For example, packet-based event-driven simulators such as NS-3 [15], and OmNet++ [16] are widely preferred for modeling computer networks. However, often in such simulators, even a small change in the system may require the entire system to be re-simulated. In addition, it may not be possible to run these simulations in real time, as it is highly computationally expensive to compute simulations of particularly detailed and complex objects. Simulating a sophisticated computer network can take hours with the tools above [17]. Digital Twins, on the other hand, are working to a groundbreaking advantage in these problematic areas of simulators. Since DTs receive data from the physical system in real time, they can observe changes in the system. Thanks to their structure, they are also successful in adapting to these changes. As is familiar with Artificial Intelligence (AI) models, although training DTs can be time-consuming, it is considerably faster to run tests with the trained model. It is also possible to run these models in real-time for some use cases. For example, DTs can be used for predictive maintenance in the manufacturing industry, for modeling patients in personalized healthcare, or for planning smart cities.

Since DTs need to make inferences from data, they are often a variant of AI models. The appropriate technique may vary according to the structure of the system being modeled. RNN models are more suitable for objects where sequence-based data is generated, while Convolutional Neural Network (CNN) models are more suitable when data in image format must be processed.

Computer networks are also commonly modeled with DTs. DTs are used in areas such as optimization, monitoring, testing what-if scenarios, and training personnel. The fact that neighborhood relations are a very important parameter in these networks highlights the use of specific models for modeling these networks. Fortunately, there are models developed in the literature that can detect these spatial and temporal data. Graph Neural Networks (GNNs) are one of the most promising methods in this field. This method, which is often used to build DTs of computer networks, will be analyzed in the next subsection.

3.1.1 Graph neural networks

GNNs are a specialized data learning technique for analyzing data represented by graphs, such as protein structures, social networks, etc. It is used to understand the connections between people in social networks, develop new protein structures, and find new relationships in information graphs. Because it can perceive complex relationships in graphs and extract local information from them, it can also be generalized to more different graphs with similar characteristics. Although it varies from model to model, features held at nodes and edges are shared with neighboring nodes over several iterations to update the values of these features. This method is called message passing. This is the basic mechanism of GNNs. The messages from the neighbors are combined with the Aggregate function and the Update function is used to obtain the new state from the current state and the output of the function. How many hops away the message is received from neighbors is a hyperparameter of the GNN model, as are the Update and Aggregate functions.

Many types of GNNs exist, including Graph Convolutional, graph SAGE, Graph Attention, and Graph Transformer Networks. GCN is a model that carries the filter structure from CNN to GNNs. As in CNN, these filters can be learned. GraphSAGE performs aggregation by sampling some of the neighbors to make the GCN more scalable. On the other hand, Graph Attention Network (GAT) models the attention layer in Transformers in GNNs. Graph Transformer Network can also interpret long-range relationships thanks to its self-attention mechanism.

3.1.2 GNNs in computer networks

The main use of GNNs in computer networks is to model the network with high accuracy. Two different RouteNet studies compare the performance of GNN-based models with Queueing Theory modeling for network metrics. Both GNN-based models significantly outperformed the queueing theory model [18] [19].

Thanks to the real-time and accurate network modeling of GNNs, the optimization of many parameters of the network has been facilitated. One of these areas is packet routing optimization. In Almasan et al.'s work [20], the Deep Reinforcement Learning (DRL) agent optimizes packet routing to maximize allocated bandwidth using the network's GNN-based DT. The agent not only outperformed Fluid models but also, unlike these models, was able to adapt to dynamic changes in the network such as link failure, and could be generalized for networks with similar characteristics. In a similar study [21], a GNN called TwinNet was developed for network optimization. Instead of DRL, a classical optimization algorithm was used to optimize the average per-flow delay. The model worked quite successfully compared to RouteNet and Multi Layer Perceptron (MLP) alternatives. It achieved a Mean Absolute Percentage Error (MAPE) of around 3 percent and an R^2 score of more than 97 percent. In optimization, it has been much more successful than fluid-based models, especially in high-density traffic, since those models cannot model queueing delay. There are also studies on how to model networks into GNNs. In [17], modular GNN models modeled in terms of expressiveness and granularity are designed as xNet. For 3 different usage scenarios, the performance of the proposed model is tested. The model was able to predict delay with a MAPE rate of less than 5 percent for data with sampling intervals of ms.

Network slicing is a technology designed for next-generation network applications where the physical network is divided into virtual networks. GNNs can also be utilized in the management of these networks. In the paper [22], in a scenario with different delay agreements for different network slices, it is tested by training a Graph Linformer Network (GLN) based DT with Federated Learning (FL) and using a heuristic optimization method for both delay estimation performance and meeting these agreements. The model both outperformed state-of-the-art GNNs and was

able to meet the Service Level Aggrement (SLA) requirements with the optimization algorithm. In another study [23], end-to-end latency was measured for all slices. It was able to predict DT based on GraphSAGE with less than 5 percent error on all slices. Furthermore, similar to the tests in similar studies in the literature, the performance in link failures and SLA optimization performance were also tested. In addition, the model was trained for the jitter metric to show that DT can be trained faster for a different metric.

GNNs are also used to predict the traffic of the network. In a study in the literature [24], the feature extraction technique was used to derive features from the data in the network and predict the traffic. Similar performance was achieved with the extracted features and the training time was significantly reduced.

4. WORK DONE

This thesis proposes a forecaster layer to facilitate the integration of DTs into IoT networks. For this purpose, a series of IoT networks were simulated in the designed simulator, and the packets generated by these networks were collected. Subsequent packets were predicted with the collected packets. These predictions were given to the DT to estimate the network's throughput. Actual throughput values are obtained from the simulation and compared with the predictions. As seen from this process, the studies can be divided into three parts: simulator design, data packet forecaster, and DT development. These sections and the data exchange between them are shown in Figure 4.1.

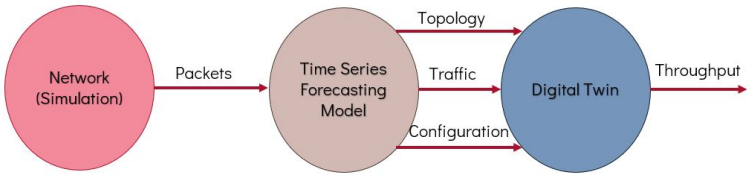

Figure 4.1 : Model of the system.

To make the simulation in the study more meaningful in the real world, a real forest was taken as a reference. As mentioned earlier, the Aegean and Mediterranean regions are the most dangerous regions in Turkey in terms of forest fires. Therefore, the reference forest in the simulation is selected from this region. The simulation areas were designed based on a forest area around Akseki, Antalya, which can be seen from Figure 4.2. In the network simulated in the study, data from temperature and humidity sensors are assumed to be carried, although packets at the application layer have no impact. A clustered topology is preferred. Intra-cluster communication was provided by LoRa, and inter-cluster communication was provided by General Packet Radio Service (GPRS).

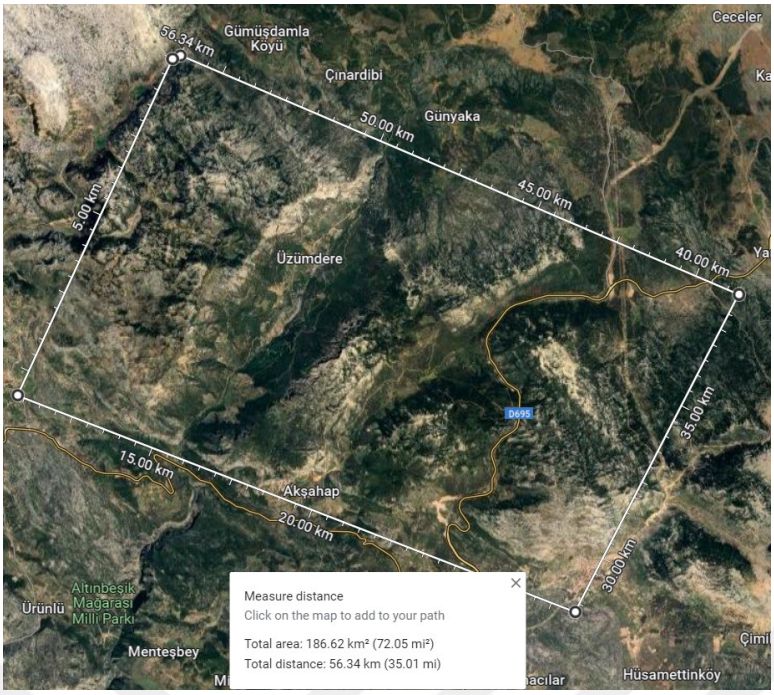

Figure 4.2 : Modeled forest.

4.1 Simulator

The simulator is responsible for creating a network with the desired parameters and running this network for the desired period. It also saves the data of all packets sent during this time in a Comma Seperated Values (CSV) file. The file stores the sending and receiving nodes of the packets, the start and end times of the transmission, the signal strengths, and whether the packet reached the receiving node. Several network scenarios with up to 16 clusters were tested in the simulator. Some of these scenarios involved collecting data from hundreds of nodes over a day, so physically designing these networks was not feasible. Therefore, the studies were carried out in the simulator. In addition, DT needs a plethora of collision data to understand the behavior of the network. Obtaining such data from the physical network would be quite laborious. In physical networks, it is considered wise to obtain this training data from the simulator.

A custom simulator was designed to be used in the study, capable of performing the tasks mentioned above. Advanced event-based network simulators such as NS3 and Omnet++ could accomplish these tasks comfortably. However, it was realized that

the most effective and easy way to collect and report data in the format required by the study would be to design a new simulator. Also, although these simulators have LoRa modules, there was not enough documentation to design topologies other than LoRaWAN as proposed in the network.

All parameters used in the simulation can be found in Table 4.1. All functions of the running simulation are contained in the main Simulator class. The simpy library is used to create event-based simulations. To avoid confusion, Simpy's Environment, which contains all events, will be referred to as env. The environment in the study is the class that generates the dimensions and noises of the simulated area. Only thermal noise was calculated as noise in the environment. It should be noted that in practice other noises will also affect the packet transmission.

The simulation also includes the Topology class that designs the network. In this class, devices are positioned according to parameters such as the number of sensor nodes, number of clusters, and number of servers. The size of the environment is also important in the physical placement of the devices. First, the servers, then the cluster heads and nodes are placed in a grid environment. The number of rows and columns of the grid is another parameter of the simulation. In the simulations, it was assumed that there was only one server and the number of clusters and nodes was determined as 9 nodes per cluster. The number of rows and columns was chosen as close or equal as possible. Some of the topologies constructed for simulation can be seen in Figure 4.3.

It is assumed that the server device receives data from the base station in the environment. The packets sent by the Cluster Heads with GPRS signals are listened to by the base station in the center of the environment, given the single station parameter in the environment, and transmitted to the server.

The communication between the Cluster heads (CHs) and the Base station is organized by the TDMA scheduler. The duration of a TDMA window is taken as a parameter, while the length of the frame allocated for CHs is found by dividing this duration by the number of CHs. If there are packets transmitted to the CH by the nodes, it will transmit these packets to the base station in the frame allocated to it.

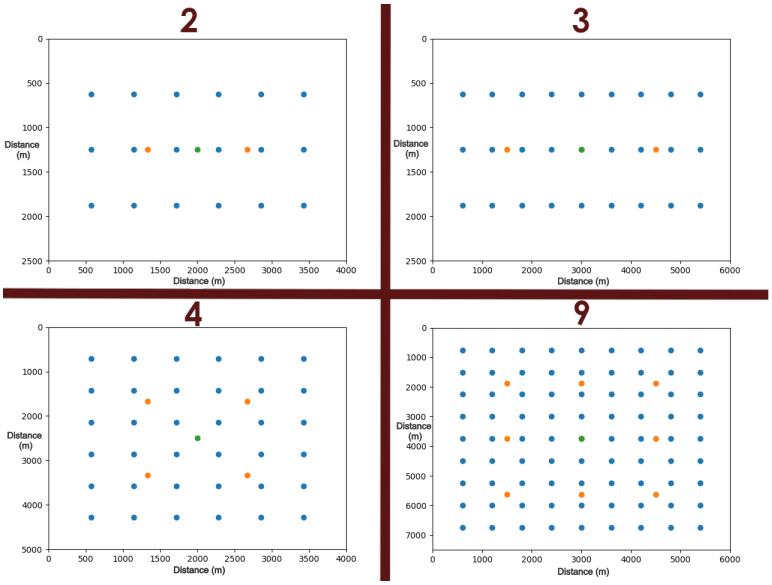

Figure 4.3 : Simulation topologies.

Sensor nodes send their measurements to the nearest CH. Since there is a limited number of nodes in a CH and the network structure is kept as simple as possible, ALOHA is used for intra-cluster medium access control. The packet generation of the nodes was enabled by a pseudo-random Poisson distribution with the mean as the parameter. In this way, packet formation is modeled in such a way that it is neither easily predictable nor completely random, but close to reality. In such sensor networks, data can also be generated periodically or event-based bursty. These scenarios were not included in the study.

LoRa was used for intra-cluster communication and GPRS technologies were used for inter-cluster communication. The nodes sent their packets only with SF7, both because it was considered to be the most convenient way to use it and to facilitate the simulation. The bit rates, transmission powers and frequencies, and SNIR thresholds of the nodes were determined as realistic values as can be seen in Table 4.1. Since CHs would need faster communication and energy problems would be more tolerable, they were designed to communicate with GPRS instead of LoRa. Again, similar parameters were assigned realistic values for GPRS. To clarify, it should be noted that CHs must-have modules that include both communication technologies. The path loss was also modeled in the system to accurately model packet states. Hata's path loss model [25] was selected for use. Since communication technologies have different

frequencies, by formula, and of course due to physical realities, they experience different power losses over similar distances.

Finally, once the simulation is complete, all generated packages are reported with the previously mentioned properties. These features are chosen so that the forecaster can make its predictions and the actual throughput can be calculated during the evaluation phase. With the help of the Logger object of the simulator, all this data is reported to a file.

Table 4.1 : Simulation parameters.

Parameter	Value	Parameter	Value
runtime	24 hrs	mean packet period	1 min
packet size	50b	CH min rx power	-130 dBm
node height	10 meters	GW min rx power	-115 dBm
GW height	1000 meters	Temperature	300 K
sensor tx power	14 dBm	CH tx power	33 dBm
sensor tx frequency	433 MHz	CH tx frequency	950 MHz
CH bitrate	50000 Kb/s	GW SNIR threshold	0 dB
sensor bitrate	5700 Kb/s	CH SNIR Threshold	-6 dB

4.2 Forecaster

The Forecaster module tries to predict future data with data from the simulator. The first 80% of the collected data is reserved for training and the last 20% for testing. As can be seen from Table 4.3, predicting future data directly with current data is different and more complex than classical AI problems. Therefore, an intermediate step called data augmentation was applied to the report before forecasting.

In the data augmentation step, a time series of data is generated for each node in the network to see if they have sent data during the entire simulation. At the specified sampling interval, the simulation is sampled to determine whether the node is currently transferring data or is in sleep mode. The sample data generated for a node can be examined from Table 4.4. Then, for each node, time series binary classification is performed using the AI model with the specified 80/20 ratio.

Predictably, the sensor nodes are asleep for most of the simulation. This makes the dataset created for the problem imbalanced. Therefore, if the data generated after

augmentation is given to the model after preprocessing, the model can make more accurate inferences. Thus, the undersampling technique was applied. Oversampling techniques such as Synthetic Minority Oversampling Technique (SMOTE) are also preferred to remove data imbalance. However, since the dataset generated for each node is massive, more than 800 thousand samples for 1 day simulation time and 100ms sampling interval, and the model needs to be retrained for each node, it was decided that undersampling is much more appropriate. For a node, 4 times the number of samples it sends, the sample it sleeps is randomly selected and the model is trained.

After addressing the data imbalance issue with undersampling, predictions are generated for each node for the last 20% of the simulation time, including the status of whether they have sent a packet or not. Since a prediction problem involving sequence data was attempted to be solved, the RNN-based Long-Short Term Memory(LSTM) model was preferred for forecasting. This is because the recurrent connections in the architecture of RNN models allow the current input to be influenced by past inputs by maintaining a hidden state that is updated at each time step, effectively creating a memory of the sequence. Since LSTM separates short-term and long-term information through its gates, it is considered to be the most suitable among different RNN techniques for this problem. Later in the study, different forecasting techniques were also tried for comparison. The model includes 3 layers of LSTM and an MLP in the final layer. The window size is selected as 25. Other parameters can be seen in the Table 4.2.

Table 4.2 : Forecaster parameters.

Parameter	Value
Learning Rate	0.01
Undersampling Ratio	1/4
Neuron Numbers	128, 64, 32, 1
Window Size	25

The Forecaster module is an important contribution to this thesis and a critical part of DTs' work in an IoT environment. As mentioned, it is not easy to ensure continuous and two-way communication in these networks. Forecaster's predictions virtually

replicate this communication from physical object to digital twin consistently and accurately.

The data imbalance is also a consideration when evaluating the model's performance. The fact that a large proportion of the data contains sleep states may favor models with a bias toward this option. Hence, it is more convenient to consider the precision and recall values of the states than to consider the accuracy directly. Also, since it is a binary classification problem, the precision and recall values of the majority class tend to be high for biased algorithms. The F1 score is an evaluation criterion that shows the harmonic mean of the precision and recall values. Therefore, the F1 score of the packet-sending state was chosen as the evaluation criterion.

Table 4.3 : Sample report.

Source	Destination	Tx Start (s)	Tx End (s)	Size (b)	Strength (dBm)	Status
12	36	0.105	0.180	50	-93.83	0
4	37	0.300	0.373	50	-93.83	0
28	39	0.503	0.576	50	-72.26	1
28	39	1.147	1.220	50	-72.26	1
8	36	3.948	4.022	50	-78.72	1

Table 4.4 : Sample data augmentation for a node.

Time (s)	Status
0	0
0.1	0
0.2	1
0.3	0
0.4	0

4.3 Digital Twin

Once the models have been trained and run, the state of each node is obtained separately for each node at specific time t. However, for DT to predict the throughput at a time t, it needs the states of all nodes at that time. Therefore, for each time instant, the state information of the nodes is used to generate the state information of the whole network at that instant. The state information of the network at a time instant is a

graph that contains the topology of the network and the properties of each node and edge. The Pytorch geometric library is used and the graph is given to the GNN in the format required by this library. For edges, the library expects two-dimensional arrays containing source and destination. The node and edge properties must also be provided as matrices.

DT takes the current state of the network and outputs the current throughput. One may question why DT is needed to calculate throughput if the entire state of the network is known. Computer networks are sophisticated structures. It cannot be assumed that every package arrives at its destination. Packets can be lost due to collisions or interference, especially in wireless networks. The occurrence of these situations also varies from network to network. It is therefore essential to calculate throughput with an intelligent model.

The chosen AI model needs to comprehend these situations. For this, it is crucial to make sense of the neighborhood relations of the nodes of the network. GNNs are models designed specifically for this field and can successfully capture these relationships. Therefore, the GNN-based GAT model is employed as the DT model in this study. The GAT model can embed features for both nodes and edges. The link and the node have as features the packet sending status and a parameter specifying whether the node is a sensor or a cluster head. Both features are binary values. The model also includes 3 MLP layers after the GAT layer. The parameters of the design can be found in Table 4.5. Since the problem could not be solved optimally with the same parameters in different cluster simulations, some parameters were changed specific to the simulations.

Mean Squared Error (MSE) is used as the evaluation metric to measure the throughput detection performance of the model. Furthermore, the R^2 metric was used to examine the correlation between the actual throughput and the predictions.

In order to compare the performance of some GNN-based models other than the GAT model, models such as GCN, GraphSAGE, and Transformer Convolution were also run for a few of the simulation scenarios.

Table 4.5 : GNN parameters.

Parameter	Value
learning rate	0.001
batch size	256
hidden layer dimension	128
dropout rate	0.1
train ratio	0.8
epoch number	50

4.4 How The System Works?

As demonstrated in the model of the system Figure 4.1, data packets are generated by the custom network simulation module. Then, the developed forecasting model predicts the future packets generated in the network. From these predicted packets, the DT forecasts the throughput of the network. However, generated data from the previous steps cannot be used directly by both the forecaster and the DT. Hence, data formation or aggregation steps are added between the modules additionally to accomplish the data formats of the models.

With the added two steps, there are a total of 5 phases of the system. The first phase is packet data generation by the custom simulator. The simulator runs the simulation according to the given parameters and generates reports for packets as explained in the simulator section and can be seen from Figure 4.4. The forecaster predicts the successive packets based on the given packet data as in Figure 4.5. However, this problem is arduous to satisfy with the data in the report. Thus, the problem is divided into multiple time-series forecasting more straightforward decisions. To do that, with predefined sampling intervals, for every node, the forecaster infers the node's state, which is either sleep or transmit throughout the simulation. This is the data augmentation step and it is shown in Figure 4.6.

Aftermentation, binary state forecasting is performed for every node to predict the future states of the nodes, as illustrated in Figure 4.7. Nonetheless, because the DT needs the total state of the network to predict the throughput, The entire state of the network has to be constructed from the forecasted states of the nodes for each sampled time that throughput is predicted. This is done by the network state generation step,

which produces these states for the DT as expressed in Figure 4.8. Lastly, the DT predicts the throughput of the network for each generated state. The prediction of DT for an example time sample is demonstrated in Figure 4.9.

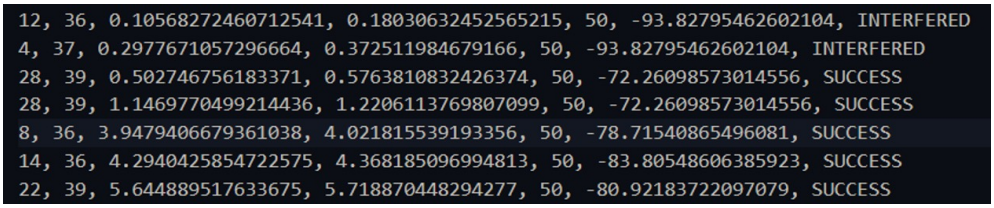

Figure 4.4 : Generated packet reports with simulator.

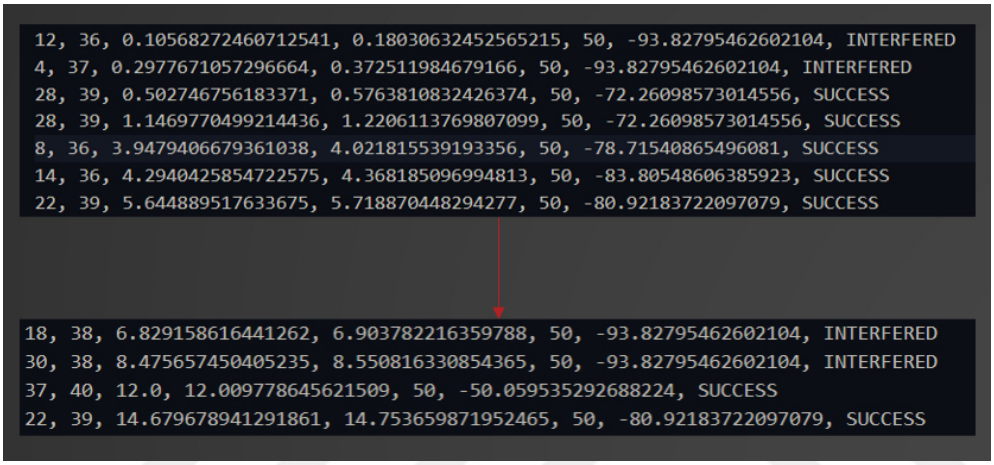

Figure 4.5 : Expected forecasting operation.

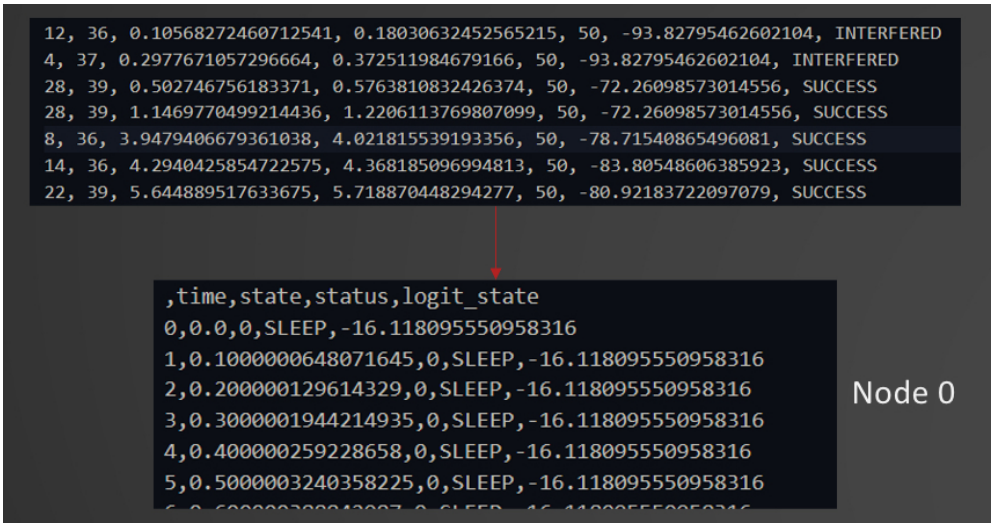

Figure 4.6 : Data augmentation for a node.

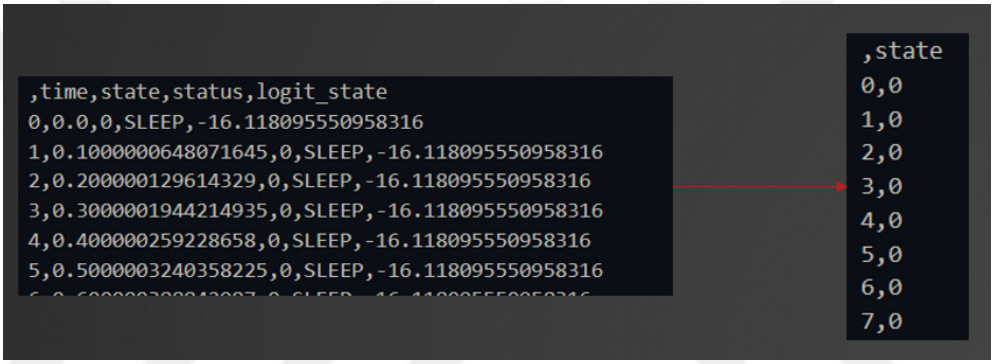

Figure 4.7 : Forecasting for a node.

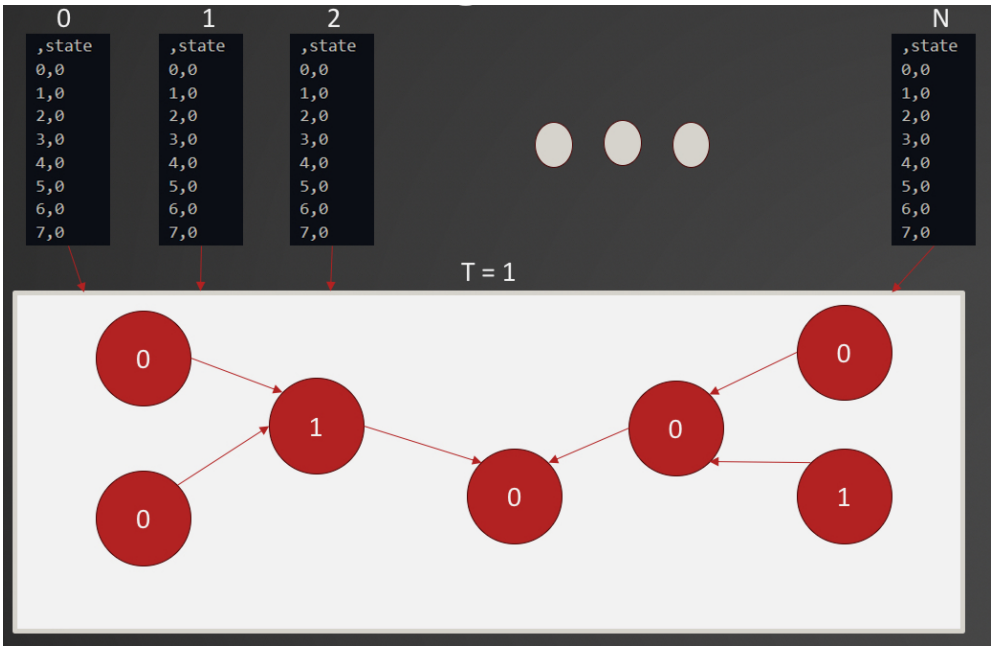

Figure 4.8 : Network state generation with forecasting results.

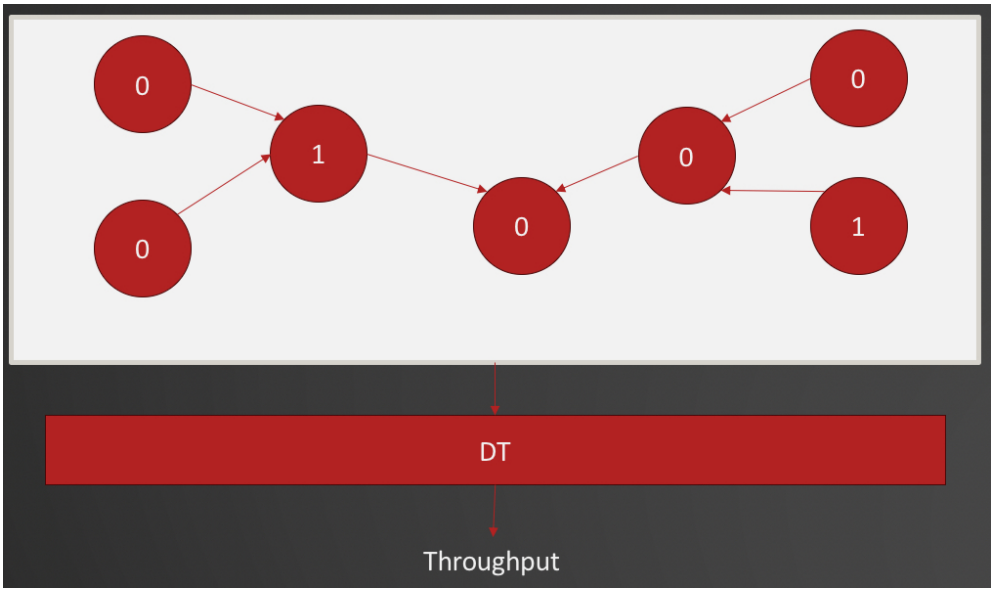

Figure 4.9 : Throughput prediction with the DT.

5. RESULTS & DISCUSSION

The performance of the system was tested for different network topologies. Networks with 2, 3, 4, 5, 9, and 16 clusters were used in the tests. Forecaster and DT modules were first evaluated separately and then integrated as a whole system. F1 score for Forecaster and MSE and R^2 score for DT were used as metrics. A separate test scenario was designed to observe the collision understanding capacity of DTs. Results were also obtained using small-scale networks to compare the performance of the selected LSTM and GAT pair with other techniques.

5.1 Forecaster Performance

To measure the performance of the forecaster, the predictions of the forecaster were compared with the actual results in networks with different cluster numbers. Figure 5.1 shows the performance of all nodes in the network at a 95 percent confidence interval for a run. The average performance of the nodes was around 0.9 F1 score regardless of the number of clusters.

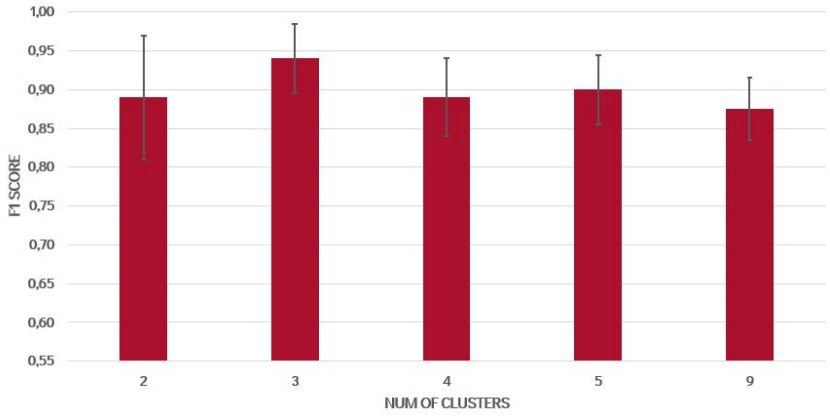

Figure 5.1 : Forecasting performance for different networks.

The performance of each node was also analyzed separately to observe the effect of the location of the nodes or the cluster to which they are assigned. For this test, the simulation was run for 3 rounds on the same network. The results with confidence

intervals for these 3 results are shown. Figures 5.2, 5.3, 5.4, and 5.5 show the performance of nodes for networks with 2, 3, 9, and 16 clusters respectively. From these results, no effect of node locations or clusters on the results could be detected. Also, being a sensor node or cluster head had no obvious effect on the results.

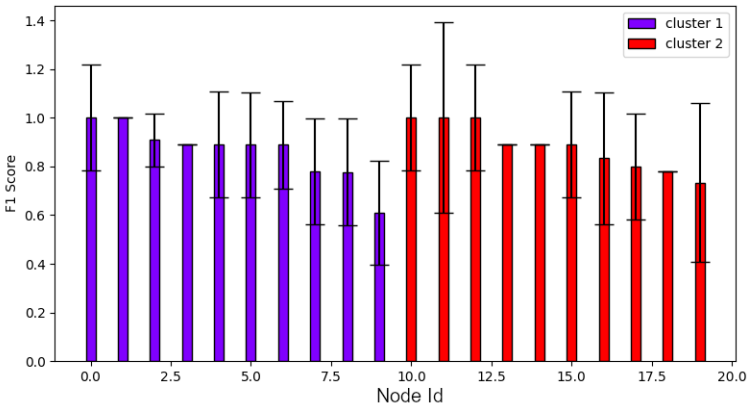

Figure 5.2 : F1 scores of the nodes in 2 clustered network.

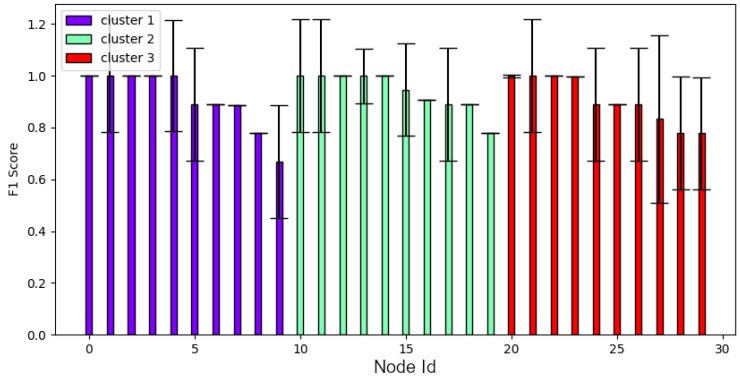

Figure 5.3 : F1 scores of the nodes in 3 clustered network.

5.2 DT Performance

In order to measure the performance of DT independently of the forecaster, after augmentation of the simulation data, DT is given real results instead of forecaster predictions as test data. In this scenario, it was observed that the MSE error of DT did not increase when tests were performed for different cluster numbers. These results were expected, as many studies in the literature have shown that GNN-based DT techniques can be scaled and generalized to model computer networks. The results can be examined in Figure 5.6.

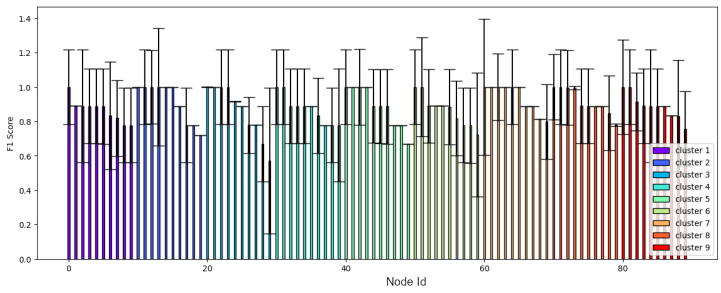

Figure 5.4 : F1 scores of the nodes in 9 clustered network.

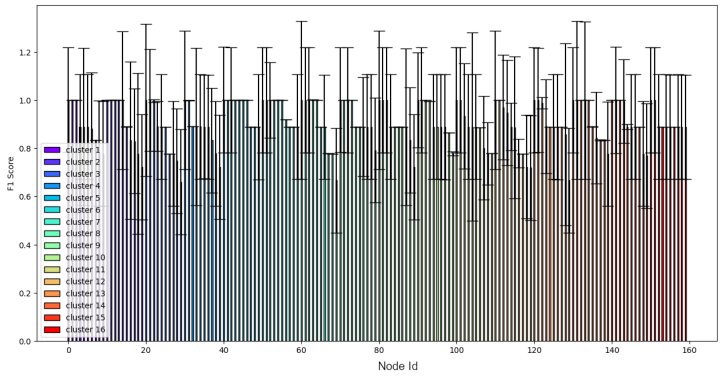

Figure 5.5 : F1 scores of the nodes in 16 clustered network.

5.2.1 Effect of collisions

One of the ways to reveal whether DT understands the structure of the IoT networks being operated is to measure how well it can capture collisions. For this reason, the dataset was first organized as if all packets in the network were reaching their destination. With this organized data set, the model was trained and estimated the throughput. The actual throughput was also calculated with this data and the results were compared with the results where collisions could occur. Comparative MSE and R^2 results are plotted in the graph in Figure 5.7. As can be seen from the similarity of the results, DT can detect collisions at an acceptable level.

5.3 System's Performance

After obtaining the individual performance of the modules, the integrated performance of the whole system was also tested. Since the system finally estimates the throughput,

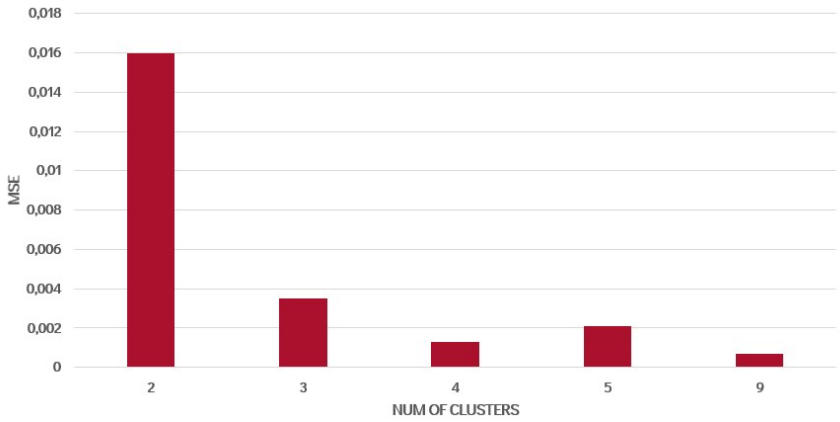

Figure 5.6 : MSE for DT in different simulations.

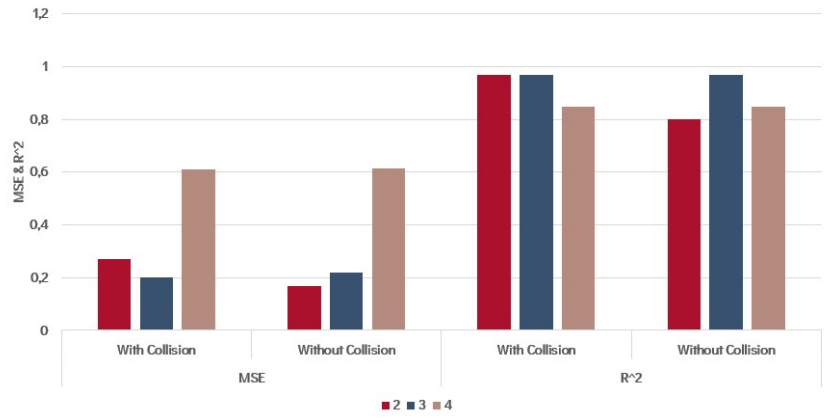

Figure 5.7 : Performance of DT with and without collisions.

like the DT module, the MSE and R^2 metrics are used to evaluate the performance. For small-scale networks, the performance of the whole system performs remarkably well, as performs the modules, while for a 9-cluster network, the performance is not satisfactory. As can be seen from Figure 5.8, the R^2 score in the network with 9 clusters drops below 40 percent. This implies that the correlation between results and predictions has been lost.

5.4 Comparison with Other Models

In addition to the selected LSTM and GAT methods, other state-of-the-art models have been previously studied in the literature. For networks with 2 and 4 clusters, the proposed system is also designed with these models, and the results are tested. GCN, GraphSAGE, and Transformer Convolution GNN models were selected for DT, while Gated Recurrent Unit (GRU), Artificial Neural Network (ANN), and Support

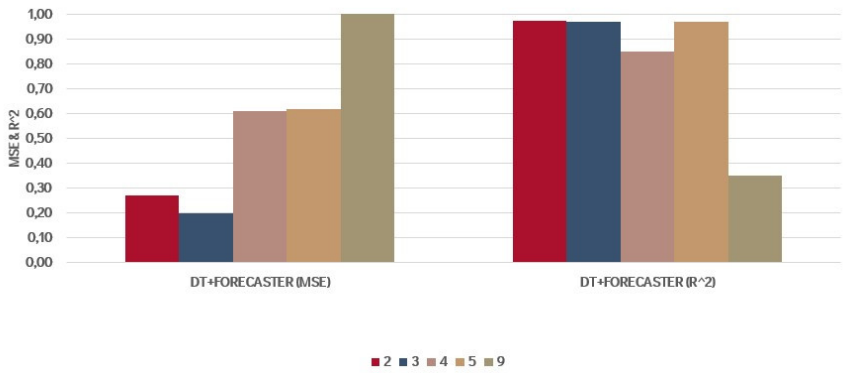

Figure 5.8 : Performance of the system.

Vector Regressor (SVR) techniques were applied for Forecaster. As can be seen from Table 5.1, there are slight differences in the results between the models. Only the ANN model was not successful in the sequence-based binary classification task. According to the results, it is possible that the pairing of GRU and GraphSAGE provides a tiny performance improvement over the chosen pair. However, it is not enough to solve the scalability problem, nor does it significantly improve the satisfactory performance of the model in small-scale networks.

Table 5.1 : Comparative results table for different models.

Forecaster	LSTM				GRU				ANN			
Metric	2CH		4CH		2CH		4CH		2CH		4CH	
DT	MSE	R^2	MSE	R^2	MSE	R^2	MSE	R^2	MSE	R^2	MSE	R^2
SAGE	0.219	0.885	0.408	0.971	0.224	0.882	0.316	0.978	27.29	-13.36	14.09	0
GAT	0.518	0.727	0.426	0.970	0.743	0.609	0.386	0.973	-	-	291.2	-19.7
Transformer Conv	0.222	0.883	0.489	0.965	0.228	0.880	0.455	0.968	26.15	-12.76	252.1	-17.0

Finally, a time series comparison of the actual throughput values of the throughput of the system and the predictions are made. It can be examined from Figure 5.9. The blue line shows the actual throughput values while the orange line shows the predictions of the system. The system's prediction for the first 100 seconds is taken as a basis. It is observed that the estimates mostly coincide with the actual values.

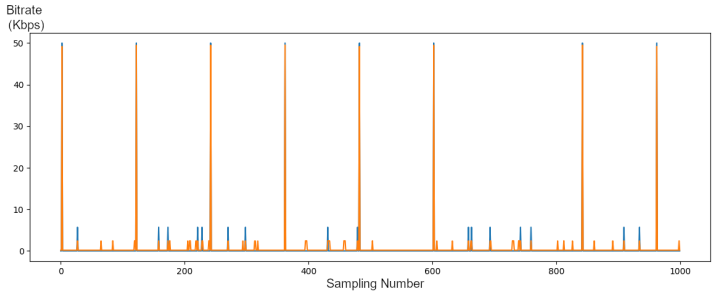

Figure 5.9 : Actual throughput vs. predicted throughput.

5.5 Discussion

The tests have revealed that the proposed system is especially promising for small-scale IoT networks. With the proposed method for these networks, DT integration can be performed with high accuracy and used in applications such as network optimization and testing of different scenarios. However, in larger-scale networks, the system's performance degrades critically. The system is far from being scalable.

The main reason for the non-scalability is claimed to be the cumulative accumulation of the errors made by the forecaster at each node when generating the current state of the system at a point in time. While the performance of the forecaster for each node is decent, deciding the instantaneous throughput of a system with hundreds of nodes by trying to predict whether all nodes should send packets individually can only work properly with extremely sharp node state accuracy. Perhaps, for large networks, instead of using a technique that makes node-based predictions, more holistic prediction mechanisms may be preferred. For example, one could try to estimate how many packets are currently being sent for the entire cluster. This changes the problem from binary classification to multiclass classification.

The performance of the proposed forecaster system was found to be satisfactory. Not only does it know the interval over which a very high percentage of packets will be transferred, but it is also shown to be scalable no matter how large the network is. Running the model separately for each node contributes a significant amount in this respect. It can be considered that the purpose of the model is to estimate the package generation function of the nodes. Therefore, it is impossible to

evaluate the model's performance independently of the packet generation techniques of the nodes. The tests show that the model works well both for CHs that generate packets periodically but of different sizes and for nodes that generate packets of fixed size but at pseudo-random times. In networks that generate packets with different functions, the module's performance will naturally change. However, the technique is considered to be a reasonable strategy under all circumstances. In an environment with completely random packet generation, the module may not be able to work, but in this environment, it will not be possible to make accurate throughput estimation with any method.

The fact that the model tries to estimate package production functions makes its performance independent of location or cluster. In real networks, these parameters may certainly change the packet generation of nodes, but in the simulation, the same technique is applied to all nodes.

GAT-based DT has also performed well. Many studies on Digital Twin Network (DTN) in the literature have already shown that GNN-based AI models are groundbreaking in DT applications. In this study, GNN-based DT was found to be a suitable choice also with forecasting data. The other GNN-based models tested also predicted the network's throughput with generally close performances. From these results, it can be concluded that GNNs are an important option for problems involving graph-based data types.

Due to the scope of the study, the effect of some parameters considered to be important could not be analyzed. For example, changes in the performance of the system, especially the forecaster, under different package generation scenarios could not be studied. This kind of test would be meaningful in demonstrating the generalizability of the system. It is also important to conduct simulations under different environmental conditions for similar reasons. However, these tests could not be performed due to the limitations of the study. For example, due to the effect of the height of nodes on path loss, packet loss patterns depend on the node height parameter.

It may also be useful to test Transformer-based models in the forecaster module. Although it is not expected to solve the scalability problem due to the source of the

scalability problem, demonstrating their performance can improve the study since they are cutting-edge technology in sequence-based classification. Finally, the real-time inference performance of the system can also be tested. One of the most important advantages of DT systems over simulations is that they can be used in real-time. Therefore, it is important that this proposed system also produces real-time results. In order for the system to produce results, the results of all forecasters should be obtained first and then the state of the network should be created and the result of DT should be waited. Although forecasters can be run in parallel, results from all of them must be obtained before running DT. It is worthwhile to test whether the performance of this working technique will be sufficient for real-time operation.

6. CONCLUSION & FUTURE WORK

The world's rising average temperature is increasing the risk of forest fires. Therefore, the importance of forest fire management systems is also increasing. IoT sensor networks are frequently used in these systems to detect forest fires early. However, due to the different requirements of forests, the structures of these networks also vary widely. This requires specific decisions to be made in the management of each network. Due to the size of forests, the management of these large networks can be difficult. DT technology can help network administrators in this area. DTs are widely used in computer networks for optimization and running test cases. Nonetheless, DTs are challenging to use in IoT networks due to their constant communication requirements. Therefore, this paper proposes a forecaster mechanism to facilitate this integration. The proposed model generates the data needed by DT by trying to predict the network's packets in advance.

Computer networks are complicated structures where neighborhood relationships are important. Understanding such structures can be compelling for some classical AI models. Fortunately, GNN-based models are capable of understanding the spatial and spectral data required for these networks. Also, some critical decisions have to be made in the design of the network due to the large area to be covered in forests. While LPWAN technologies such as LoRa meet the requirements of these networks, such as high-distance communication and low power consumption, the packet generation frequency and size of the nodes remain within the limits of the data-carrying capacity of these technologies. Cellular communication technologies also enable CHs to send aggregated data.

There are studies in the literature on IoT networks for forest fires and DTs for computer networks. Clustered, mesh topologies and communication technologies such as LPWAN with low power consumption and short-range technologies such as Zigbee and BT were generally preferred in the networks. In DTNs, GNN-based models are the

most widely used option in recent studies. DT has been used in areas such as network slicing, anomaly detection, optimization, and testing of what-if scenarios. However, the integration of DTs and IoT networks and the Forecaster method used in this study is a novelty compared to previous studies.

In the study, simulations were created for networks with different numbers of clusters, and the generated packets were collected. Then, the Forecaster, which was trained with these packets, was asked to detect the packets that the network would produce for the next time. Then, DT was given the packets detected by the forecaster and was expected to determine the throughput at the given moment. The actual throughput values for these times were also obtained from the simulation and the performances of the system and the modules were evaluated separately.

In the tests, it was observed that the forecaster module correctly recognized the sent packets with an F1 score of approximately 0.9 for each network. Moreover, the DT module, when trained independently of the forecaster, achieved an MSE score lower than 0.02 and a high R^2 score of 0.8 in each network. In the test of the network's understanding of collisions, the difference between the collision on and off scores is less than 5 percent. However, when the whole system was integrated, the MSE error increased by more than 100 times and the R^2 score dropped below 40 percent for the network with 9 clusters, although similar scores were obtained for the small-scale networks.

The proposed system seems to have a scalability problem. It is assumed that this is due to the accumulation of errors in the individual predictions of all nodes. A more holistic forecasting approach and a forecasting model can be considered as future work. Also, as future work, testing the system with different traffic generation scenarios and environmental conditions could improve the study. Also, instead of throughput prediction, packets can be prioritized and DT can forecast the number of incoming packets for each prioritization level. This technique can be critically beneficial for optimizing fire alert packets' delivery rates.

REFERENCES

[1] **OGM** (2024). *Official Statistics*, https://www.ogm.gov.tr/tr/e-kutuphane/resmi-istatistikler, https://www.ogm.gov.tr/tr/e-kutuphane/resmi-istatistikler.

[2] **Chowdary, V.**, **Gupta, M. and Singh, R.** (2018). A Review on Forest Fire Detection Techniques: A Decadal Perspective, *International Journal of Engineering & Technology*, *7*, 1312.

[3] **Özel, B.**, **Alam, M.S. and Khan, M.U.** (2024). Review of Modern Forest Fire Detection Techniques: Innovations in Image Processing and Deep Learning, *Information*, *15*(9), https://www.mdpi.com/2078-2489/15/9/538.

[4] **Kocakulak, M. and Butun, I.** (2017). An overview of Wireless Sensor Networks towards internet of things, *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, pp.1–6.

[5] **Devalal, S. and Karthikeyan, A.** (2018). LoRa Technology - An Overview, *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, pp.284–290.

[6] **Farooq, M.O.** (2019). Introducing Scalability in LoRa-Based Networks through Multi-Hop Communication Setups, *2019 IEEE Global Communications Conference (GLOBECOM)*, pp.1–6.

[7] **Grover, K.**, **Kahali, D.**, **Verma, S. and Subramanian, B.** (2020). WSN-Based System for Forest Fire Detection and Mitigation, ***B. Subramanian**, **S.S. Chen and K.R. Reddy**, editors, Emerging Technologies for Agriculture and Environment*, Springer Singapore, Singapore, pp.249–260.

[8] **Molina-Pico, A.**, **Cuesta-Frau, D.**, **Araujo, A.**, **Alejandre, J. and Rozas, A.** (2016). Forest Monitoring and Wildland Early Fire Detection by a Hierarchical Wireless Sensor Network, *Journal of Sensors*, *2016*(1), 8325845, https://onlinelibrary.wiley.com/doi/abs/10.1155/2016/8325845, https://onlinelibrary.wiley.com/doi/pdf/10.1155/2016/8325845.

[9] **Saldamli, G.**, **Deshpande, S.**, **Jawalekar, K.**, **Gholap, P.**, **Tawalbeh, L. and Ertaul, L.** (2019). Wildfire Detection using Wireless Mesh Network, *2019 Fourth International Conference on Fog and Mobile Edge Computing (FMEC)*, pp.229–234.

[10] **Granda Cantuña, J.**, **Bastidas, D.**, **Solórzano, S. and Clairand, J.M.** (2017). Design and implementation of a Wireless Sensor Network to detect forest fires, *2017 Fourth International Conference on eDemocracy & eGovernment (ICEDEG)*, pp.15–21.

[11] **Dampage, U.**, **Bandaranayake, L.**, **Wanasinghe, R.**, **Kottahachchi, K. and Jayasanka, B.** (2022). Forest fire detection system using wireless sensor networks and machine learning, *Scientific Reports*, *12*(1), 46, https://doi.org/10.1038/s41598-021-03882-9.

[12] **Benzekri, W.**, **Moussati, A.E.**, **Moussaoui, O. and Berrajaa, M.** (2020). Early Forest Fire Detection System using Wireless Sensor Network and Deep Learning, *International Journal of Advanced Computer Science and Applications*, *11*(5), http://dx.doi.org/10.14569/IJACSA.2020.0110564.

[13] **Pokhrel, P. and Soliman, H.** (2018). Advancing Early Forest Fire Detection Utilizing Smart Wireless Sensor Networks, ***A. Kameas and K. Stathis**, editors, Ambient Intelligence*, Springer International Publishing, Cham, pp.63–73.

[14] **Imran**, **Iqbal, N.**, **Ahmad, S. and Kim, D.H.** (2021). Towards Mountain Fire Safety Using Fire Spread Predictive Analytics and Mountain Fire Containment in IoT Environment, *Sustainability*, *13*(5), https://www.mdpi.com/2071-1050/13/5/2461.

[15] **Riley, G.F. and Henderson, T.R.**, (2010). The ns-3 Network Simulator, **K. Wehrle**, **M. Güneş and J. Gross**, editors, Modeling and Tools for Network Simulation, Springer Berlin Heidelberg, Berlin, Heidelberg, pp.15–34, https://doi.org/10.1007/978-3-642-12331-3_2.

[16] **Varga, A.**, (2010). OMNeT++, **K. Wehrle**, **M. Güneş and J. Gross**, editors, Modeling and Tools for Network Simulation, Springer Berlin Heidelberg, Berlin, Heidelberg, pp.35–59, https://doi.org/10.1007/978-3-642-12331-3_3.

[17] **Wang, M.**, **Hui, L.**, **Cui, Y.**, **Liang, R. and Liu, Z.** (2022). xNet: Improving Expressiveness and Granularity for Network Modeling with Graph Neural Networks, *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, pp.2028–2037.

[18] **Ferriol-Galmés, M.**, **Rusek, K.**, **Suárez-Varela, J.**, **Xiao, S.**, **Shi, X.**, **Cheng, X.**, **Wu, B.**, **Barlet-Ros, P. and Cabellos-Aparicio, A.** (2022). RouteNet-Erlang: A Graph Neural Network for Network Performance Evaluation, *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, pp.2018–2027.

[19] **Ferriol-Galmés, M.**, **Paillisse, J.**, **Suárez-Varela, J.**, **Rusek, K.**, **Xiao, S.**, **Shi, X.**, **Cheng, X.**, **Barlet-Ros, P. and Cabellos-Aparicio, A.** (2023).

RouteNet-Fermi: Network Modeling With Graph Neural Networks, *IEEE/ACM Transactions on Networking*, *31*(6), 3080–3095.

[20] **Almasan, P.**, **Suárez-Varela, J.**, **Rusek, K.**, **Barlet-Ros, P. and Cabellos-Aparicio, A.** (2022). Deep reinforcement learning meets graph neural networks: Exploring a routing optimization use case, *Computer Communications*, *196*, 184–194, http://dx.doi.org/10.1016/j.comcom.2022.09.029.

[21] **Ferriol-Galmés, M.**, **Suárez-Varela, J.**, **Paillissé, J.**, **Shi, X.**, **Xiao, S.**, **Cheng, X.**, **Barlet-Ros, P. and Cabellos-Aparicio, A.** (2022). Building a Digital Twin for network optimization using Graph Neural Networks, *Computer Networks*, *217*, 109329, https://www.sciencedirect.com/science/article/pii/S1389128622003681.

[22] **Abdel-Basset, M.**, **Hawash, H.**, **Sallam, K.M.**, **Elgendi, I. and Munasinghe, K.** (2023). Digital Twin for Optimization of Slicing-Enabled Communication Networks: A Federated Graph Learning Approach, *IEEE Communications Magazine*, *61*(10), 100–106.

[23] **Wang, H.**, **Wu, Y.**, **Min, G. and Miao, W.** (2022). A Graph Neural Network-Based Digital Twin for Network Slicing Management, *IEEE Transactions on Industrial Informatics*, *18*(2), 1367–1376.

[24] **Shin, H.**, **Oh, S.**, **Isah, A.**, **Aliyu, I.**, **Park, J. and Kim, J.** (2023). Network Traffic Prediction Model in a Data-Driven Digital Twin Network Architecture, *Electronics*, *12*(18), https://www.mdpi.com/2079-9292/12/18/3957.

[25] **Hata, M.** (1980). Empirical formula for propagation loss in land mobile radio services, *IEEE Transactions on Vehicular Technology*, *29*(3), 317–325.

CURRICULUM VITAE

Name SURNAME: Buğra AYDIN

EDUCATION:

- **B.Sc.:** 2022, Istanbul Technical University, Faculty, Department of Computer Engineering
- **High School:** 2017, Giresun Science High School

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2022- Digital Transformation Office

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Aydin B.**, Oktug, S. F. (2025) Details of a Digital Twin for a LoRa Based Forest Fire Management System, *ITU JWCC*, 2(1), 27–36.
- **Aydin B.**, Oktug, S. F. (2024). Employing Digital Twin to Forest Fire Management Systems *2024 9th International Conference on Computer Science and Engineering (UBMK)*, October 26-28, 2024 Antalya, Turkey.

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- **Aydin B.**, Sari T.T., Oktug, S.F. (2023). Accelerating smart campus development with an extensible framework. *IEEE Potentials*, 42(4), 24-28.