electronics

Article

A Smart Control System for the Oil Industry Using Text-to-Speech Synthesis Based on IIoT

Ali Raheem Mandeel [1,*], Ammar Abdullah Aggar [2], Mohammed Salah Al-Radhi [1] and Tamás Gábor Csapó [1,*]

[1] Department of Telecommunications and Media Informatics, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics (BME), 1117 Budapest, Hungary; malradhi@tmit.bme.hu
[2] Ministry of Education, Baghdad 10082, Iraq; ammar3ag@gmail.com
[*] Correspondence: aliraheem.mandeel@edu.bme.hu (A.R.M.); csapot@tmit.bme.hu (T.G.C.)

Abstract: Oil refineries have high operating expenses and are often exposed to increased asset integrity risks and functional failure. Real-time monitoring of their operations has always been critical to ensuring safety and efficiency. We proposed a novel Industrial Internet of Things (IIoT) design that employs a text-to-speech synthesizer (TTS) based on neural networks to build an intelligent extension control system. We enhanced a TTS model to achieve high inference speed by employing HiFi-GAN V3 vocoder in the acoustic model FastSpeech 2. We experimented with our system on a low resources-embedded system in a real-time environment. Moreover, we customized the TTS model to generate two target speakers (female and male) using a small dataset. We performed an ablation analysis by conducting experiments to evaluate the performance of our design (IoT connectivity, memory usage, inference speed, and output speech quality). The results demonstrated that our system Real-Time Factor (RTF) is 6.4 (without deploying the cache mechanism, which is a technique to call the previously synthesized speech sentences in our system memory). Using the cache mechanism, our proposed model successfully runs on a low-resource computational device with real-time speed (RTF equals 0.16, 0.19, and 0.29 when the memory has 250, 500, and 1000 WAV files, respectively). Additionally, applying the cache mechanism has reduced memory usage percentage from 16.3% (for synthesizing a sentence of ten seconds) to 6.3%. Furthermore, according to the objective speech quality evaluation, our TTS model is superior to the baseline TTS model.

Keywords: industrial internet of things (IIoT); industry 4.0; speech synthesis; oil industry

check for updates

Citation: Mandeel, A.R.; Aggar, A.A.; Al-Radhi, M.S.; Csapó, T.G. A Smart Control System for the Oil Industry Using Text-to-Speech Synthesis Based on IIoT. *Electronics* **2023**, *12*, 3380. https://doi.org/10.3390/electronics12163380

Academic Editors: Ahmed Abu-Siada and Nikolay Hinov

Received: 25 June 2023
Revised: 3 August 2023
Accepted: 3 August 2023
Published: 8 August 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The development of the oil industry has witnessed a great leap and rapid shift from traditional production to intelligent machines. The Industrial Internet of Things (IIoT) size is rapidly growing in modern industrial processes. IIoT opens the door to a better acquaintance of the manufacturing process, allowing for more efficient and sustainable production [1]. The Fourth Industrial Revolution (Industry 4.0) presents digital technologies that potentially convert the entire industrial process. Simultaneously, this revolution has also influenced the oil industry [2]. These technologies would lower health, safety, and environmental risks. Additionally, it increases productivity and reduces operational and maintenance costs. Moreover, human-centered assistance systems help workers flexibly and efficiently perform their responsibilities. However, to provide optimal support, the systems must be context-sensitive regarding the present product condition and the information the laborer requires. IIoT has dramatically improved the availability of machine-to-machine interfaces allowing workers to gather, process, analyze, and visualize their IoT data [3].

The oil industry has grown in terms of the number of facilities and the complexity of the processes. Nevertheless, human and material casualties continue due to accidents involving human mistakes [4]. Oil refineries and petrochemical plants have a high rate of catastrophic accidents at their facilities, whereas only petrochemical and oil refining plants accounted for 67% of all significant accidents in Republic of Korea between 2005 and 2021 [5]. Since 1992, the oil refining industry has experienced many fatal and catastrophic incidents related to releasing highly hazardous chemicals in the U.S. [6]. The Chemical Safety Board recorded 125 serious process safety incidents in U.S. oil refineries in 2012 [7]. Human mistake is the main reason for over 90% of accidents in the oil and gas industries [8]. Although it is widely acknowledged that human mistakes cause significant incidents, a few primary hazard sites actively pursue possible human performance issues [9]. Furthermore, the economic impact of incidents in the oil industry is also significant. For instance, the financial losses from the oil refinery fire accident on Pulau Bukom island, Singapore, in 2011 were estimated from $143–222 million for the fast initial recovery scenario to $204–375 million for the slow initial recovery plan [10]. Therefore, building an efficient smart alarm will reduce the risk of laborers' mistakes at these sites.

Alarms are tools used to obtain crucial information in cognitively demanding circumstances. They have been used in hospitals and hazardous areas (oil industries) to warn workers about critical conditions. They are either auditory or visual alarms (flashing lights and bright colors). The acoustic method is superior to another approach because it offers faster response times and is noticeable in out-of-sight locations [11,12]. The acoustic structure of alarms is critical for their ability to communicate effectively. Notifications using voice alerts showed significantly higher proportions of identification (98%) with considerably more confidence and in a substantially shorter time than traditional state-of-the-art device alarms (nonspeech) [13].

Smart sound alarms using Text-to-Speech Synthesis (TTS) in the industry can help workers take faster action. It can be difficult for laborers to realize an alarm type for every possible emergency variant. Therefore, assistive alarm systems with IoT accessibility, which provide sound alarms, can support workers. Using TTS in the industrial process promises many benefits, such as workers not having to lose focus on multitasking simultaneously and allowing hands-free operation [14]. In an emergency, people often panic and block themselves, forget the tasks they have practiced many times, and react poorly to the sounds of different types of sirens. As a psychological impact, we can tell that human voices can convey emotions and create a sense of urgency. Using a human voice for alarms with direct guidance may help workers respond effectively to high-stress situations. These benefits may be exceptionally valuable to those with less experience. Furthermore, voice instructions can assist people with disorders such as cognitive impairments.

TTS aims to develop human-like speech signals from written text based on end-to-end techniques. These techniques produce high-quality synthesized speech in terms of naturalness and expressiveness. However, they usually fall under low-quality synthesized speech when the used dataset is inadequate. To solve these issues, frequently speaker adaptation is utilized. A pre-trained TTS model (trained on a large dataset) is adapted to a target speaker's smaller dataset. Similarly, we can customize our model to synthesize speech for any target speaker using a small dataset of target speakers.

The main challenge in implementing TTS models in real-world applications is the low inference speed of these models due to employing enormous neural networks with millions of parameters. Moreover, computational resources are available occasionally. Much research has been conducted on creating light TTS models with excellent synthesis quality [15]. These studies have been summarized to enhance acoustic models [16] (make acoustic representations from input text) and neural vocoders [17] (convert these representations to waveforms). However, different optimizations of the two models limit the execution of TTS systems [18]. Moreover, trade-offs exist for the computational cost, inference speech, and synthesized speech quality [19]. One essential solution for optimizing inference speed in low computational resources real-time is to use the cache mechanism, which uses recorded prompts with "slot filling" of variable data [20]. It also uses the stored synthesized sentences in memory to call when they are requested. Therefore, a TTS model can synthesize high-intelligibility speech at a sufficient speed.

Following our previous study [21], we built an intelligent alarm model using a modified TTS of an acoustic model (FastSpeech 2) with a neural vocoder (HiFi-GAN V3). We compared our TTS model inference speech speed with a baseline TTS model (FastSpeech 2 and MelGAN vocoder). In addition, we utilized speaker adaptation to customize the target speaker's speech using a small dataset of target speakers. We adapted pre-trained FastSpeech 2 to two target speakers (one female and one male). The TTS models were integrated into a single-board computer (Raspberry Pi). We performed a latency evaluation (real-time factor (RTF)) to measure system performance in a real-time environment. Our model targets the oil and gas industries, such as oil refineries. We demonstrated our system using data from industries such as gas sensors. Moreover, it can be extended to collect data such as temperature, pressure, flow rate, etc. The main contributions of this study can be summarized as follows:

1. We built a novel extension intelligent control system utilizing IIoT and a text-to-speech model for industrial purposes (particularly oil refineries).
2. We successfully designed a TTS model that can run on low computational resources: Raspberry Pi 4 (RPi).
3. Speaker adaptation was used to customize the synthesized speech for the two target speakers (male and female) using only a small adaptation dataset.
4. We proposed a real-time end-to-end TTS model using the cache mechanism technique to increase the efficiency of our design inspired by [20]. Our results reveal that TTS is no longer an issue in real-time applications.

The remainder of this paper is organized as follows. Section 2 outlines related studies on speaker adaptation, speech processing in Industry 4.0, and using IIoT in the oil and gas industry. After that, Section 3 describes the methods used to construct the system. Section 4 provides an overview of the findings of this paper. Finally, we present the conclusions of the study.

2. Related Studies

2.1. IIoT in Oil and Gas Industry

Supervisory control and data acquisition (SCADA) systems have been used in the oil industry to monitor processes over the last few decades [22]. However, SCADA systems have many drawbacks, such as the interoperability of the hardware and software and low-density time and space. Moreover, it shows high-cost maintenance and difficulties in updating communication protocols [23,24]. Furthermore, wireless sensor networks (WSN) have been employed in the oil industry for monitoring and data collection [25]. Nevertheless, they are not uniform systems and frequently require more collaborative communication [22].

Consequently, IoT has addressed the constraints of SCADA and conventional WSNs [26]. IoT solution makes quicker and more accurate decisions using the data gathered by IoT sensors because it reduces latency and improves information access [27]. The oil industry uses the IIoT to improve mainly resource performance and reduce environmental, health, and safety threats. Many researchers have investigated IIoT in oil industry services, such as monitoring tank liquid levels [28], corrosion [29], pipelines [30], and leakage detection [31]. Data collected by sensors-based IoT can be used to make a maintenance prediction where data are stored in the cloud for analysis by algorithms. IoT technology in industries has many advantages, including decreased test time and calibration, lower warranty expenses, reduced production shutdown, predictive maintenance efficiency, and improved supply chain performance [32]. Table 1 lists a few of the most related works in IIoT of the oil and gas industry.

Table 1. The most related work in IIoT of the oil and gas industry.

Ref.	Contribution	Techniques	Results
[33]	Enhance oil industry business interactions	- Computing-aided software-defined networking (SDN) - Emergent configuration (EC)	EC and SDN-based IIoT improve the oil extraction procedures
[34]	Reducing fire accidents by using IIoT	The copula theory to integrate the sensors	It surpasses two baseline methods in fire detection
[35]	Data management for IIoT systems.	Machine learning techniques	IIoT is used for tracking the oil and gas industry operations flow
[31]	Leak detection in oil pipelines utilizing IIoT	- 2D CNN - LSTM AE	Efficient wireless sensor system for leakage detection in the metallic pipeline
[36]	Leak detection in oil pipelines utilizing IIoT	Hybrid localization and distributed leakage detection techniques	- Eliminates single point of failure - Increase sensitivity to leakage detection
[37]	Risk oil pipelines evaluation without failure data	- Information fusion theory - Fuzzy set theory	Control risks and avoid leakage consequences
[38]	Spilled oil diffusion underwater distance prediction	- Computational fluid dynamics - Volume-of-fluid technique	Provide an emergency treatment plan for offshore oil spill accidents

In conclusion, the oil and gas industry-based IIoT research papers mainly collect, analyze, and manage data and have not been studied to incorporate a smart alarm. Therefore, creating an extended intelligent IIoT system that uses a TTS model-based neural network to notify staff in oil refineries' control rooms during emergencies is essential. It will utilize the IIoT benefits of covering a wide area, reducing latency, and improving information access to reduce workers' health risks in the hazardous fields of oil refineries.

2.2. *Converging Speech Processing in Industry 4.0*

Speech-based interactions between humans and machines are becoming increasingly prevalent in 5G IoT scenarios. Many industries (IoT, security, healthcare, and computer control) use speech control commands rather than written instructions [39]. Speech assistance technology should have low latency, noise tolerance, high accuracy scalability, and reliability to deliver practical solutions [39].

An assistive system-based TTS model for generating spoken instructions was designed to assist workers in manufacturing [14]. The study demonstrated that all employees finished assembling tasks after listening to synthesized spoken commands faster than reading written documents. Emo-VITS, a VITS-based emotional speech synthesis model, has been developed to suit the IoT sufficiently [40]. The subjective and objective evaluations demonstrate that the Emo-VITS model provides significant outcomes in representing emotions.

A safety method based on voice command-based Automatic Speech Recognition (ASR) and Q-learning algorithm was proposed for small planets robots [41]. The workers' voice commands were sent to the plant control room as an emergency signal. Furthermore, the model was tested on an S7-1200 Siemens programmable logic controller (PLC) [41]. To help operators in the manufacturing industry control robots without requiring advanced skills, robots were integrated with a depth camera (Microsoft Kinect version 2) and an inertial measurement unit to capture both operators' gestures and speech commands [42]. This technique saves operators from having to devote ample training time to master these robots. Table 2 summarizes a few papers on using speech in industry and shows their limitations.

Table 2. A summary of some related work papers on using speech in Industry 4.0.

Ref.	Techniques	Contribution	Limitation
[14]	TTS model	Using a TTS model to help workers in the assembly industries.	It did not customize the target speakers.
[40]	TTS model	Developed Emo-VITS	It was not tested in a real IoT application.
[41]	ASR model	A safety method to stop small plants working in emergencies.	It has yet to benefit from IoT to extend the coverage.
[42]	ASR model	Control robots with a user's voice	It needs to address incorrect human instructions.
[43]	ASR model	Using ASR in the underground mining industry	The average accuracy of form completion was 70–80% which may be insufficient for the critical tasks.
[44]	ASR model	ASR & gestures recognizer to control an industrial robot	Environmental constraints: noise, lighting conditions, and moving objects.
[45]	TTS model	Digital-twin human-machine interface sensors enable the control of digital devices using gestures. Speech synthesis feedback expresses the operational requirements of inconvenient work.	Validations of real-world scenarios were excluded.
[46]	ASR model	It mixed reality and speech interactions to improve aircraft maintenance.	System latency and response time were overlooked.

Overall, TTS models need to be investigated further in Industry 4.0 based on IIoT. It has been used in many industrial fields, such as healthcare [47], education [48], and entertainment [49], but it has yet to be invested in the oil industry. Therefore, we implemented state-of-the-art text-to-speech synthesis in our proposed design.

2.3. Speaker Adaptation

Speaker adaptation (sometimes called speaker cloning or customizing) is the process of adaptation of a TTS model to synthesize the speech of a target speaker in a small dataset. It is usually done by training a TTS model on a large dataset (occasionally of a multi-speaker dataset) and then fine-tuning it to a small target speaker's dataset. Neural end-to-end TTS models need to train on at least ten hours of a high-quality dataset to mimic the speaker characteristics (acoustic features and prosody) [50]. It might be an optimal solution for the languages with a few datasets (e.g., Arabic or Bengali), i.e., training a TTS model on a large English dataset and adapting it to Arabic. This process is called a cross-lingual TTS adaptation. Correspondingly, speaker adaptation reduces the cost and time by creating a dataset for some languages. It offers adequate network parameters for training the model in a low computational resources CPU.

In recent decades, speaker adaptation has been a widely studied subject. Mainly, the research focuses on reducing the target speaker's dataset, increasing the similarity to the target speaker, enhancing the synthetic speech quality (intelligibility and naturalness), and improving expressive speech. Investigations showed that the continuous vocoder is reliable for synthesizing target individuals' speech when used with an RNN (recurrent neural network)-DNN (deep neural network)-based TTS system [51,52]. Zero-shot attempts to customize target speech from an unseen target speaker's speech by extracting a speaker embedding from the original target speaker's dataset without using parameters. Investigations indicate improved speaker similarity and demonstrate that the neural all-pass warp (APW) using Tacotron2 (encoder–decoder architecture) raises the generalizability of a multi-speaker model with a zero-shot speaker adaptation [53]. However, the zero-shot method usually suffers from inadequate speaker similarity.

With only a few shots of data, a suggested light Transformer-based postfilter architecture enhanced speech quality and effectively adapted to new target speakers [54]. This study investigated and adapted many postfilter architectures with minimal data. Using the TTS model (Tacotron2), it was found that five minutes of the target speaker's adaptation data with a low training time of checkpoint 900 (an iteration point in the training process) is enough to have a reasonable synthesized speech quality [55]. Moreover, a meta-learning algorithm was applied to the speaker adaptation method to increase the target speaker similarity and decrease the adaptation data [56].

In conclusion, we considered speaker adaptation in our TTS model to overcome the issues above (dataset availability and costs). We adapted our TTS model to two English target speakers.

3. Methods

The overall system structure is depicted in Figure 1. The sensors send measured data value signals (e.g., liquid temperature, pressure, flow, level, gas percentage, etc.) to the RPi in each industry field. Then, RPi executes these data. After that, it carries these data to the cloud for central control. There are two alarm devices in our system design. Local alarm in the field and the second alarm in the main control room. The TTS models have been implemented in RPi. We adapted our TTS model to two target speakers (female and male). After that, we experimented with the cache mechanism. The following sections will explain the hardware design, IoT structure, and TTS model.

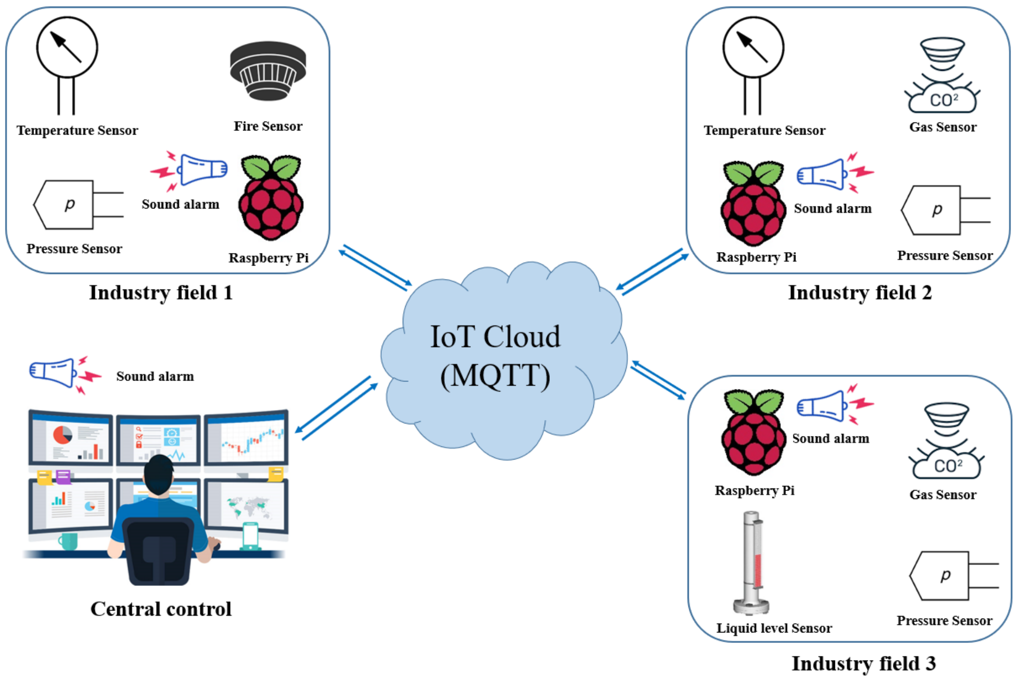


Figure 1. The overall system structure.

3.1. Hardware and IoT System Design

The hardware and IoT setup are explained in Figure 2. The sensors are used to measure the data, and a microcontroller collects the data from these sensors. For example, we used a gas sensor (MQ-2) to detect a spectrum of gas (such as propane, hydrogen, and liquefied petroleum gas (LPG)) in the air. After that, the microcontroller sends the data to XBee. RPi receives the data and processes it. Later, RPi uses the MQTT (Message Queuing Telemetry Transport) to publish it in the MQTT broker. In the following subsection, we will explain the process of this structure in detail.

3.1.1. XBee

XBee is a radio frequency (RF) module providing wireless connectivity to the system design. It utilizes the ZigBee protocol, a wireless communication technology [57]. ZigBee protocol allows XBee to connect with other devices in the network using low power. It has flexibility for the developers of IoT applications and industrial automation. XBee module has many versions, such as S1, S2, etc.

Sometimes XBee modules can be embedded in the data-gathering circuit without combining them with an external microcontroller. The XBee module can collect data directly from sensors through analog and digital input pins. This method has benefits in reducing power consumption, saving board space, cost savings, and reducing the board's weight. The independent applications of the XBee model face some limitations in more complicated projects. It lacks analog and pulse width modulation (PWM) outputs needed to operate a wide range of devices. Moreover, a restricted number of output and input pins are available in the XBee module [58].

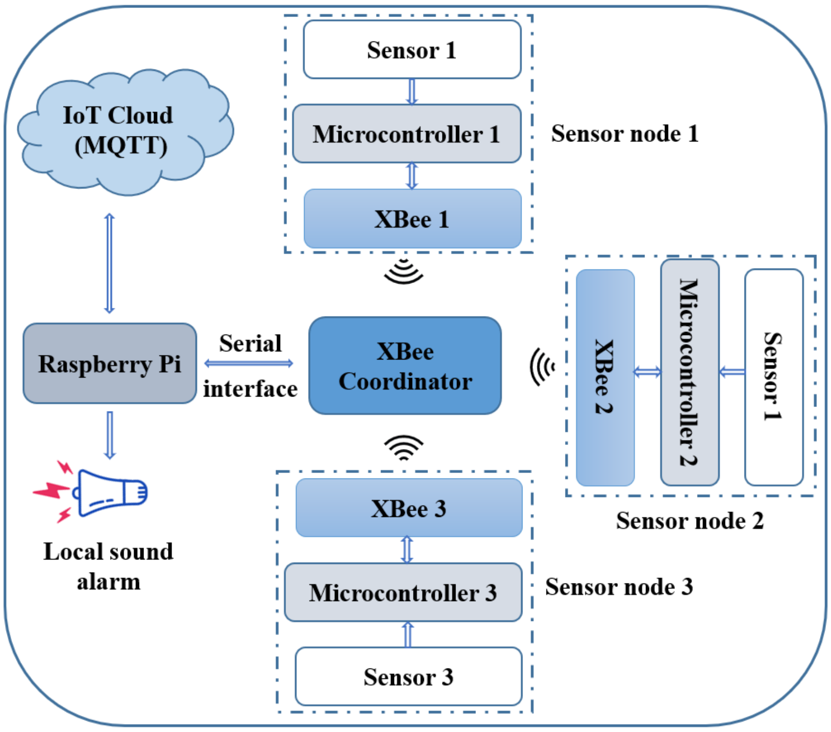


Figure 2. The hardware structure.

In our paper, S1 has been used to manage data transfer between sensor nodes distributed in different places and the Raspberry Pi. We configured the XBee module that connected sensors as End Devices. The End Device XBee module receives the data from a sensor through a microcontroller (Arduino) by a serial interface. After that, it sends these data wirelessly to another XBee module called Coordinator XBee. Coordinator XBee receives the data from all the End Device XBee modules simultaneously and sends the data to the RPi through a serial interface. In oil refineries, most sensor nodes are deployed over large areas. In our paper, we adopted XBee instead of Bluetooth, WiFi, WiMAX, and mobile communication because XBee has features that make it compatible with oil refineries. XBee provides easy-to-implement solutions [59], low power consumption and cost [33], and transfers data for long distances compared to Bluetooth [60]. Table 3 compares ZigBee to other communication technologies regarding power consumption and cost.

Table 3. A comparison of ZigBee to the communication technologies regarding power consumption and cost [33].

Communication Technology	Transmission Range	Power Consumption	Cost
ZigBee	**10–100 m**	**Low**	**Low**
Bluetooth	8–10 m	Low	Low
WiFi	20–100 m	High	High
WiMAX	Less than 50 Km	Medium	High
Mobile Communication	Entire Cellular Region	Medium	Medium

We configured XBee module functions (Coordinator or End Device) using XCTU software. We set Coordinator Enable (CE) in this manner:

CE: Coordinator Enable (to be a coordinator module)
CE: End Devices (to be an end device module)

Regarding the measured data, we used the MQ2 gas sensor to demonstrate our proposed system work. The sensitivity of the MQ2 gas sensor depends on the gas types and their concentrations. We set the sensor calibration range to 1000 ppm for LPG and iso-butane, as recommended in the MQ2 datasheet. Arduino microcontroller reads analog data from the MQ2 gas sensor connected to pin A0. After that, the Arduino's analog-to-digital converter (10-bit resolution) represents these values by digital readings (0–1023). The microcontroller converts the raw sensor value to the corresponding voltage (with a 5 V reference that the Arduino board uses). In our paper, we set the threshold to a value of 1 and then compared it to the readings from the gas sensor.

3.1.2. Raspberry Pi

The RPi Foundation created a series of compact, single-board computers known as RPi. The original RPi was introduced in 2012, and the series has since expanded to encompass other models and variations. The RPi is designed to be a relatively low-cost, credit-card-sized computer that can be used for various purposes, including education and commercial applications. Some common uses for RPi include building media centers, home automation systems, retro gaming consoles, and even robotics projects. It runs on a Linux-based operating system and can be programmed using a variety of programming languages, including Python, C, and Java.

In this paper, RPi receives data from sensor nodes via Coordinator XBee. RPi compares the received data to a threshold value (that was previously set). The sound alarm runs using the TTS model or the cache mechanism if the data weight exceeds or equals the threshold. At the same time, the RPi forwards the data, which achieves the threshold condition, to another RPi in the central control via MQTT protocol using the HiveMQ broker. The alarm message will be converted to a sound alarm by the RPi there.

3.1.3. MQTT and HiveMQ

MQTT (Message Queuing Telemetry Transport) is a popular messaging protocol in the IoT industry due to its efficiency and low bandwidth usage [61,62]. It is an effective protocol for small, inexpensive, low-power, and limited-memory devices [33]. It utilizes a broker (such as HiveMQ) to control the transmission between machines, where transmitting machines publish messages on a particular topic. Other devices can subscribe to that topic to obtain these messages. HiveMQ is a leading MQTT broker that offers a scalable and robust solution for connecting devices and applications in real time. This broker is designed to handle millions of concurrent connections while providing high availability and reliability [63]. HiveMQ MQTT over WebSockets, MQTT over SSL/TLS, and MQTT 5.0 specification support set it apart from other brokers. Moreover, the HiveMQ plug-in and integration capabilities further enhance its functionality. For example, the dashboard is a web-based visualization tool that provides real-time monitoring and visualization of data traffic. The plug-in SDK also enables developers to customize and extend HiveMQ's capabilities.

It is essential to ensure proper security measures to protect the data and ensure the integrity of communication. The Secure Sockets Layer (SSL)/Transport Layer Security (TLS) protocol is implemented in our design on top of MQTT for transport encryption (encoding data) to secure client and broker connections. SSL/TLS is a cryptographic protocol that guarantees data transfer between applications on the Internet [64]. SSL/TLS is suitable for IoT and is designed to work with TCP. However, the SSL/TLS protocol has disadvantages such as high energy consumption, system complexity, and overhead [64]. Moreover, the MQTT protocol provides a client identifier and username/password credentials to authenticate devices at the application level.

The development in hardware technologies has made IoT services rapidly grow due to the extreme increase in IoT nodes that led to making the scalability of IoT more difficult. This paper uses the HiveMQ MQTT Platform to solve this issue, which can achieve 200 million concurrent connections [65]. HiveMQ is executed in Java and is now obtainable as professional, community, and enterprise editions in addition to an IoT cloud platform alternative with hourly subscription costs. The HiveMQ DNS detection plug-in uses DNS service discovery to remove or add broker cases to the cluster at runtime [66].

For all these reasons, HiveMQ broker and MQTT protocol have been used in our proposed system to transfer data between the field and central control (the supervisor side). When the connection is secured on the field side, the data will be sent to the supervisor via the MQTT protocol. Otherwise, the data will not be transmitted to the central control in the case of failed authentication or if the data are less than the threshold values. The received data in central control will be processed by the TTS model or the cache mechanism and passed to the alarm.

3.2. Text-to-Speech Model Architecture

We applied the two TTS models on the Raspberry Pi. The modified TTS model consists of the acoustic model FastSpeech 2 and neural vocoder HiFi-GAN V3. Conversely, the baseline TTS model has FastSpeech 2 and neural vocoder MelGAN.

3.2.1. End-to-End TTS: FastSpeech 2

FastSpeech 2 is a TTS model that converts the input text to mel-spectrograms [67]. Unlike the old version FastSpeech which uses the simplified output from the teacher, it is trained on the ground-truth target. Moreover, it offers many speech variations (such as pitch, energy, and duration) to solve the one-to-many mapping issue (one input text and many output speech differences). Moreover, it solves the slowness of the autoregressive models (e.g., Tacotron2), which sequentially generate each acoustic frame for a speech waveform. This parallelism technique enhances both training and inference speech at the same time. Nevertheless, FastSpeech 2 remains large (27 M parameters), resulting in high memory consumption and inference latency when implemented on limited computing resources devices [16].

FastSpeech 2 structure consists of an encoder, variance adaptor, and decoder. The encoder converts phoneme embedding to a hidden sequence. After that, the variance adaptor adds speech variations (pitch, duration, and energy) to this hidden sequence. Finally, the decoder transforms this adapted sequence with speech variations to mel-spectrogram.

We used the same FastSpeech 2 structure, which has four feed-forward Transformer blocks in the encoder and decoder [67]. Also, we employed two attention heads and kernel sizes of 9 and 1. The output mel-spectrograms of the encoder are 80-dimension. The dropout rate of the encoder and decoder is 0.5. Conversely, the variance predictor filter size is 256, the kernel size: is 3 and the dropout rate: is 0.5.

According to the ablation analysis of [16], the encoder and decoder consume the majority (11.56 M parameters) of the model size and inference time. The predictors (duration predictor: 0.40 M, pitch predictor: 1.64 M, and energy predictor: 1.64 M parameters) account for roughly one third of the total size and inference time.

3.2.2. Neural Vocoders: HiFi-GAN and MelGAN

HiFi-GAN converts the output mel-spectrogram produced by FastSpeech 2 to a waveform [68]. This vocoder is based on a generative adversarial network. Our paper used HiFi-GAN because it performs more elevated speech quality and computational efficiency than other autoregressive vocoders (e.g., WaveNet and WaveGlow) [68].

HiFi-GAN has one generator and two discriminators (see Figure 1 in [68]). The generator is a fully convolutional neural network. It upsamples the input mel-spectrogram by transposed convolutions to match the length of raw waveforms. After each transposed convolution, the multi-receptive field fusion (MRF) produces the total of all residual block outcomes. The residual block has diverse kernel sizes and dilation rates to deliver different receptive field types. On the other side, the discriminator is a classifier that attempts to determine actual audio signals from the data synthesized by the generator.

We can adjust some parameters in the transposed convolutions and MRF modules of the HiFi-GAN generator to have a trade-off between synthesis efficiency and sample quality. For instance, we can set the transposed convolutions' hidden dimensions and kernel sizes. In the same way, kernel sizes, and dilation rates are controllable in MRF modules.

In our paper, we accomplished experiments on the two types of generators, V1 and V3, while keeping the same discriminator configuration. HiFi-GAN V1 has 13.92 M parameters, and HiFi-GAN V3 needs 1.46 M parameters. Table 4 shows the hyperparameters for the two generator variants suggested by [68]. Compared to V1, V3 has a significantly lower number of layers.

Table 4. The hyperparameters for the two generator versions (V1 and V3).

Model	Hidden Dimension	Transposed Convolutions Kernel Sizes	Kernel Sizes	Dilation Rates
V1	512	[16, 16, 4, 4]	[3, 7, 11]	[[1, 1], [3, 1], [5, 1]] × 3
V3	256	[16, 16, 8]	[3, 5, 7]	[[1], [2]], [[2], [6]], [[3], [12]]

On the other hand, MelGAN is also a non-autoregressive generative model that uses GANs for speech synthesis [69]. Also, it is a feed-forward convolutional architecture that consists of a generator and a multi-scale architecture discriminator. To upsample the input mel-spectrogram, a stack of transposed convolutional layers is used. A stack of residual blocks with dilated convolutions is on the top of transposed convolutional layers. Three discriminators have similar network architecture while operating on various audio scales. We used it to convert the input mel-spectrograms from FastSpeech 2 to the waveform. Moreover, it has parameters (4.26 million) more than HiFi-GAN V3 neural vocoder (1.46 million).

3.3. Speech Corpora

We trained the acoustic model (FastSpeech 2) on the TTS LJSpeech dataset [70]. Additionally, we used the pre-trained neural vocoders HiFi-GAN and MelGAN on the same dataset. This dataset has 13,100 pairs of text and audio (a single-channel 16-bit PCM WAV). The entire duration of the speech is 24 h. The clips' length is between one to ten seconds, with a sample rate of 22,050 Hz. Additionally, we utilized Hi-Fi multi-speaker English TTS dataset [71]. It has a speech of 292 h sampled at 44.1 kHz in WAV format. Later, we resampled it to 22,050 Hz. A group of ten speakers, six of them females, participated in making the speech of this dataset. We determined two target speakers (Tony Oliva and Helen Taylor) for the speaker adaptation process. We aligned the phonemes to the texts for the Hi-Fi dataset using the Montreal Forced Aligner (MFA) trained with English US ARPA [72]. According to [67], MFA enhances the alignment accuracy and decreases the information gap between the model input and output. Also, MFA provides a more accurate duration than the teacher model of FastSpeech.

3.4. Training Topology and Speaker Adaptation

We trained FastSpeech 2 using NVidia Titan X GPU on the LJSpeech dataset. The following hyperparameters were used in training: batch size: 16 sentences, Adam optimizer, β_1 = 0.9, β_2 = 0.98, total step: 900 k. We used 96% of the data for the training set and the rest for validation.

After that, we adapted FastSpeech 2 with two English target speakers (female and male). For the female target speaker, 580 sentences totaling 32 min were randomly selected. On the other hand, 840 sentences (47 min) were chosen for the male speaker. We used the same training hyperparameter to train FastSpeech 2 on the large dataset. Moreover, we used 118.5 k steps and divided the data by 90%, and the remaining data were used for the validation set.

3.5. Cache Mechanism Implementation

The decisive contribution of this paper is the cache mechanism. Real-time is vital in IoT to make the best decision about a specific event. Here comes the role of the cache mechanism. In our proposed design, the data are collected from the sensors and passed through for comparison with the threshold values. If it is more significant or equal, it is passed to the TTS model and passed to the sound alarm. The most time-consuming part of our design is the speech synthesis model because current neural speech synthesizers provide high naturalness but at a high computational cost. Therefore, we used the cache mechanism to reduce the time spent in the speech synthesis by storing the data from the initial reading in a temporary stack. The input data are compared to previous data stored in a temporary stack. If it equals the stored data, the sound alarm is turned on depending on the previously stored data without passing the new data through the TTS model. If we read new data not in the temporary stack, it will be passed to the TTS model and kept in the temporary stack. This process is used only once with new data. By cache mechanism, we reduce the time needed for each further reading, making our IoT project compatible with running in real time. Figure 3 shows the sequence of the cache mechanism in our study.

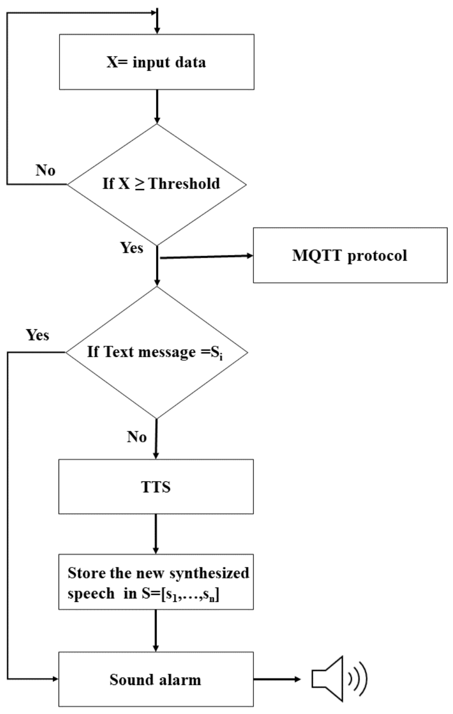


Figure 3. The cache mechanism sequence in our design.

We summarize the general operations of the proposed system in Algorithms 1 and 2. The abbreviations are used in Algorithms 1 and 2 explained before references:

Algorithm 1 Field-Side

```
INPUT: $X = \{x_1, \ldots, x_n\}$; $S = \{s_1, \ldots, s_n\}$; T; TTS; A; U; P; $H_s$; $M_t$; R; Hi; $M_p$;
1:  Begin
2:    while $x_i \neq 0$ do                          // waiting for input data
3:      if $x_i \geq T$                              // Compare input data to the threshold (T)
4:        Generate "Text message"                    // assign text message for this sensor condition
5:        for "Text message" in S do
6:          if "Text message" = $s_i$ then go to Continue:     // Use the cache mechanism
7:          else
8:            TTS ← "Text message"                   // Initialize the TTS model
9:            $s_i$ ← TTS                            // Store the new synthesized speech in the stack
10:         end if
11:       end for
12:     else
13:       No abnormal detected
14:     end if
15:   end while
16:   Continue:
17:   A←$s_i$                                        // Speak the alarm message
18:   if (U, P, $H_s$) = $H_i$ then                  // Secured MQTT connection
19:     Mp ←($x_i$, $M_t$, U, P, $H_s$, R)
20:   else
21:     No connection to the MQTT protocol           // Authentication fail
22:   end if
23: End
```

Algorithm 2 Supervisor Side

```
INPUT: $D = \{d_1, \ldots, d_n\}$; $S = \{s_1, \ldots, s_n\}$; TTS; A; U; P; $H_s$; $M_t$; R; Hi; $M_p$;
1:  Begin
2:    if (U, P, $H_s$) = $H_i$ then                  // Secured MQTT connection
3:        Mp ←($x_i$, $M_t$, U, P, $H_s$, R)
4:    else
5:        No connection to the MQTT protocol         // Authentication fail
6:    end if
7:    while $d_i \neq 0$ do                          // waiting for input data
8:      Generate "Text message"                      // assign text message for this sensor condition
9:      for "Text message" in S do
10:       if "Text message" = $s_i$ then go to Continue:       // Use the cache mechanism
11:       else
12:         TTS ← "Text message"                     // Initialize the TTS model
13:         $s_i$ ← TTS                              // Store the new synthesized speech in the stack
14:       end if
15:     end for
16:    end while
17:   Continue:
18:   A←$s_i$                                        // Speak the alarm message
19: End
```

Algorithm 1 represents the field side where the RPi reads the data (X) from the sensors. After that, it compares the input data to the threshold value (T). RPi sends X to the central control when X exceeds or equals T. Moreover, it uses either the TTS model or the cache mechanism to play the sound alarm (A). Authentication is required on both sides to initialize the MQTT protocol communication.

In the proposed smart control system for the oil industry, the agents in Algorithm 1 are initialized with specific variable values to ensure efficient and safe monitoring of oil refineries. As explained in the paper, the initialization process involves setting the values of various parameters, including X, S, T, TTS, A, U, P, H_s, M_t, H_i, M_p, and R. For instance, X is initialized to 0, which represents the initial input data of the system. S is also set to 0, indicating that no data have been stored in the cache memory yet. T is initialized to 1, which is the threshold value used to compare input data. TTS is set to FastSpeech2 and HiFi-GAN V3, which are the TTS synthesis models used in the proposed system. A is initialized to "No alarm message", which is the default message displayed when no alarm is triggered. Also, the system requires authentication for accessing the MQTT account, which is achieved by setting the values of U, P, H_s, M_t and H_i. M_p is set to MQTT protocol, which is used for communication between the system and the MQTT broker. Finally, R is set to 1883, which is the default port number for MQTT communication.

Overall, initializing the agents in Algorithm 1 is a crucial step in ensuring the proper functioning of the smart control system for the oil industry. The system can monitor and control oil refineries in real time by setting the appropriate variable values, ensuring safety and efficiency.

On the other side, Algorithm 2 manages the central control. First, the connection should be secured. After that, RPi checks for the coming data value (D) from the side, which is greater or equal to the threshold value. Similarly, it creates the sound alarm by the TTS model or the cache mechanism. Furthermore, we initialized the agents in Algorithm 2 as follows: D = 0; S = 0; TTS = FastSpeech 2 and HiFi-GAN V3; (U; P; H_s; M_t; Hi) = MQTT account authentications; M_p = MQTT protocol; R = 1883.

4. Results

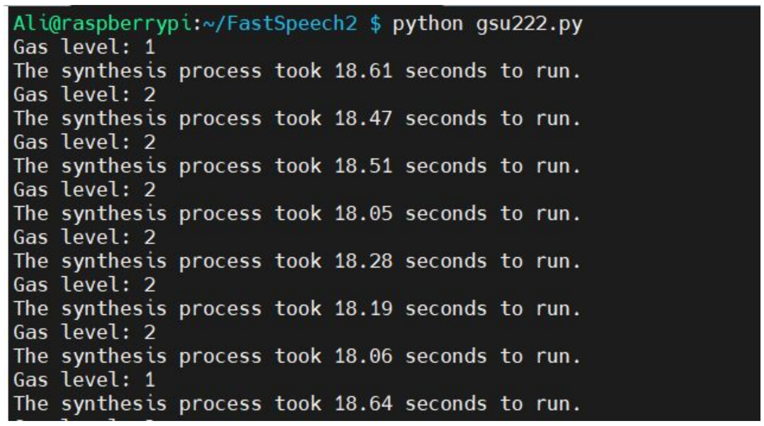
We conducted ablation analysis by accomplishing many experiments to evaluate the performance of our design (IoT connectivity, inference speed, complexity analysis, memory usage, and output speech quality). In our experiment evaluation, we collected data from the gas sensor. Subsequently, we evaluated the naturalness of the synthesized speech by measuring several objective metrics.

4.1. IoT System Design Efficiency

One of the essential aspects that must be taken into account in our IIoT design is to secure communication between the sensor nodes that are distributed in the oil field and the central control to guarantee error-free data transmission. In the experiment, real data were read from the gas sensor node (MQ-2) on the oil field side and transferred to the control room via the MQTT protocol. The experimental result shows that our system needs almost 18 s to receive the data from the field, compare it to the threshold, and synthesize one word using the TTS model, shown in Figure 4. These data represent a value greater or equal to the threshold value of the gas level in the oil field (which means an emergency condition has occurred). In Figure 4, the measured data (gas level) are either 1 or 2, achieving our threshold condition. In the experiment, we prove the connectivity between both sides is stable by testing real real-time data gathered from the gas sensor.

4.2. Real-Time System Evaluation and Runtime Analysis

Using the two TTS models, we measured the Real-Time Factor (RTF) of the inference speed on the RPi. Several waveform lengths have been employed to assess the computational performance of the TTS models. The utterances have been generated without batching. We calculate RTF by dividing the time required for synthesizing a sentence by the total duration of the same original sentence. The test set contains ten utterances (samples) with waveform lengths ranging from 1.0 to 10.0 s. For each sentence, we repeated the examination three times (R1, R2, and R3) and obtained the average running time of three attempts. Table 5 shows the RTF of the modified TTS model versus the baseline TTS model. Our proposed modified TTS model has a faster inference speech speed than the baseline TTS model by receiving RTF 6.4, while the baseline model obtained 7.5. A pairwise t-test was used to assess the statistical significance between the averages of the two systems. With $p > 0.05$, the two systems were significantly different. Moreover, it outperforms our earlier study results on RPi with the TTS model of FastSpeech 2 and HiFi-GAN V1, which obtained RTF 8.93 [21].



```
Ali@raspberrypi:~/FastSpeech2 $ python gsu222.py
Gas level: 1
The synthesis process took 18.61 seconds to run.
Gas level: 2
The synthesis process took 18.47 seconds to run.
Gas level: 2
The synthesis process took 18.51 seconds to run.
Gas level: 2
The synthesis process took 18.05 seconds to run.
Gas level: 2
The synthesis process took 18.28 seconds to run.
Gas level: 2
The synthesis process took 18.19 seconds to run.
Gas level: 2
The synthesis process took 18.06 seconds to run.
Gas level: 1
The synthesis process took 18.64 seconds to run.
```

Figure 4. The IoT connectivity test.

Table 5. The RTF of the two TTS models.

Sentence Duration	**FastSpeech 2 + HiFi-GAN V3**					**FastSpeech 2 + MelGAN**				
	R1	**R2**	**R3**	**Avg. sec**	**RTF**	**R1**	**R2**	**R3**	**Avg. sec**	**RTF**
1	20.2	20.3	20.4	20.3	**6.4**	23.0	22.8	23.4	23.1	7.5
2	21.2	21.9	21.3	21.4		25.5	27.1	25.5	26.0	
3	21.7	21.7	21.7	21.7		24.8	25.4	25.6	25.2	
4	22.6	22.5	22.8	22.7		25.6	26.1	26.5	26.1	
5	22.1	22.0	22.0	22.0		25.4	26.8	25.1	25.8	
6	22.9	23.1	22.9	23.0		26.4	27.5	28.1	27.3	
7	23.9	23.7	23.8	23.8		28.6	29.8	28.3	28.9	
8	24.4	24.3	24.3	24.3		31.6	31.5	28.2	30.4	
9	24.9	24.9	25.1	25.0		31.3	30.2	29.2	30.2	
10	25.2	24.8	24.9	24.9		33.8	29.8	39.9	34.5	

After that, we tested the cache mechanism in the system with the same ten sentences. We stored 250, 500, and 1000 sentences in the temporary stack (memory) to test the speed at different memory sizes. Our system compares the input data with the data stored in the stack. If they are identical, then it plays the sound alarm. We repeated the experiment three times (R1, R2, and R3) for each memory size. Table 6 shows the results of our investigation. The RTF of the system was 0.16, 0.19, and 0.29 when the memory had 250, 500, and 1000 stored audio files, respectively.

Table 6. The system RTF after applying the cache mechanism.

No. of Stored Samples in the Memory	**R1 sec**	**R2 sec**	**R3 sec**	**Average Time sec**	**RTF**
250	0.58	0.62	0.65	0.62	**0.16**
500	0.71	0.76	0.85	0.77	**0.19**
1000	0.89	1.04	1.08	1.00	**0.29**

4.3. Complexity Analysis

Algorithm analysis predicts the resources an algorithm will require, such as memory, communication bandwidth, or computer hardware. It estimates the algorithms' performance and efficiency features [73]. The complexity calculation of Algorithm 1:

- INPUT: $X = \{x_1, \ldots, x_n\}$; $S = \{s_1, \ldots, s_n\}$; TTS; A; U; P; H_s; M_t; R; Hi; M_p: O(1)
- While the reading data from the sensor node (X) is not null, go to the next step: O(1)
- If input data (x_i) $\geq$ the threshold (T), then go to the next step: O(1)
- Check if "Text message" exists in the list of previously stored values: O(1)
- If "Text message" is equal to saved synthesized speech (S), then go to the alarm step: O(1)
- Synthesize the input text using the TTS model: O(n)
- Store the data in the stack: O(1)
- Alarm step: O(1)
- MQTT step: secured MQTT connection: O(1)
- Send data to HiveMQ: O(1)

From this analysis, the computational complexity of Algorithm 1 is O(n) without using the cache mechanism, and it is O(1) using the cache mechanism. The complexity calculation of Algorithm 2 is explained below:

- INPUT: $D = \{d_1, \ldots, d_n\}$; $S = \{s_1, \ldots, s_n\}$; TTS; A; U; P; H_s; M_t; R; Hi; M_p: O(1)
- Secured MQTT connection: O(1)
- Check if "Text message" exists in the list of previously stored values: O(1)
- If "Text message" = saved synthesized speech (S), then go to the alarm step: O(1)
- Synthesize the input text using the TTS model: O(n)
- Store the data in the stack: O(1)
- Alarm step: O(1)

Similarly, the computational complexity of Algorithm 2 is also linear O(n) without using the cache mechanism. In contrast, it is constant O(1) using the cache mechanism.

4.4. Memory Experiment

Inherent power and memory size limitations in the IoT domain restrict its functionality in safely transmitting sensitive data [74]. In this paper, a cache mechanism has been utilized to reduce RPi memory usage. The synthesized speech sentence is stored once in the memory and will be used next time without storing it. On the other hand, we measured the RPi random access memory (RAM) consumption in three cases. The first case has been measured without processing or waiting for new data. The second scenario uses the cache mechanism in the model to produce ten sentences (from one second to ten seconds). The last experiment was measured using the TTS model to synthesize the same ten sentences. The experimental result shows that RAM usage is 5.3% of the total memory in the first case (waiting for new data), as shown in sample 0 (no produced sentence) in Figure 5. In our model, 6.3% of the RAM was used to create ten sentences in a row utilizing the cache mechanism. Finally, 11.7–16.3% was the RAM percentage deployed to synthesize the ten sentences without the cache mechanism.

4.5. Loss Measurement

To analyze the speaker adaptation training process, we plot the total loss and mel-spectrogram loss curves of FastSpeech 2 on the training and validation set of the male target speaker (Figures 6 and 7). The total variant loss was 4.269/4.555 at the training initialization and degraded to 0.739/0.738 on step 1.85 M for training and validation, respectively. Furthermore, the Mel Loss variant was 1.004/1.094 and 0.349/0.364 for training and validation, respectively. The gap between each loss training and validation loss curve indicates an improved generalization.

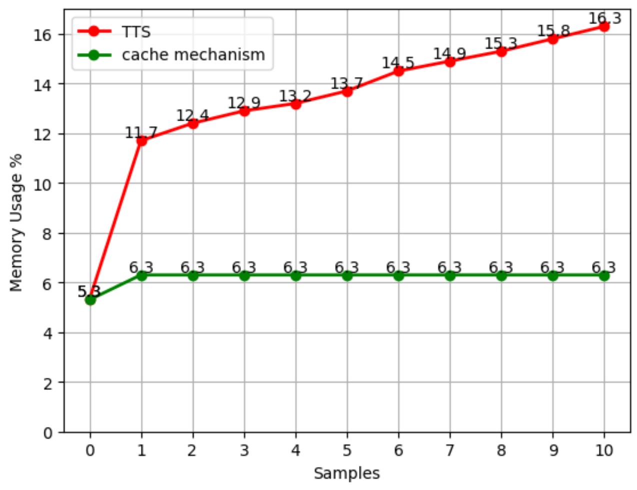
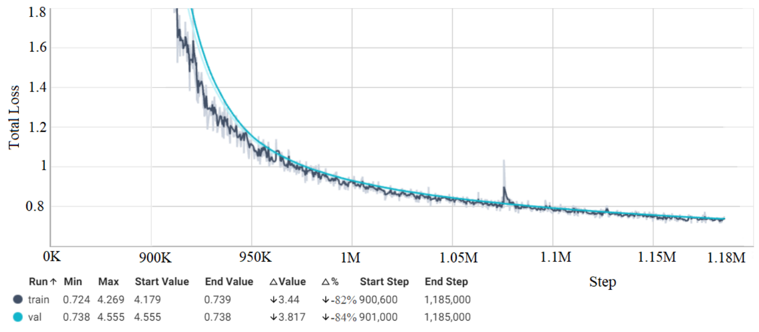
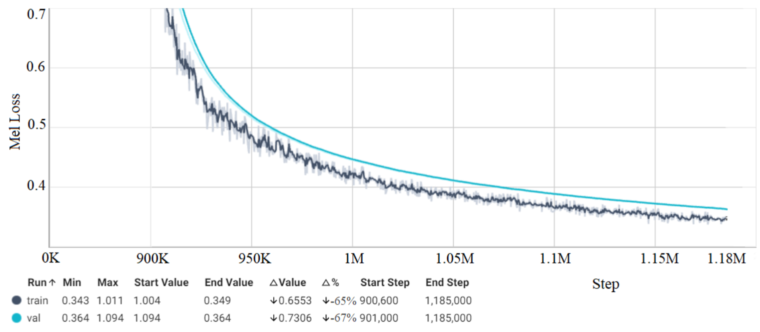


Figure 5. The memory experiment results using the TTS model and cache mechanism.



Run ↑	Min	Max	Start Value	End Value	△Value	△%	Start Step	End Step
● train	0.724	4.269	4.179	0.739	↓3.44	↓-82%	900,600	1,185,000
● val	0.738	4.555	4.555	0.738	↓3.817	↓-84%	901,000	1,185,000

Figure 6. The total loss of training and validation curves for FastSpeech 2 during the adaptation training.



Run ↑	Min	Max	Start Value	End Value	△Value	△%	Start Step	End Step
● train	0.343	1.011	1.004	0.349	↓0.6553	↓-65%	900,600	1,185,000
● val	0.364	1.094	1.094	0.364	↓0.7306	↓-67%	901,000	1,185,000

Figure 7. Mel Loss of the training and validation curves for FastSpeech 2 during the adaptation training.

4.6. Synthesized Speech Objective Evaluation

We utilized objective metrics evaluations to test the quality of the synthesized speech of the two target speakers. We compared all the results obtained by our modified TTS model to the baseline TTS model. We used Mel-Cepstral Distortion (MCD) and Frequency Weighted Segmental SNR (fwSNRseg).

1. MCD (dB):

The MCD indicates how closed the synthesized sentences are to the ground-truth sentences. It estimates the difference between the mel cepstra series (Equation (1)). As much as the MCD value is small, it means the synthesized sentences are close to the original sentences.

$$MCD = \frac{1}{N} \sum_{j=1}^{N} \sqrt{\sum_{i=1}^{K} (x_{i,j} - y_{i,j})^2} \tag{1}$$

x and y are the i th cepstral coefficients of the ground truth and synthesized speech signals. j represents the index of the frames.

2. FwSNRseg (dB):

It assesses the similarity between two characteristic sequences. Higher FWSEG implies a more satisfactory speech rate. It is an extension of the Segmental SNR (signal-to-noise ratio). It is computed using Equation (2).

$$\text{fwSNRseg} = \frac{10}{M}\sum_{m=0}^{M-1}\frac{\sum_{j=1}^{K} W(j,m)\log_{10}\frac{X(j,m)^2}{(X(j,m)-\hat{X}(j,m))^2}}{\sum_{j=1}^{K} W(j,m)} \tag{2}$$

M is the total number of segments; K is frequency bands; $X(j,m)$ is the Discrete Fourier Transform (DFT) coefficient magnitude; $\hat{X}(j,m)$ is the corresponding spectral magnitude of the synthesized speech signal in the same band; m is the original speech signal; m is the original speech signal; $W(j,m)$ is the frequency weighting factor.

Table 7 shows each target speaker's average MCD and FWSEG of ten synthesized sentences. The results show the superiority of the modified TTS model to the baseline TTS model in terms of synthesized speech quality.

Table 7. Objective metrics.

Speaker	Modified TTS Model		Baseline TTS Model	
	MCD	**FwSNRseg**	**MCD**	**FwSNRseg**
F	**5.17**	**1.0**	5.76	0.8
M	**6.0**	**0.67**	7.13	0.18

5. Discussion

Our target study is the possibility of creating IoT talk for industrial solutions by building an intelligent control system extension. The main challenge in creating such a system is the high computational requirements of the TTS models. Providing high computing resource devices is also an expensive solution for industries. To this end, we hypothesized using the cache mechanism to solve this issue because state-of-the-art TTS models currently do not offer appropriate real-time running times. Consequently, urgent requests or notifications are no longer valuable.

The results reveal that even when using FastSpeech 2 and HiFi-GAN V3, the output system sound alarm production is slow (RTF = 6.4, 20.3–24.9 s, as shown in Table 5). Therefore, this outcome fails to meet real-time needs. However, adding the cache mechanism to our system algorithms optimized the production of sound alarm speed to RTF = 0.29 as a maximum (Table 6). Moreover, studying the RAM efficiency in RPi has proven that using the cache mechanism enhances the performance of the embedding device (by reducing its usage by 61.35% when producing 10 s sentence). The computational complexity of the two algorithms exhibits a linear performance O(n) without using the cache mechanism, which increases with the input text length for each circumstance. On the other hand, the cache mechanism optimizes the complexity of our algorithm to O(1). Eventually, customizing the synthesized speech for the two target speakers successfully used small datasets.

To the best of our knowledge, this is the first study to consider employing a TTS model-based neural network with IoT for industrial purposes. Meanwhile, results may not be in real-life study settings because it was accomplished in a highly controlled environment. Therefore, this study will benefit from a case study in oil refineries and studying its impacts on real-life scenarios. Moreover, the data collected was obtained from only one sensor to demonstrate the effectiveness of the system. Accordingly, studying system behavior with different data types under various operational conditions should be considered. Regardless, the system method is the same for different sensed data types (by comparing the input data to the thresholds).

Recent advances in oil refineries have resulted in the complex monitoring of operational conditions and emergencies. For example, an abnormal condition causes a notification indicating a fire or operation problem (such as high temperature, level, and pressure in a drum). The priority response varies depending on these conditions. Furthermore, each alarm unit has a standardized set of simple tone alarms. On the other hand, modern control refinery panels are complicated because they are linked to numerous instruments, sensors, drums, and pumps. Typically, each device sounds an alarm without prioritization, coordination with others, or providing a guide. As a result, critical alarms may be lost in the noise and distraction of trivial ones, resulting in a cacophony. These levels of noise and distraction can have negative consequences. To solve this problem, our system can issue an emergency sound alarm (spoken). This sound alarm may also contain additional information regarding the situation, such as its location and possible solutions. For example, if the system detects a gas leak, it can generate a human-like voice alarm that warns operators about potential dangers and provides instructions for evacuating the area or shutting down specific equipment. Overall, the speech alarm messages generated by our proposed intelligent extension control system can play a critical role in ensuring the safety and efficiency of oil refineries. This system can prevent accidents and minimize downtime by providing operators with real-time alerts and instructions.

The development of new technologies raises ethical questions from several directions. On the one hand, from the point of view of research ethics—what kind of research can be allowed, and whether the freedom of research can be restricted on any basis. On the other hand, from the point of view of result implementation—the question is how we can "correctly" apply the new scientific results [75]. It is beyond the scope of this article to deal in more detail with the moral and ethical dilemmas surrounding technical development, robotics, and the rise of artificial intelligence. Researchers must know the limit regarding the research serving the community's interests. In our case, it is an ethical and social question of whether it poses a danger or what feelings it may evoke if an employee's voice is used as a model for the operation of the system. The uncanny valley phenomenon [76] can be a social dilemma when we send messages through our tested system as a real-time warning in direct response to a given situation.

It also has ethical implications, such as privacy concerns. Human-like speech-based alarm systems can potentially record and store audio data, therefore raising privacy concerns for workers. Appropriate safeguards should be implemented to protect the privacy of individuals and ensure that the collected data are used solely for safety purposes. The possibility of misuse must be ruled out, and internal regulation based on the national legal system must be created to determine how and in what way the voice samples used for training the system must be stored, who can access them, and how they can be used. It is essential to define the use framework and ensure appropriate legal protection for the person providing the sample. In this regard, European regulations are particularly strict. General Data Protection Regulation (GDPR), the data protection regulation of the European Union that entered into force in 2018 [77], specifically places great emphasis on protecting individuals against the state and companies. However, the law is simply unable to keep up with technical progress; therefore, in many cases, it is difficult to comply with inflexible rules.

6. Conclusions and Future Work

We built an intelligent extension IIoT design-based TTS for the control systems of oil refineries. We improved a TTS model (FastSpeech 2 and HiFi-GAN V3) to synthesize speech faster with better speech quality. We customized the TTS model using a small adaptation dataset to generate two target speakers (female and male). We compared our modified TTS model with a baseline model using RTF and objective speech quality metrics. We invested a cache mechanism method in our proposed design to maintain the generating uttered alarm inference suitable for real-time. We tested our system on a low resources-embedded system (Raspberry Pi 4) in a real-time environment using data from a remote gas sensor. The results illustrate that our system Real-Time Factor (RTF) is 6.4 (without the cache mechanism). Using the cache mechanism, our proposed model successfully solves the real-time TTS issue and runs on a low-resource computational device at real-time speed (RTF = 0.16, 0.19, and 0.29 when the memory contains 250, 500, and 1000 WAV files, respectively). Furthermore, the cache mechanism lowered memory usage in the embedded device from 16.3% (for producing a ten-second sentence) to 6.3%. Moreover, our TTS model outperformed the baseline TTS model in terms of objective speech quality evaluation. Our proposed system shows improved efficiency, real-time capabilities, and speech quality by utilizing advancements in IIoT and TTS technologies. The perspective of this study is to enhance operators' decision-making in emergencies with an effective and fast response in plant control rooms at industrial facilities (such as oil refineries).

A limitation of this study was that it was conducted in a controlled environment. Moreover, only one sensor (gas sensor) was used. Therefore, extending this work to real-life scenarios such as refinery oil should be considered with different input data types. Cybersecurity is a critical aspect to be addressed as an IIoT system. SSL/TLS protocol used in our approach should be reconsidered because of its drawbacks [64] for limited-resource devices such as IoT. Moreover, the performance depends on the accuracy of the abnormal condition detection (in our case, the threshold comparison). There remains the possibility of false alarms triggered by various factors, such as noisy data, unexpected environmental conditions, or minor fluctuations that may not pose significant risks. Therefore, other abnormal detection methods should be investigated.

In our future work, we will leverage the performance of FastSpeech 2 in hardware with low computational resources by reducing its parameters. Furthermore, loading the TTS model to the memory from the disk requires a long time. One solution is that the TTS model is continuously running and keeping it in the memory, and if there is a new request, the synthesis speed will be faster. We will change the TTS model architecture to run consistently in the memory. In addition, we will consider Autovocoder [78] in our TTS model. Autovocoder produces waveforms from a frame-based representation of equivalent size to the mel-spectrogram, and it is faster than other neural vocoders. Furthermore, we will utilize the zero-shot adaptation method to generalize our system to many languages (especially languages with insufficient resources). Subsequently, we extend it to other languages, such as Arabic and Hungarian. Finally, we will investigate our proposed system in a real-world oil refinery. This investigation includes economic analysis, evaluation of the system effect, and commercialization.

Author Contributions: A.R.M. designed the speech processing model and the computational experiment and analyzed the data. A.A.A. designed the IoT architecture and did the efficiency analysis. A.R.M. and A.A.A. wrote the manuscript. M.S.A.-R. reviewed the paper. T.G.C. was in charge of the study overall planning and direction. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The research was partly supported by the EU project RRF-2.3.1-21-2022-00004 within the AI National Laboratory. T. G.C.'s research was supported by the Bolyai Research Fellowship of the HAS and by the ÚNKP-22-5-BME-316 New National Excellence Program of the Ministry for Innovation and Technology (source of NRDIF). The Titan X GPU used was donated by NVIDIA.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

X	Data from the sensor node
S	Saved synthesized speech
S_i	The stored value element in the memory
T	Threshold value
TTS	Text-to-Speech model
U	HiveMQ user ID
P	HiveMQ password
Hs	HiveMQ hostname
R	connection port
H_i	HiveMQ cluster credentials
A	The Alarm
M_t	MQTT topic
M_p	MQTT protocol
D	Data received via MQTT

References

1. Sisinni, E.; Saifullah, A.; Han, S.; Jennehag, U.; Gidlund, M. Industrial Internet of Things: Challenges, Opportunities, and Directions. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4724–4734. [CrossRef]
2. Gosine, R.; Warrian, P. Digitalizing extractive industries: The state-of-the-art to the art-of-the possible. In *Munk School of Global Affairs Innovation Policy Lab White Paper Series 2017–004*; University of Toronto: Toronto, ON, Canada, 2017.
3. Hazra, A.; Adhikari, M.; Amgoth, T.; Srirama, S.N. A Comprehensive Survey on Interoperability for IIoT: Taxonomy, Standards, and Future Directions. *ACM Comput. Surv.* **2021**, *55*, 9. [CrossRef]
4. Ramos, M.A.; Droguett, E.L.; Mosleh, A.; Moura, M.D.C. A Human Reliability Analysis Methodology for Oil Refineries and Petrochemical Plants Operation: Phoenix-PRO Qualitative Framework. *Reliab. Eng. Syst. Saf.* **2020**, *193*, 106672. [CrossRef]
5. Yoo, B.-T.; Shim, W.S. Evaluating the Efficiency of the Process Safety Management System through Analysis of Major Industrial Accidents in South Korea. *Processes* **2023**, *11*, 2022. [CrossRef]
6. Bloch, K.P.; Wurst, D.M. Process Safety Management Lessons Learned from a Petroleum Refinery Spent Caustic Tank Explosion. *Process Saf. Prog.* **2010**, *29*, 332–339. [CrossRef]
7. U.S. Chemical Safety and Hazard Investigation Board. *Investigation Report Catastrophic Rupture of Heat Exchanger (Seven Fatalities)—Report 2010–08-I-WA2014*; U.S. Chemical Safety and Hazard Investigation Board: Washington, DC, USA, 2014.
8. Nwankwo, C.D.; Arewa, A.O.; Theophilus, S.C.; Esenowo, V.N. Analysis of accidents caused by human factors in the oil and gas industry using the HFACS-OGI framework. *Int. J. Occup. Saf. Ergon.* **2021**, *28*, 1642–1654. [CrossRef] [PubMed]
9. U.K. Health and Safety Executive (HSE). *Core Topic 3: Identifying Human Failures*; U.K. Health and Safety Executive (HSE): London, UK, 2005.
10. Mandapaka, P.V.; Lo, E.Y. Assessing Shock Propagation and Cascading Uncertainties Using the Input–Output Framework: Analysis of an Oil Refinery Accident in Singapore. *Sustainability* **2023**, *15*, 1739. [CrossRef]
11. Foley, L.; Anderson, C.J.; Schutz, M. Re-Sounding Alarms: Designing Ergonomic Auditory Interfaces by Embracing Musical Insights. *Healthcare* **2020**, *8*, 389. [CrossRef]
12. Chikara, R.K.; Ko, L.W. Modulation of the Visual to Auditory Human Inhibitory Brain Network: An EEG Dipole Source Localization Study. *Brain Sci.* **2019**, *9*, 216. [CrossRef]
13. Webster, C.S.; Sanderson, P. Need for a New Paradigm in the Design of Alarms for Patient Monitors and Medical Devices. *Br. J. Anaesth.* **2021**, *127*, 677–680. [CrossRef]
14. Haslwanter, J.D.H.; Heiml, M.; Wolfartsberger, J. Lost in translation: Machine translation and text-to-speech in industry 4.0. In Proceedings of the 12th ACM International Conference on PErvasive Technologies Related to Assistive Environments, Rhode Island, Greece, 5–7 June 2019; pp. 333–342.
15. Ning, Y.; He, S.; Wu, Z.; Xing, C.; Zhang, L.-J. A Review of Deep Learning Based Speech Synthesis. *Appl. Sci.* **2019**, *9*, 4050. [CrossRef]
16. Luo, R.; Tan, X.; Wang, R.; Qin, T.; Li, J.; Zhao, S.; Chen, E.; Liu, T.-Y. Lightspeech: Lightweight and Fast Text to Speech with Neural Architecture Search. In Proceedings of the ICASSP 2021—2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 6–11 June 2021. [CrossRef]
17. Yang, G.; Yang, S.; Liu, K.; Fang, P.; Chen, W.; Xie, L. Multi-Band Melgan: Faster Waveform Generation for High-Quality Text-To-Speech. In Proceedings of the 2021 IEEE Spoken Language Technology Workshop (SLT), Shenzhen, China, 19–22 January 2021. [CrossRef]
18. Kawamura, M.; Shirahata, Y.; Yamamoto, R.; Tachibana, K. Lightweight and High-Fidelity End-to-End Text-to-Speech with Multi-Band Generation and Inverse Short-Time Fourier Transform. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Rhodes Island, Greece, 4–10 June 2023. [CrossRef]

19. Achanta, S.; Antony, A.; Golipour, L.; Li, J.; Raitio, T.; Rasipuram, R.; Rossi, F.; Shi, J.; Upadhyay, J.; Winarsky, D.; et al. On-Device Neural Speech Synthesis. In Proceedings of the 2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), Cartagena, Colombia, 13–17 December 2021. [CrossRef]
20. Zainkó, C.; Bartalis, M.; Németh, G.; Olaszy, G. A Polyglot Domain Optimised Text-to-Speech System for Railway Station Announcements. In Proceedings of the Interspeech 2015, Dresden, Germany, 6–10 September 2015. [CrossRef]
21. Mandeel, A.R.; Aggar, A.A.; Al-Radhi, M.S.; Csapó, T.G. Implementing a Text-to-Speech Synthesis Model on a Raspberry Pi for Industrial Applications. In Proceedings of the 1st Workshop on Intelligent Infocommunication Networks, Systems and Services, Budapest, Hungary, 7 February 2023. [CrossRef]
22. Ramzey, H.; Badawy, M.; Elhosseini, M.; Elbaset, A.A. I2OT-EC: A Framework for Smart Real-Time Monitoring and Controlling Crude Oil Production Exploiting IIOT and Edge Computing. *Energies* **2023**, *16*, 2023. [CrossRef]
23. Priyanka, E.; Maheswari, C.; Ponnibala, M.; Thangavel, S. SCADA Based Remote Monitoring and Control of Pressure & Flow in Fluid Transport System Using IMC-PID Controller. *Adv. Syst. Sci. Appl.* **2019**, *19*, 140–162. [CrossRef]
24. Priyanka, E.B.; Maheswari, C.; Thangavel, S. A Smart-integrated IoT Module for Intelligent Transportation in Oil Industry. *Int. J. Numer. Model. Electron. Netw. Devices Fields* **2020**, *34*, e2731. [CrossRef]
25. Henry, N.F.; Henry, O.N. Wireless Sensor Networks Based Pipeline Vandalisation and Oil Spillage Monitoring and Detection: Main Benefits for Nigeria Oil and Gas Sectors. *SIJ Trans. Comput. Sci. Eng. Its Appl. (CSEA)* **2019**, *7*, 1–7. [CrossRef]
26. Wanasinghe, T.R.; Gosine, R.G.; James, L.A.; Mann, G.K.; De Silva, O.; Warrian, P.J. The Internet of Things in the Oil and Gas Industry: A Systematic Review. *IEEE Internet Things J.* **2020**, *7*, 8654–8673. [CrossRef]
27. Carroll, K.; Chandramouli, M. *Scaling IoT to Meet Enterprise Needs—Balancing Edge and Cloud Computing*; Deloitte: London, UK, 2019.
28. Hossain, M.S.; Rahman, M.; Sarker, M.T.; Haque, M.E.; Jahid, A. A Smart IoT Based System for Monitoring and Controlling the Sub-Station Equipment. *Internet Things* **2019**, *7*, 100085. [CrossRef]
29. Parjane, V.A.; Gangwar, M. *Corrosion Detection and Prediction Approach Using IoT and Machine Learning Techniques*; Lecture Notes in Networks and Systems; Springer: Berlin/Heidelberg, Germany, 2022; pp. 205–215. [CrossRef]
30. Singh, R.; Baz, M.; Narayana, C.L.; Rashid, M.; Gehlot, A.; Akram, S.V.; Alshamrani, S.S.; Prashar, D.; AlGhamdi, A.S. Zigbee and Long-Range Architecture Based Monitoring System for Oil Pipeline Monitoring with the Internet of Things. *Sustainability* **2021**, *13*, 10226. [CrossRef]
31. Spandonidis, C.; Theodoropoulos, P.; Giannopoulos, F. A Combined Semi-Supervised Deep Learning Method for Oil Leak Detection in Pipelines Using IIoT at the Edge. *Sensors* **2022**, *22*, 4105. [CrossRef]
32. Lade, P.; Ghosh, R.; Srinivasan, S. Manufacturing Analytics and Industrial Internet of Things. *IEEE Intell. Syst.* **2017**, *32*, 74–79. [CrossRef]
33. Ijiga, O.E.; Malekian, R.; Chude-Okonkwo, U.A. Enabling Emergent Configurations in the Industrial Internet of Things for Oil and Gas Explorations: A Survey. *Electronics* **2020**, *9*, 1306. [CrossRef]
34. Javadi, S.H.; Mohammadi, A. Fire Detection by Fusing Correlated Measurements. *J. Ambient. Intell. Humaniz. Comput.* **2017**, *10*, 1443–1451. [CrossRef]
35. AlSuwaidan, L. The Role of Data Management in the Industrial Internet of Things. *Concurr. Comput. Pract. Exp.* **2020**, *33*, e6031. [CrossRef]
36. Ahmed, S.; Le Mouël, F.; Stouls, N.; Lipeme Kouyi, G. Development and Analysis of a Distributed Leak Detection and Localisation System for Crude Oil Pipelines. *Sensors* **2023**, *23*, 4298. [CrossRef] [PubMed]
37. Zhang, P.; Chen, X.; Fan, C. Research on a Safety Assessment Method for Leakage in a Heavy Oil Gathering Pipeline. *Energies* **2020**, *13*, 1340. [CrossRef]
38. Liu, R.; Ding, S.; Ju, G. Numerical Study of Leakage and Diffusion of Underwater Oil Spill by Using Volume-of-Fluid (VOF) Technique and Remediation Strategies for Clean-Up. *Processes* **2022**, *10*, 2338. [CrossRef]
39. Varga, P.; Bácsi, S.; Sharma, R.; Fayad, A.; Mandeel, A.R.; Soos, G.; Franko, A.; Fegyo, T.; Ficzere, D. Converging Telco-Grade Solutions 5G and Beyond to Support Production in Industry 4.0. *Appl. Sci.* **2022**, *12*, 7600. [CrossRef]
40. Zhao, W.; Yang, Z. An Emotion Speech Synthesis Method Based on VITS. *Appl. Sci.* **2023**, *13*, 2225. [CrossRef]
41. Kiangala, K.S.; Wang, Z. An Experimental Safety Response Mechanism for an Autonomous Moving Robot in a Smart Manufacturing Environment Using Q-Learning Algorithm and Speech Recognition. *Sensors* **2022**, *22*, 941. [CrossRef]
42. Du, G.; Chen, M.; Liu, C.; Zhang, B.; Zhang, P. Online Robot Teaching with Natural Human–Robot Interaction. *IEEE Trans. Ind. Electron.* **2018**, *65*, 9571–9781. [CrossRef]
43. Stefaniak, P.; Stachowiak, M.; Koperska, W.; Skoczylas, A.; Śliwiński, P. Application of Wearable Computer and ASR Technology in an Underground Mine to Support Mine Supervision of the Heavy Machinery Chamber. *Sensors* **2022**, *22*, 7628. [CrossRef]
44. Chen, H.; Leu, M.C.; Yin, Z. Real-Time Multi-Modal Human–Robot Collaboration Using Gestures and Speech. *J. Manuf. Sci. Eng.* **2022**, *144*, 101007. [CrossRef]
45. Mo, D.-H.; Tien, C.-L.; Yeh, Y.-L.; Guo, Y.-R.; Lin, C.-S.; Chen, C.-C.; Chang, C.-M. Design of Digital-Twin Human-Machine Interface Sensor with Intelligent Finger Gesture Recognition. *Sensors* **2023**, *23*, 3509. [CrossRef]
46. Siyaev, A.; Jo, G.-S. Towards Aircraft Maintenance Metaverse Using Speech Interactions with Virtual Objects in Mixed Reality. *Sensors* **2021**, *21*, 2066. [CrossRef]
47. Latif, S.; Qadir, J.; Qayyum, A.; Usama, M.; Younis, S. Speech Technology for Healthcare: Opportunities, Challenges, and State of the Art. *IEEE Rev. Biomed. Eng.* **2021**, *14*, 342–356. [CrossRef]

48. Silvestri, R.; Holmes, A.; Rahemtulla, R. The Interaction of Cognitive Profiles and Text-to-Speech Software on Reading Comprehension of Adolescents with Reading Challenges. *J. Spec. Educ. Technol.* **2021**, *37*, 498–509. [CrossRef]
49. Kato, S.; Yasuda, Y.; Wang, X.; Cooper, E.; Takaki, S.; Yamagishi, J. Modeling of Rakugo Speech and Its Limitations: Toward Speech Synthesis That Entertains Audiences. *IEEE Access* **2020**, *8*, 138149–138161. [CrossRef]
50. Chung, Y.-A.; Wang, Y.; Hsu, W.-N.; Zhang, Y.; Skerry-Ryan, R.J. Semi-Supervised Training for Improving Data Efficiency in End-to-End Speech Synthesis. In Proceedings of the ICASSP 2019—2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019. [CrossRef]
51. Mandeel, A.R.; Al-Radhi, M.S.; Csapó, T.G. *Speaker Adaptation with Continuous Vocoder-Based DNN-TTS*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2021; pp. 407–416. [CrossRef]
52. Mandeel, A.R.; Al-Radhi, M.S.; Csapó, T.G. Investigations on Speaker Adaptation Using a Continuous Vocoder within Recurrent Neural Network Based Text-to-Speech Synthesis. *Multimed. Tools Appl.* **2023**, *82*, 15635–15649. [CrossRef]
53. Schnell, B.; Garner, P.N. Investigating a Neural All Pass Warp in Modern TTS Applications. *Speech Commun.* **2022**, *138*, 26–37. [CrossRef]
54. Eren, E.; Demiroglu, C. Deep Learning-Based Speaker-Adaptive Postfiltering with Limited Adaptation Data for Embedded Text-to-Speech Synthesis Systems. *Comput. Speech Lang.* **2023**, *81*, 101520. [CrossRef]
55. Mandeel, A.R.; Al-Radhi, M.S.; Csapó, T.G. Speaker Adaptation Experiments with Limited Data for End-to-End Text-To-Speech Synthesis Using Tacotron2. *Infocommun. J.* **2022**, *14*, 55–62. [CrossRef]
56. Huang, S.-F.; Lin, C.-J.; Liu, D.-R.; Chen, Y.-C.; Lee, H.-Y. Meta-TTS: Meta-Learning for Few-Shot Speaker Adaptive Text-to-Speech. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2022**, *30*, 1558–1571. [CrossRef]
57. Wang, X. Embedded Task System and Gaussian Mixture Model in the Analysis and Application of User Behavior in Marketing Management. *Wirel. Netw.* **2021**, 1–13. [CrossRef]
58. Karami, M.; McMorrow, G.V.; Wang, L. Continuous Monitoring of Indoor Environmental Quality Using an Arduino-Based Data Acquisition System. *J. Build. Eng.* **2018**, *19*, 412–419. [CrossRef]
59. Leonard, B. (Ed.) *Advances on P2P, Parallel, Grid, Cloud and Internet Computing*; Lecture Notes in Networks and Systems; Springer: Berlin/Heidelberg, Germany, 2023. [CrossRef]
60. Champaty, B.; Nayak, S.K.; Thakur, G.; Mohapatra, B.; Tibarewala, D.N.; Pal, K. Development of Bluetooth, Xbee, and Wi-Fi-Based Wireless Control Systems for Controlling Electric-Powered Robotic Vehicle Wheelchair Prototype. In *Robotic Systems: Concepts, Methodologies, Tools, and Applications;* IGI Global: Hershey, PA, USA, 2020; pp. 1048–1079. [CrossRef]
61. Dewanta, F. A Study of Secure Communication Scheme in MQTT: TLS vs. AES Cryptography. *J. Infotel* **2022**, *14*, 269–276. [CrossRef]
62. Fadhil, T.Z.; Mandeel, A.R. Live Monitoring System for Recognizing Varied Emotions of Autistic Children. In Proceedings of the International Conference on Advanced Science and Engineering (ICOASE), Duhok, Iraq, 9–11 October 2018. [CrossRef]
63. Di Paolo, E.; Bassetti, E.; Spognardi, A. Security Assessment of Common Open Source MQTT Brokers and Clients. In *ITASEC*; ITASEC: São Paulo, Brazil, 2021; pp. 475–487.
64. Paris IL, B.M.; Habaebi, M.H.; Zyoud, A.M. Implementation of SSL/TLS Security with MQTT Protocol in IoT Environment. *Wirel. Pers. Commun.* **2023**, 1–20. [CrossRef]
65. Hyperscale with HiveMQ: Learn about Scale from Our 200 Million Benchmark. Available online: https://www.hivemq.com/blog/hyperscale-iot-iiot-applications-up-to-200-mil-connections-with-hivemq/ (accessed on 21 May 2023).
66. Koziolek, H.; Grüner, S.; Rückert, J. *A Comparison of MQTT Brokers for Distributed IoT Edge Computing*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2020; pp. 352–368. [CrossRef]
67. Ren, Y.; Hu, C.; Tan, X.; Qin, T.; Zhao, S.; Zhao, Z.; Liu, T.-Y. Fastspeech 2: Fast and High-Quality End-to-End Text to Speech. In ICLR. OpenReview.net, 2021. Available online: https://openreview.net/forum?id=piLPYqxtWuA (accessed on 8 July 2021).
68. Kong, J.; Kim, J.; Bae, J. HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 17022–17033.
69. Kumar, K.; Kumar, R.; De Boissiere, T.; Gestin, L.; Teoh, W.Z.; Sotelo, J.; De Brebisson, A.; Bengio, Y.; Courville, A.C. Melgan: Generative adversarial networks for conditional waveform synthesis. In Proceedings of the Advances in Neural Information Processing Systems 32 (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019.
70. Ito, K.; Johnson, L. The LJ Speech Dataset. 2017. Available online: https://keithito.com/LJ-Speech-Dataset/ (accessed on 22 December 2020).
71. Bakhturina, E.; Lavrukhin, V.; Ginsburg, B.; Zhang, Y. Hi-Fi Multi-Speaker English TTS Dataset. In Proceedings of the Interspeech, Brno, Czech Republic, 30 August–3 September 2021. [CrossRef]
72. McAuliffe, M.; Socolof, M.; Mihuc, S.; Wagner, M.; Sonderegger, M. Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi. In Proceedings of the Interspeech 2017, Stockholm, Sweden, 20–24 August 2017. [CrossRef]
73. Takieldeen, A.E.; El-kenawy, E.S.M.; Hadwan, M.; Zaki, R.M. Dipper Throated Optimization Algorithm for Unconstrained Function and Feature Selection. *Comput. Mater. Contin.* **2022**, *72*, 1465–1481. [CrossRef]
74. Philip, M.A.; Vaithiyanathan. A Survey on Lightweight Ciphers for IoT Devices. In Proceedings of the International Conference on Technological Advancements in Power and Energy (TAP Energy), Kollam, India, 21–23 December 2017. [CrossRef]
75. Homicskó, Á. (Ed.) *Egyes Modern Technológiák Etikai, Jogi és Szabályozási Kihívásai. Acta Caroliensia Conventorum Scientiarum Iuridico-Politicarum XXII*; KRE: Budapest, Hungary, 2018; p. 223.

76. Rothstein, N.; Kounios, J.; Ayaz, H.; de Visser, E.J. Assessment of Human-Likeness and Anthropomorphism of Robots: A Literature Review. *Adv. Intell. Syst. Comput.* **2020**, *28*, 190–196. [CrossRef]
77. Otto, M. Regulation (EU) 2016/679 on the Protection of Natural Persons with Regard to the Processing of Personal Data and on the Free Movement of Such Data (General Data Protection Regulation—GDPR). In *International and European Labour Law*; Nomos Verlagsgesellschaft mbH & Co. KG: Baden-Baden, Germany, 2018; pp. 958–981. [CrossRef]
78. Webber, J.J.; Valentini-Botinhao, C.; Williams, E.; Henter, G.E.; King, S. Autovocoder: Fast Waveform Generation from a Learned Speech Representation Using Differentiable Digital Signal Processing. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Rhodes Island, Greece, 4–10 June 2023. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.