SPECIAL FEATURE

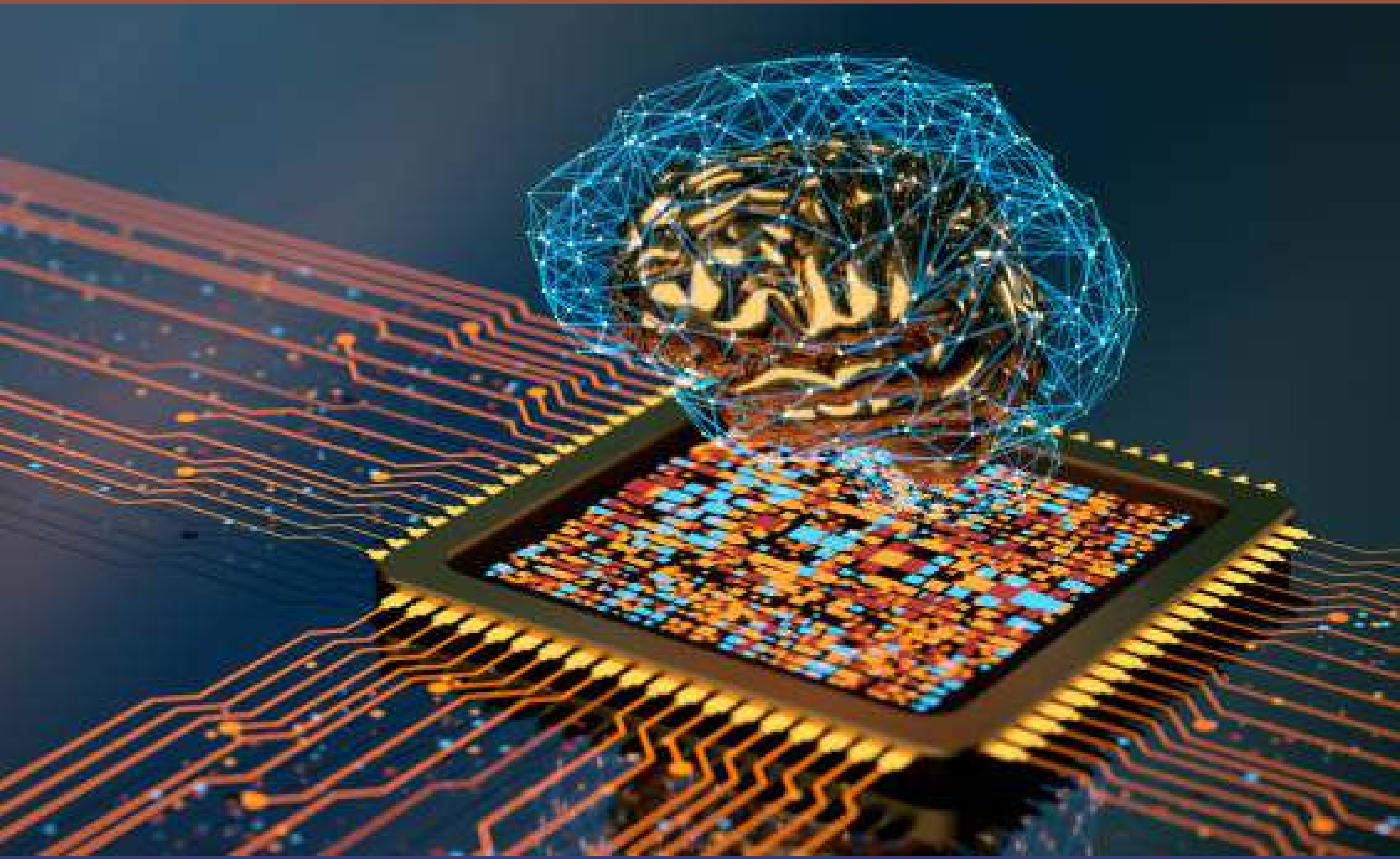
Stable Image Diffusion for Enhanced Visual Synthesis

Silicon University

The Science & Technology Magazine

Digital Digest

Vol. 32 • October - December 2024



Our Vision: "To become a center of excellence in the fields of technical education & research and create responsible citizens"

Illuminated Machine Learning

The fast development of artificial intelligence has led to increasingly complicated deep neural network models that defy traditional electrical computing systems. These networks that power today's most demanding machine-learning applications have grown so large and complex that they are pushing the limits of traditional electronic computing hardware. To meet this demand, a photonic processor has been built that uses light to do machine-learning computations which vastly improves speed and energy efficiency. Photonic systems thrive at matrix multiplication for linear neural network transformations and are thus miles above traditional electronic hardware. However, the photonic systems struggle with nonlinear operations for pattern recognition and complicated problem-solving. Furthermore, there are some types of neural network computations that a photonic device can't perform which requires the use of off-chip electronics or other. To overcome the limitations of the photonic device, an optical device has been developed to do all deep learning computations on the chip by integrating nonlinear optical function units (NOFUs). The initial model was first developed by a group of researchers in 2017. This “photonic chip” represents a quantum leap forward in machine learning computing power and efficiency thanks to its optical neural network architecture and its network of interconnected modules. It encodes the parameters of a deep neural network into light. Then, an array of programmable beam splitters performs matrix multiplication on those inputs. The data is then passed to programmable NOFUs, which implement nonlinear functions by siphoning off a small amount of light to photodiodes that convert optical signals to electric current. This process, which eliminates the need for an external amplifier, consumes very little energy. This device outperforms conventional hardware by a wide margin, finishing critical calculations for machine learning tasks in less than half a millisecond with an accuracy of more than 92%. The optical device has wide use in popular industries like LIDAR (Light Detection and Ranging) technology which is used in many industries like mining, space, and meteorology. The optical device can also be widely used in telecommunications as well as for research endeavors in domains like particle physics and astronomy. The possibilities for real-time learning applications are further expanded when ultra-low latency and energy economy are achieved simultaneously. The U.S. National Science Foundation is funding large-scale research in this area for scaling up the photonic processor and incorporating it with useful electrical devices like cameras and telecommunications networks. Thus, the “photonic chip” has the potential to revolutionize the fields of artificial intelligence and computing hardware.

Dr. Pragyan Paramita Das
Dept. of CSE

Stable Image Diffusion for Enhanced Visual Synthesis

***Abstract** – Recent advancements in Artificial Intelligence (AI), particularly in stable diffusion models, have significantly impacted image synthesis, but their role in digital rock analysis remains underexplored. This article investigates the application of stable diffusion to enhance digital rock studies, aiming to boost image clarity, noise reduction, segmentation, and the reconstruction of 3D formations from 2D data. The model effectively bridges gaps in rock imagery, distinguishes between rock types, and extracts network properties, offering deeper insights into rock formations. Nonetheless, current models encounter limitations such as the lack of authentic digital rock datasets and an incomplete understanding of the physical phenomena involved. This work aims to refine current image synthesis techniques by addressing resolution, detail accuracy, and realism. By implementing unique modifications using deep learning platforms like TensorFlow and PyTorch, as well as leveraging hardware accelerators such as Graphics Processing Units (GPUs) and Tensor Processing Units (TPUs), the study aims to advance fields like virtual reality, healthcare, and computer vision. Integrating stable diffusion into digital core analysis holds great potential to revolutionize this area and foster breakthroughs in diverse industries.*

***Keywords** – Stable Diffusion, Image Processing Libraries, Quality Enhancement, Realistic image synthesis, Hardware accelerators.*

I. Introduction

In the dynamic realm of artificial intelligence and machine learning, the exploration of stable diffusion, and generative modeling stands as a cornerstone in understanding and replicating complex real-world phenomena [1]. At its essence, stable diffusion delves into the behavior of random variables over time, particularly focusing on processes exhibiting long-range dependence and heavy-tailed distributions. Meanwhile, generative modeling seeks to create models capable of generating synthetic data samples mirroring real-world observations. Both fields heavily rely on modeling data as distributions, recognizing the inherent stochastic nature of real-world phenomena and leveraging probability theory to capture underlying patterns and uncertainties.

II. Literature Survey

Stable diffusion, rooted in statistical physics and stochastic processes, deals with understanding how random variables evolve while preserving specific statistical properties. This concept finds relevance in a multitude of domains, from financial markets to fluid dynamics, where traditional Gaussian models often fall short of capturing the inherent complexity of observed phenomena [2]. By studying stable diffusion, researchers aim to unravel the intricate dynamics of systems characterized by long-range dependencies and heavy-tailed distributions, offering insights into phenomena exhibiting diverse behaviors.

On the other hand, generative modeling focuses on creating models capable of generating synthetic data samples resembling real-world observations. Through techniques like variational autoencoders (VAEs), generative adversarial networks (GANs), and autoregressive models, generative modeling has seen significant advancements, enabling the creation of synthetic data with remarkable fidelity[3]. This capability finds applications across various domains, including image generation, text synthesis, and anomaly detection, where generating realistic data samples is crucial for training and evaluation purposes.

The paradigm of modeling data as distributions is central to both stable diffusion and generative modeling [4]. Rather than viewing data as discrete observations, treating data as probability distributions provides a powerful framework for understanding underlying variability and uncertainty. This approach acknowledges the stochastic nature of real-world phenomena, allowing practitioners to capture complex dependencies and model uncertainties effectively.

The decision to model data as distributions is grounded in the recognition that many real-world phenomena exhibit inherent stochasticity and variability [5]. From stock market fluctuations to weather patterns, the dynamics of such systems cannot be fully captured by deterministic models alone. By embracing the probabilistic nature of data, practitioners can develop more robust and flexible modeling approaches, capable of capturing the inherent variability and uncertainties present in the data. Moreover, modeling data as distributions allows for the incorporation of domain knowledge and prior beliefs into the modeling process. By leveraging probabilistic models, practitioners can encode prior information about the underlying structure of the data, enhancing the interpretability of models and enabling more principled decision-making in real-world applications.

Treating data as distributions also facilitates the development of generative models capable of capturing underlying variability and structure. By explicitly modeling the probability distributions underlying the observed data, generative models can learn to generate synthetic samples closely resembling real-world observations [2]. This enables practitioners to explore the latent space of the data [6], uncover hidden patterns, and generate novel insights, further enriching our understanding of complex phenomena.

The project addresses limitations present in current image synthesis methods. Challenges such as image resolution, detail capture, and overall realism are identified and targeted for improvement.

III. Methodology

Stable diffusion is a text-to-image deep learning model, based on diffusion model [7]. Diffusion models are advanced machine learning algorithms [8] that uniquely generate high-quality data by progressively adding noise to a dataset and then learning to reverse this process. The forward process is to destroy all the information in the image progressively in a sequence of time steps t, whereas, in each step, we add a little bit of Gaussian noise and by the end of many steps what we have is a completely random noise mimicking a sample from a normal distribution. The Eq. 1 [9] below is the transition function used to move from x_{t-1} to x_t based on conditional probability. Here b_t is the variance of the diffusion process, N is the normal distribution and I is the identity matrix.

Then a reverse process is learned for the same via a neural network. But how does modeling a forward process benefit us if ultimately an image is converted to noise? Well, it turns out that under certain conditions that are met here the reverse process with transition has the same functional form which means that our reverse process is generating data from random noise will also be a Markov chain with Gaussian transition probability. However, this would require computations involving the entire data distribution. However, we can approximate it by a distribution r_q which is formulated as a Gaussian in Eq. 2 [9]. Here m_q is the mean.

$$q(x_t \mid x_{t-1}) := N(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t I) \qquad (1)$$

$$p_\theta(x_t - 1 \mid x_t) \sim N(x_{t-1};\ \mu_\theta, \Sigma_\theta(x_t, t)) \qquad (2)$$

Reverse distribution can't be computed, but then approximated using p. Training is performed by optimizing the usual variational bound on negative log-likelihood [3]:

$$\log p(x) = \log \int p(x,z)\ dz \qquad (3)$$

$$\log p(x) >= E_{q(z|x)}[\log(p(x|z)/q(z|x)] \qquad (4)$$

Using Eq. 3 [9] to reach Eq. 4 [9] which is the Evidence Lower Bound (ELBO). We minimize the loss which in turn maximizes the ELBO and this will maximize our likelihood. The training and sampling processes are mentioned in algorithms 1 and 2 respectively [9].

```
Algorithm 1 Training
1: repeat
2:   $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3:   $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4:   $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5:   Take gradient descent step on
         $\nabla_\theta \left\| \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t) \right\|^2$
6: until converged
```

During training, select an image from our dataset, a time step T evenly, and random noise from the normal distribution. After getting the noisy version of the image. Then the noisy image is fed to the neural network, and we train the network using the loss to ensure that the predicted noise ($\in_0$) is as close to the actual noise ($\in$) as possible.

For image generation, we need to go through a reverse process, and that can be done by iterative sampling through the approximation of the denoising step distribution p that the neural network has learned. A random sample from a normal distribution (x_t) is fed to the trained model to get predicted noise.

Algorithm 2 Sampling

```
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: for $t = T, \ldots, 1$ do
3:     $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4:     $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
5: end for
6: return $\mathbf{x}_0$
```

Text-to-Image Architecture: Text-to-image generation involves the process of generating realistic images from textual descriptions as shown in Fig. 1 [10]. The workflow for text-to-image generation typically follows these steps:

a. Prompt Generation: The process begins with the formulation of a textual prompt that describes the desired image. For example, if the goal is to generate an image of a dog wearing glasses, the prompt would be "a dog with glasses."

b. Noise Sampling: Random noise is sampled from a predefined distribution, such as a Gaussian distribution. This noise serves as the initial input for the generation processes [6].

c. Encoding: The sampled noise is then encoded using a variational autoencoder (VAE) [6]. The VAE compresses the noise into a lower-dimensional latent space representation, often denoted as Z. This latent representation captures the essential characteristics of the noise.

d. Denoising: The latent representation Z is passed through a denoising unit along with a conditioning signal derived from the textual prompt [11]. The denoising unit aims to remove noise from the latent representation while incorporating the conditioning signal to guide the generation process toward fulfilling the prompt. At each denoising step, the unit estimates the amount of noise present in the latent representation.

e. Iterative Denoising: The denoising process is performed iteratively for multiple steps. At each step, the amount of noise in the latent representation is progressively reduced, bringing the generated image closer to the desired output described by the prompt [2]. The number of denoising steps may vary depending on the complexity of the generation task.

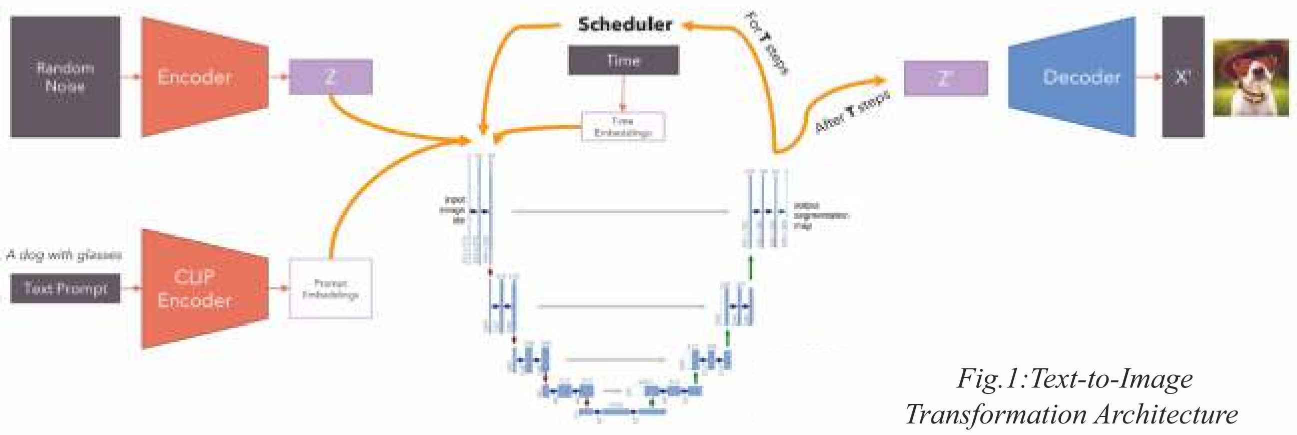


Fig.1:Text-to-Image Transformation Architecture

f. Decoding and Image Generation: Once the denoising process is complete, the resulting latent representation Z is passed through a decoder network. The decoder reconstructs the latent representation into an image that corresponds to the textual prompt. This generated image is the final output of the text-to-image generation process.

Image-to-Image Architecture.

Image-to-image generation involves transforming one image into another while preserving its essential characteristics. The process is shown in Fig. 2 [10] and can be described as follows:

a. Input Image Encoding: The image to be transformed (referred to as the input image) is encoded using the encoder of a variational autoencoder (VAE). This encoding process produces a latent representation of the input image, capturing its key features and characteristics.

b. Noise Injection: Noise is added to the latent representation of the input image. The amount of noise added determines the degree of freedom the model has in altering the input image during the

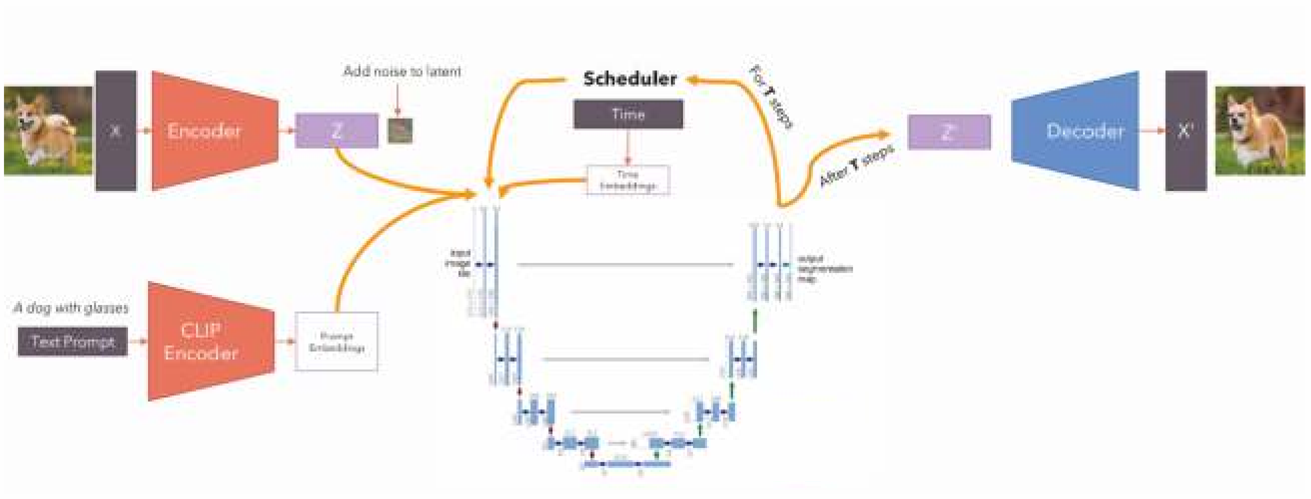


Fig.2:Image to Image Transformation Architecture

transformation process. More noise allows for greater modifications, while less noise restricts changes to minor alterations.

c. Denoising: The latent representation of the noisy input image is passed through a denoising unit, along with a conditioning signal derived from a prompt or textual description of the desired transformation [11]. The denoising unit iteratively removes noise from the latent representation while incorporating the conditioning signal to guide the transformation process.

d. Iterative Denoising: The denoising process is repeated for multiple steps, gradually reducing the amount of noise in the latent representation. Each denoising step brings the transformed image closer to the desired output specified by the prompt.

e. Decoding and Image Generation: Once the denoising process is complete and there is minimal noise remaining in the latent representation, the processed latent representation is passed through a decoder network. The decoder reconstructs the latent representation into an image that represents the transformed version of the input image. This generated image is the final output of the image-to-image generation process.

Image-to-image generation allows for a wide range of transformations, from simple modifications such as adding objects or altering attributes to more complex changes like style transfer or image in painting. By conditioning the transformation process on a prompt or textual description, the model learns to generate images that match the specified criteria, enabling versatile image editing and manipulation capabilities.

IV. Experimental Studies

The required environment setup and libraries were Python Version: 3.11.3, torch: 2.0.1, NumPy: 1.25.0, tqdm: 4.65.0, transformers: 4.33.2, lightning: 2.0.9, pillow: 9.5.0. The above-mentioned libraries were used for implementing the solution and are essential for the functionality and performance of the code. The input to are the Tokenizer Files in the form of vocab.json and merges.txt. Tokenizers are one of the core components of the pipeline. They translate text into data that can be processed by the model. Models can only process numbers, so tokenizers need to convert our text inputs to numerical data.

The result of the experiments with input and output are mentioned as follows:

Input for Text to Image Transformation:

prompt="A dog with sunglasses, looking at camera, highly detailed, ultra sharp, cinematic, 100mm lens, 4k resolution."

#prompt="A man with an umbrella, highly detailed, ultra sharp, cinematic, 100mm lens, 8k resolution." #"A dog stretching on the floor"

#prompt="A kangaroo holding a beer, wearing ski goggles and passionately singing silly songs"

uncond_prompt = "" #Also known as negative prompt

do_cfg = True

cfg_scale = 8 #min: 1, max: 14

Output Image:

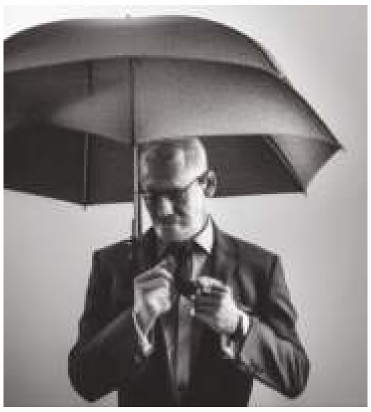


Fig.3:Image Generated In Text-To-Image Transformation

Quality and Effectiveness of Output Image:

The generated images in Fig 3, exhibit varying degrees of visual quality, depending on factors such as the complexity of the prompt, the quality of the trained models, and the presence of an input image for conditioning. Images may range from photorealistic renderings to abstract interpretations of the prompt. The effectiveness of the text-to-image generation process is evaluated based on how well the generated images align with the intended interpretation of the prompt. High-quality images that closely resemble the described scene indicate a successful generation.

Image To Image Transformation:

When an input image is provided for image-to-image transformation, the generated output image is expected to exhibit characteristics of both the input image and the specified prompt. The degree of influence of the input image on the output depends on factors such as the strength parameter and the complexity of the prompt. The higher values mean more noise will be added to the input image, so the result will be further from the input image. The Lower values mean less noise is added to the input image, so

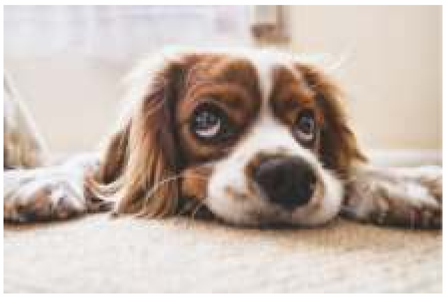


Fig.4: Input Image

the output will be closer to the input image.

For example, the output for prompt "A dog with sunglasses, looking at the camera highly detailed, ultra sharp, cinematic, 100mm lens, 4k resolution" and input image as in Fig. 4. with strength = 0.5 is shown in Fig. 5.

Visual Transformation during:

The model transforms the input image based on the semantic information provided in the prompt, resulting in a visually altered output image. This transformation may involve changes in appearance, style, or context to align with the specified prompt.

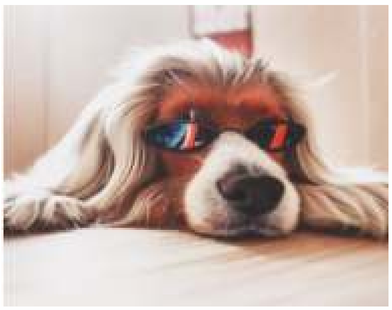


Fig.5: Image Generated In Image-to-Image Transformation

V. Conclusion

Stable image diffusion techniques have emerged as powerful tools in the realm of image processing and computer vision by providing effective solutions for noise reduction, edge preservation image restoration, etc. Overall, the ongoing advancements in stable image diffusion hold promise for addressing a wide range of image processing challenges and contributing to the development of innovative solutions for real-world applications

Stable diffusion holds a lot of promise for the future of creative endeavors and problem-solving, and its open-source nature fuels even more potential. Here are some exciting possibilities for how stable diffusion can be applied in various sectors like Concept Art & Prototyping, Data Visualization, Rapid Brainstorming, Personalized Report Elements, 3D Modeling & Animation, Fine-Tuning & Control, and Ethical Considerations.

Acknowledgment

We take this opportunity to acknowledge our supervisor and mentor, Prof. Sanjay Kumar Giri, for their invaluable guidance, support, and encouragement throughout this study.

VI. References

[1] Maccarini, Andrea M., Andrea M. Maccarini, and Otten. Deep change and emergent structures in global society. Dordrecht: Springer, 2019.

[2] R. Rombach1, A. Blattmann1, D. Lorenz1, P. Esser, B. Ommer. “High Resolution Image Synthesis with Latent Diffusion Models”, Ludwig Maximilian University of Munich & IWR, Heidelberg University, Germany, pages 3-5.

[3] Kobyzev, Ivan, Simon Prince, and Marcus A. Brubaker. "Normalizing flows: Introduction and ideas." stat 1050 (2019): 25.

[4] Lamb, Alexander. "Generative models: a critical review." (2018).

[5]. Vandekerckhove, Joachim, Francis Tuerlinckx, and Michael D. Lee. "Hierarchical diffusion models for two-choice response times." Psychological methods 16.1 (2011): 44.

[6] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, R. Rombach, “SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis”, 2023.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin. “Attention Is All You Need”, 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 2023, pages 2-5.

[8] T. Zhang, Z. Wang, J. Huang, M. M. Tasnim, W. Shi. “A Survey of Diffusion Based Image Generation Models: Issues and Their Solutions”, Conference'17, July 2017, Washington, DC, USA, 2023, page 2.

[9] J. Ho, A. Jain, and P. Abbeel. “Denoising diffusion probabilistic models”, 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada., 2020, pages 2-4.

[10]https://github.com/hkproj/pytorch-stable-diffusion

[11]A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin. “Attention Is All You Need”, 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 2023, pages 2-5.

Rohit Kumar Gupta, Ishika Agarwal,
Hrushikesh Mohanty
7th Sem., Dept. of CSE

AI found a new way to create quantum entanglement

AI found a new way to entangle particles of light

The researchers at the Max Planck Institute for the Science of Light in Germany made a surprising breakthrough when they discovered a novel method to establish quantum entanglement for particles of light. This discovery has the potential to simplify the process of constructing quantum information networks. Because of advances in artificial intelligence, quantum entanglement has recently become less difficult. The PyTheus procedure is a new method that was found by researchers for the purpose of establishing quantum connections between particles. This method has the potential to be utilized in the construction of quantum communication networks in the future.

Source: newscientist.com

Sarvam-1: First Indian language LLM

A novel language model that has been specially trained for Indian languages has been introduced by Sarvam AI, a new entrant in the generative AI market in India. In addition to English, the new open-source Sarvam-1 AI model can understand up to ten Indian languages, including Bengali, Gujarati, Hindi, Kannada, Malayalam, Marathi, Oriya, Punjabi, Tamil, and Telugu. In August of this year, the Bengaluru-based startup introduced Sarvam 2B, its first fundamental AI model. Because Sarvam-1 "demonstrates that careful curation of training data can yield superior performance even with a relatively modest parameter count," it asserts Sarvam-1 is unique. Two billion parameters were used in the development of the recently published AI model. The number of parameters is frequently used to show how complicated an AI model is and how well it can translate inputs into outputs. Microsoft's Phi-3 Mini, for example, has 3.8 billion parameters. In contrast to large language models (LLMs) like OpenAI's GPT-4, which have more than a trillion parameters, AI models like Sarvam-1 and Phi-3 Mini are classified as small language models (SLMs), which have fewer than ten billion parameters. By utilizing fewer tokens per word than earlier LLMs, Sarvam-1-1 is reportedly more effective at processing scripts in the Indic language.

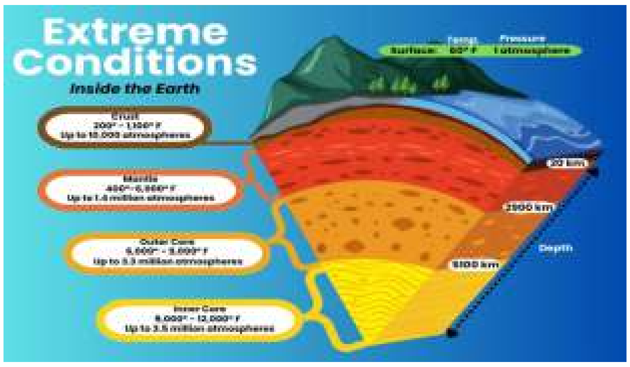
(Source: The Indian Express)

Granite 3.0 AI models

IBM's annual TechXchange featured the introduction of Granite 3.0. The new models were created to offer a blend of autonomy, flexibility, and performance that, on a variety of metrics, either matches or surpasses comparable models of comparable size from top suppliers. General-purpose language models, guardrails and safety models, and mixture-of-experts models are all part of the Granite 3.0 family. The new models perform well on important enterprise tasks like summarization, categorization, and Retrieval Augmented Generation (RAG).

(Source: campustechnology.com)

Iron Monoxide Heating Earth:



Scientists have found that iron monoxide is essential for carrying heat from the Earth's core to the surface, which affects volcanic activity and tectonic movements. Conditions get more severe as one descends from the Earth's surface through the semi-solid mantle and approaches the liquid outer core. Pressures increase to 1,200 times the depth of the deepest oceans, while temperatures climb to roughly half the sun's surface temperature. Under these circumstances, scientists found iron monoxide, a mineral thought to be plentiful in some locations along the mantle-core border, to have peculiar properties. Based on geological evidence and theoretical modeling, researchers discovered that iron monoxide enters a state known as a quantum critical state, where it possesses both conductor and insulator characteristics, close to the Earth's outer core and mantle boundary. This takes place in a transition zone between the core and the mantle and offers a path for heat to rise through the mantle and reach the surface from the core. Iron monoxide may act as a kind of gatekeeper between the Earth's core and mantle. It is a crucial discovery since life on the surface may be impacted if that gate opens and heat escapes.

(Source: SciTech Daily)

Sleep Disruption Could Increase Child's Autism Risk

The sleep of life is essential from the moment of birth. The synapses of neurons are crucial for learning,

attention, working memory, and long-term memory. These neurons grow and link with one another during sleep, establishing brain functions for the rest of one's life. The brain and behavior may suffer if this crucial process is disturbed, either by continual wakefulness or separation anxiety. Autism Spectrum Disorder (ASD), attention-deficit hyperactivity disorder, and intellectual disability are among the neurodevelopmental disorders for which sleep problems are a significant early signal. Researchers and doctors could diagnose ASD earlier and develop new treatment approaches if they had a better understanding of the connections between sleep and the illness. Throughout life, but particularly throughout brain development, sleep is essential. According to a new study headed by neuroscientist Guillaume Dumas, autistic and neurotypical people's brain synchronization during social interactions is less strong than that of two neurotypical people. According to this, autism should be seen as a relational disorder that emphasizes social connections and shared accountability. According to Dumas, "impaired social interaction is bidirectional." When someone with autism is difficult for a neurotypical person to understand, we never suggest that they have a social cognition deficiency. Responsibility is on all parties when an interaction is more challenging.

(Source: The Indian Express)

In-Memory Computing

An international group of electrical engineers has created a novel approach to photonic in-memory computing for the first time. It is optical computing. The photonic memory for AI processing has been constrained by the need for speed and energy consumption. Optical memory has not yet been able to integrate non-volatility, multi-bit storage, fast switching speed, low switching energy, and high durability in a single platform. A new technology is discovered that can be directly programmed with a CMOS (complementary metal-oxide semiconductor) circuit, integrated into computers for a faster, more efficient, and scalable optical computing architecture. It can have 2.4 billion switching cycles and nanosecond speeds. A resonance-based photonic architecture can achieve photonic in-memory computing. The photonic processing commonly multiplies a rapidly changing optical input vector by a matrix of constant optical weights. However, it has been difficult to encode these weights on-chip using conventional techniques and magneto-optical materials.

(Source: The Indian Express)

Chittaranjan Mohapatra
Dept. of CSE

Integrated System Optimization for Enhanced Performance of a Physical Model: A MATLAB and Python Approach

Abstract *– This work focuses on the development and optimization of a temperature control system utilizing a Proportional integral derivative (PID) controller with the aid of MATLAB and Python integration. The primary objectives encompass three key stages. Firstly, the implementation of PID controllers using MATLAB's toolbox tuning capabilities establishes a robust foundation for temperature control system design. Secondly, Fuzzy Inference Systems (FIS) will be employed to fine-tune the PID parameters (proportional gain-, integral gain- & derivative gain-), providing adaptability to varying system conditions and aiming to enhance system responsiveness and stability. Lastly, an innovative approach is introduced by integrating Python code into MATLAB Simulink for Fuzzy Logic Controller (FLC). In the initial stage, the PID parameters are automatically tuned using the MATLAB Tuner App. This stage aims to establish a basic system foundation. Moving to the second stage, a Fuzzy Logic Controller (FLC) is integrated with the PID controller to further enhance system performance in terms of key metrics such as overshoot, settling time, and rise time. Transitioning to the third stage involves replacing the FLC with a MATLAB Function Block in MATLAB Simulink. By incorporating Python scripts, advanced algorithms or computations can be seamlessly integrated into the control system design, expanding its capabilities and allowing for more sophisticated control strategies or analysis techniques to be employed.*

Keywords *– PID, FIS, FLC, settling time, rise time.*

I. Introduction

In optimization, the goal is to enhance the performance, stability, and efficiency of control systems, aiming to minimize errors and meet predefined standards. This process is crucial across various domains, from factories to airplanes to robots, where control systems play a pivotal role. Optimization involves employing diverse techniques to refine control systems, ensuring they operate smoothly and effectively. It encompasses more than just minor adjustments; it entails employing various methods and strategies to maximize a control system's capabilities, especially in challenging circumstances.

The primary objective is to extract the optimal performance from a control system while ensuring its resilience against unexpected disruptions. Engineers achieve this by fine-tuning the system's functionality and conducting thorough testing to strike a balance between accuracy, speed, and stability. As technology advances, the importance of optimizing control systems grows even more pronounced. Whether it's streamlining factory operations or enhancing the safety of self-driving cars, the efficacy of control systems holds significant implications. Thus, this report delves into techniques for optimizing control systems to achieve remarkable results across diverse scenarios, emphasizing faster response times and system stability as key objectives. Through this exploration, we gain insights into leveraging control systems to accomplish remarkable feats across various applications.

There are several methods for optimizing a physical system to improve its performance. The major includes PID tuning, fuzzy optimization controllers, Model Predictive Control (MPC) optimization controllers, and metaheuristic algorithms as shown in Fig. 1.

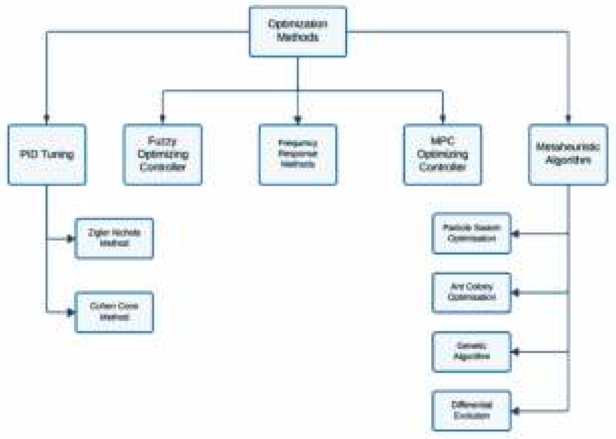


Fig. 1. Major Optimization Techniques

II. Background

A. PID Controller

PID parameters refer to the three essential coefficients used in a Proportional-Integral-Derivative (PID) controller: proportional gain (), integral gain (), and derivative gain (). These parameters play a crucial role in determining the controller's response to changes in the system it is regulating [1]. The proportional gain determines the magnitude of the controller's response to the current error, the integral gain helps eliminate steady-state errors by considering past errors over time, and the derivative gain anticipates future error trends and dampens oscillations. Tuning these parameters effectively is essential for achieving the desired system performance, and striking a balance between stability, responsiveness, and robustness. The mathematical representation of a Proportional-Integral-Derivative (PID) controller is written in equation (1).

$$pid(t) = K_p + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \quad (1)$$

B. Fuzzy Logic Controller (FLC)

A Fuzzy Logic Controller (FLC) is a type of control system that operates based on fuzzy logic principles, which allow for reasoning and decision-making in situations with uncertainty or imprecision. FLCs have found widespread applications in various fields, due to their ability to handle nonlinearities and uncertainties effectively while offering flexible and intuitive control strategies [2].

The advantages of fuzzy logic controllers over PID controllers, particularly in managing nonlinear and uncertain processes like water level control. While PID controllers have limitations in highly nonlinear systems, fuzzy control offers effective handling of such complexities through heuristic rules derived from expert knowledge. Fuzzy logic controllers emulate human-like reasoning, drawing upon the expertise of control engineers and plant operators, making them versatile in various power plants and systems.

C. Temperature Control System

A temperature control system serves as a crucial component in various fields, including household, industrial, and research applications, aiming to regulate the temperature of a system or environment within a desired range. Particularly in industrial settings, temperature control systems are essential for maintaining optimal conditions for processes such as heating, cooling, or chemical reactions [4].

Temperature control systems are integral components in various industrial processes. Typically, these systems involve regulating the temperature of electric ovens or heaters in response to reference inputs and control signals. The step response of this temperature control system is obtained from the experimental data. The transfer function of the system is given by equation (2).

$$G(s) = \frac{104e^{-2.6s}}{1+15.784s} \quad (2)$$

III. Methodology

A. The Implementation of PID Controllers Using MATLAB Toolbox Tuning

The MATLAB Simulink model for a Temperature Control System with a PID Controller provides a comprehensive platform for analyzing and optimizing temperature regulation in dynamic environments. By simulating the system's response to a step input, engineers can discern the efficacy of employing a PID controller compared to an open-loop system. The model showcases two distinct responses: one representing the system's behavior without the PID controller, and the other depicting the system's response with the PID controller integrated shown in Fig. 2. This setup facilitates a direct comparison between the two scenarios, elucidating the benefits of employing PID control in terms of response time, overshoot, settling time, and overall stability.

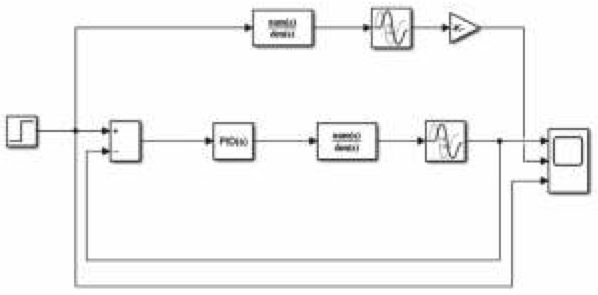


Fig. 2. Simulink Block Diagram for System Analysis

B. The Implementation of FLC to Tune PID Parameters

Implementing a Fuzzy Logic Controller (FLC) to tune PID parameters involves several sequential steps. Firstly, the FLC is designed and developed using Fuzzy Inference Systems (FIS) within the MATLAB environment. This entails defining linguistic variables, membership functions, and fuzzy rules that capture the system's behavior and control objectives. Subsequently, the FLC is integrated with the existing PID controller structure, forming a combined PID-FLC control system [5]. During this integration phase, the FLC's output serves as an input to dynamically adjust the PID parameters (K_p, K_i, K_d) in real time based on the current system conditions. To facilitate this integration, appropriate interfacing mechanisms are established between the FLC and the PID controller in MATLAB Simulink, ensuring seamless communication and parameter adjustment as shown in Fig. 3. Once integrated, the control system is simulated or implemented to evaluate its performance under various operating conditions. Performance metrics such as overshoot, settling time, and rise time are analyzed to assess the effectiveness of the PID parameters tuned by the FLC.

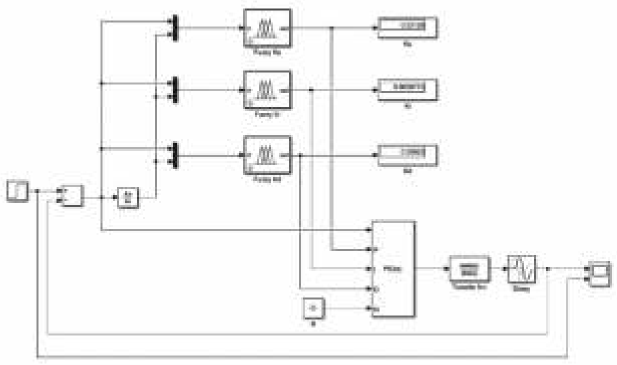


Fig. 3. Simulink of FLC Tuning PID Controller

In MATLAB, a Fuzzy Inference System (FIS) file is a data file that encapsulates all the information required to define and operate a fuzzy logic system. This file typically contains the fuzzy membership functions, fuzzy rules, and fuzzy inference engine configurations necessary for performing fuzzy logic-based computations and decision-making processes. FIS files facilitate the representation, storage, and sharing of fuzzy logic systems, allowing users to easily deploy and integrate them into their MATLAB applications for tasks such as control, modeling, and pattern recognition. To create an FIS file, we have to type fuzzy in the command window of MATLAB. This command opens the Fuzzy Logic Designer app where we can define and configure our fuzzy logic system. Once we're done, we can save the FIS file. To call that file we need to type "fuzzy FIS_file_name".

The input range for error and change in error is [-1, 1], and the output range for K_p, K_i, and K_d are [0, 0.0701], [0, 0.0018], [0, 0.1185] respectively as shown in Fig. 4-8. Membership functions play a crucial role in both the fuzzification and defuzzification processes of a Fuzzy logic system. Membership functions are vital in a Fuzzy logic system's fuzzification and defuzzification processes. They quantify linguistic terms, enabling the representation of fuzzy concepts in a numerical form. For clarity, the linguistic terms represented by membership functions have been illustrated through temperature variations.

In a fuzzy control system shown in Fig. 3, the input variables are associated with sets of membership functions known as 'fuzzy sets". Fuzzification, as a method, establishes the connection between a crisp input value and its corresponding fuzzy value. These fuzzy sets are defined for two input variables: "Error (E)" and "Changing Error (CE)," while the output variable represents K_p, K_i, and K_d. Each input parameter's domain is partitioned into seven fuzzy sets, namely NB (Negative Big), NM (Negative Medium), NS (Negative Small), ZO (Zero), PS (Positive Small), PM (Positive Medium), and PB (Positive Big), with corresponding membership functions. These membership functions map input parameters to seven membership values defined over the seven subsets. The output domain is similarly divided into seven sub-domains (NB, NM, NS, ZO, PS, PM, PB).

The Fuzzy Logic Control (FLC) methodology is structured around a sequential process encompassing input, processing, and output phases. This methodology relies on two primary inputs: the error ('E') and the changing error rate ('CE') derived from the process under consideration. By applying fuzzy rules, the system generates fuzzy presumptions, which are subsequently utilized in defuzzification. During defuzzification, the system extracts three crucial parameter values alongside an enhanced Proportional-Integral-Derivative (PID) function. These parameters, upon aggregation, contribute to the formulation of the three distinct components inherent in the enhanced PID function. Implementing fuzzy logic rules governs the manipulation of output variables, employing a set of IF-THEN conditions, wherein the antecedent section represents the "preceding" condition, while the consequent section signifies the "consequential" action. The fuzzy rules are given in Table 1-3.

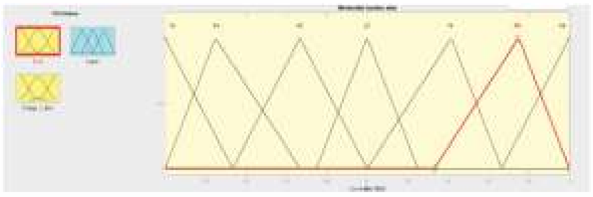


Fig. 4. Membership Functions for Error

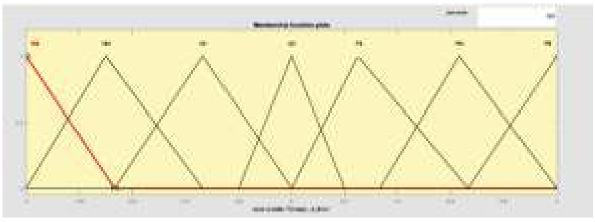


Fig. 5. Membership Functions for Change in Error

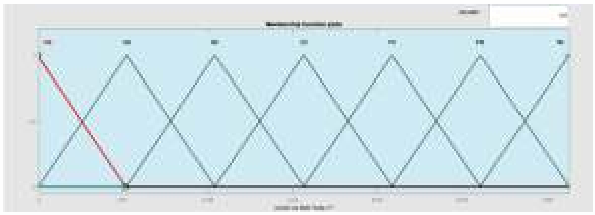


Fig. 6. Membership Functions for K_p

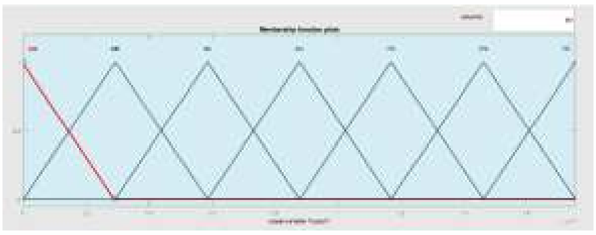


Fig. 7. Membership Functions for K_i

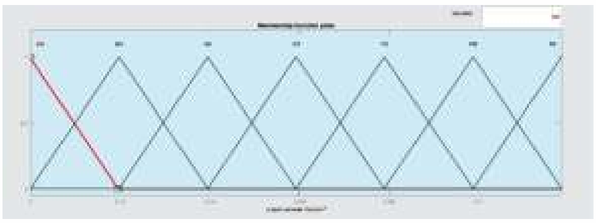


Fig. 8. Membership Functions for K_d

E/CE	NB	NM	NS	ZO	PS	PM	PB
NB	NM	PB	PB	PM	PM	ZO	ZO
NM	PB	PM	PM	PS	PS	ZO	NM
NS	NM	PM	PM	PS	ZO	NS	NM
ZO	PM	PM	PS	ZO	NS	NM	NM
PS	PS	PS	ZO	NS	NS	NM	NM
PM	PS	ZO	NS	NS	NS	PM	NM
PB	ZO	PS	NM	NM	NM	NB	NB

Table - 1: Fuzzy rules for K_P

E/CE	NB	NM	NS	ZO	PS	PM	PB
NB	NB	NM	NM	PS	ZO	ZO	ZO
NM	PS	NB	NS	NS	NS	ZO	NS
NS	NM	NS	PS	NS	ZO	NS	NS
ZO	NM	NS	NS	ZO	PS	NS	NS
PS	NS	NS	ZO	PS	PM	NM	PM
PM	ZO	NS	PS	PS	PB	NB	NB
PB	ZO	PS	PM	PM	PM	NB	NB

Table - 2: Fuzzy rules for K_I

E/CE	NB	NM	NS	ZO	PS	PM	PB
NB	NB	NS	NM	NM	ZO	ZO	ZO
NM	NM	NS	NB	PM	NS	ZO	NS
NS	PS	NS	NM	NS	ZO	NS	NS
ZO	ZO	NM	NS	ZO	PS	NM	NM
PS	PS	ZO	ZO	NS	PS	NM	NM
PM	PM	PB	NS	PS	PM	PM	NB
PB	PB	PB	PM	PM	PM	NB	NB

Table - 3: Fuzzy rules for K_D

C. Python Integration in MATLAB

Integrating Python functionality into MATLAB Simulink can be achieved seamlessly through the MATLAB Function block. This block acts as a bridge between Simulink and Python, facilitating bidirectional communication and enabling the utilization of Python scripts within Simulink models.

Python modules are called within the MATLAB Function block using the py. Prefix, enabling direct invocation of Python functions or classes. This approach ensures seamless interoperability between MATLAB/Simulink and Python environments, empowering engineers to leverage the strengths of both platforms for comprehensive system modeling and simulation tasks.

This MATLAB function block, taking two inputs representing the error and the change in error, and generating three outputs corresponding to the Proportional (K_p), Integral (K_i), and Derivative (K_d) parameters, acts as an intermediary between the Python code and the external PID controller shown in Fig. 9. By encapsulating the complex Python algorithms within this function block, the system effectively mimics the behavior of an FLC, providing enhanced adaptability and control refinement.

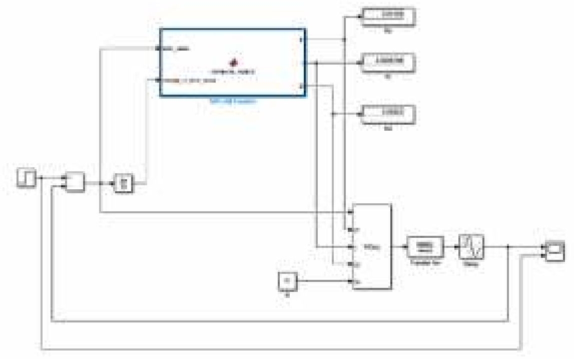


Fig. 9. Simulink of MATLAB Function Block Tuning PID Controller

IV. Results

A. Temperature Control System with PID Controller

The plot, shown in Fig. 10, depicts a visualization of the improved performance of the system with the presence of a PID Controller. Here, we can see step response with PID gives a faster response as compared to step response without PID.

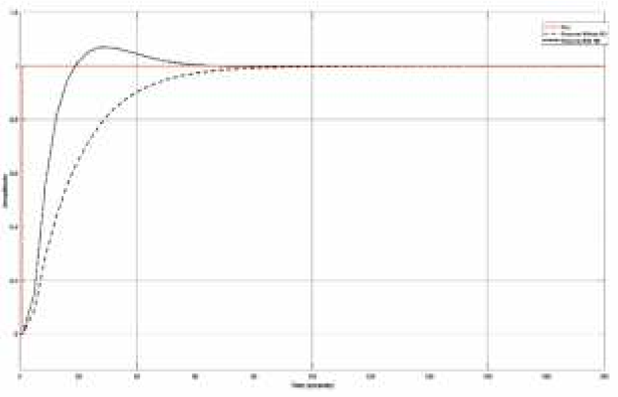


Fig. 10. Step Response of The System

B. Temperature Control System with Fuzzy PID Controller

By incorporating FLC alongside PID control, the system aims to mitigate issues such as overshoot and oscillations, typically encountered in conventional PID control setups. In the MATLAB scope, engineers can observe that while the PID controller may exhibit a noticeable overshoot, the combined Fuzzy-PID controller significantly reduces this overshoot to a negligible level shown in Fig. 11.

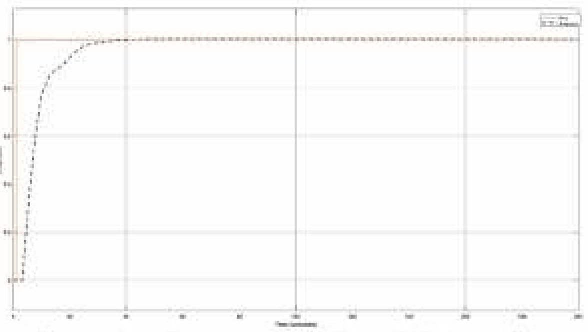


Fig. 11. Step Response of the Fuzzy PID Controller

C. Temperature Control System with Python Integration in MATLAB Simulink

The similarity in step response plots (Fig. 11 & Fig. 12) underscores the equivalence in control performance achieved through both FLC and Python integration methodologies. This shows that using Python with MATLAB Simulink works well as a replacement for Fuzzy Logic Controller. It makes things faster to compile without making the control less effective.

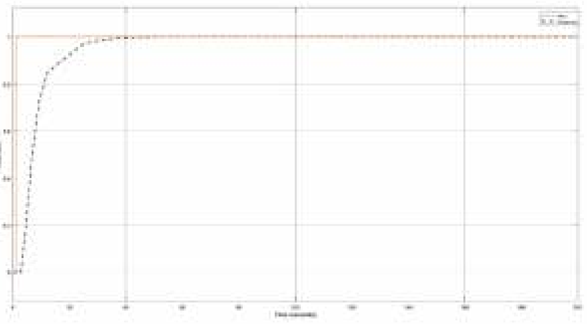


Fig. 12. Step Response of the System for Python Integration with MATLAB

D. Comparison

After our analysis, a comprehensive comparison across various parameters namely, proportional gain (Kp), integral gain (Ki), derivative gain (Kd), overshoot, settling time, and rise time was conducted across three scenarios shown in Table 4.

The optimization analysis [6] reveals that employing Fuzzy PID control yields superior performance metrics compared to traditional PID tuning. Settling time, rise time, and overshoot are notably reduced, showcasing the efficacy of Fuzzy PID integration.

Parameters	Without PID	With PID	With Fuzzy PID	With Python Integration
K_p	-	0.01506	0.03128	0.03128
K_i	-	0.00144	0.00087	0.00088
K_d	-	0.00839	0.05922	0.05922
Rise Time	34.6 sec	10.8 sec	9.9 sec	9.9 sec
Settling Time	64.3 sec	48.4 sec	31.1 sec	31.1 sec
Overshoot	0 %	6.65 %	0.15 %	0.15 %

Table – 4: Comparison of Different Techniques

V. Conclusions

In conclusion, the development and optimization of the temperature control system through the integration of MATLAB and Python has demonstrated significant advancements in control system design. Through the utilization of PID controllers and Fuzzy Inference Systems (FIS), this study has successfully addressed the challenges of adaptability and responsiveness in varying system conditions. The implementation of MATLAB's tuning capabilities for PID controllers provided a robust foundation, while the integration of Python code into MATLAB Simulink expanded the system's capabilities, allowing for more sophisticated control strategies to be employed.

References

[1] Zuraida Muhammad, Syahmin Hamka Ahmad Dziauddin, Shabinar Abd Hamid, and Nor Adni Mat Leh. "Water Level Control in Boiler System Using Self-tuning Fuzzy PID Controller". In: 2023 IEEE 13th International Conference on Control System, Computing and Engineering (ICCSCE). IEEE. 2023, pp. 232–237

[2] Fansheng Meng, Xuefei Zhang, Yan Zheng, Xi Cheng, and Zhi Weng. "Fuzzy control and simulation of boiler drum water level". In: MATEC Web of Conferences. Vol. 309. EDP Sciences. 2020, p. 05003

[3] Md Mizanur Rahman and Md Saiful Islam. "Design of a fuzzy based PID algorithm for temperature control of an incubator". In: Journal of Physics: Conference Series. Vol. 1969. 1. IOP Publishing. 2021, p. 012055.

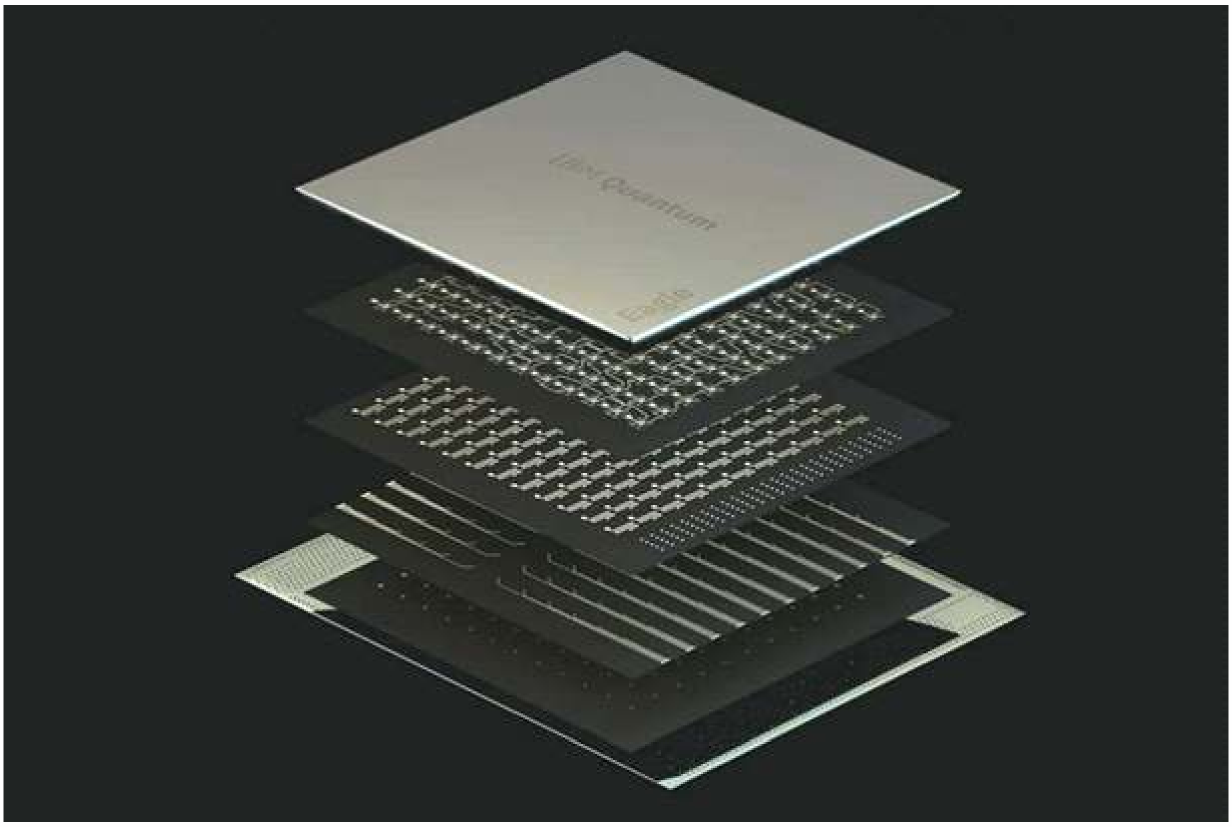
[4] Yugal K Singh, Jayendra Kumar, Keshav K Pandey, K Rohit, and A Bhargav. “Temperature control system and its control using PID controller”. In: Int. J. Eng. Res. Technol 4.02 (2016), pp. 4–6.

[5] Wei Jiang and Xuchu Jiang. “Design of an intelligent temperature control system based on the fuzzy self-tuning PID”. In: Procedia Engineering 43 (2012), pp. 307–311.

[6] Weiguo Zhao, Liying Wang, Zhenxing Zhang, Honggang Fan, Jiajie Zhang, Seyedali Mirjalili, Nima Khodadadi, and Qingjiao Cao. “Electric eel foraging optimization: A new bio-inspired optimizer for engineering applications”. In: Expert Systems with Applications 238 (2024), p. 122200.

Subham Kumar Das, Sanket Kumar Nayak
7th Sem., Dept. of EEE

IBM entangled two quantum chips to work together for the first time



A major step towards IBM's (International Business Machines Corporation) modular strategy for developing quantum computers has been the successful linking of two quantum chips to function as a single device. Combining the processing capability of two quantum computers into a single, larger machine is a huge step forward for IBM's quest for ever-more-powerful quantum computers. This outcome bodes well for the viability of the company's strategy to scale up quantum computers through modularity. Although quantum computers have great promise for outperforming traditional systems in specific areas, there are still significant challenges to overcome, such as increasing computer size while simultaneously decreasing error rates.

Source: newscientist.com

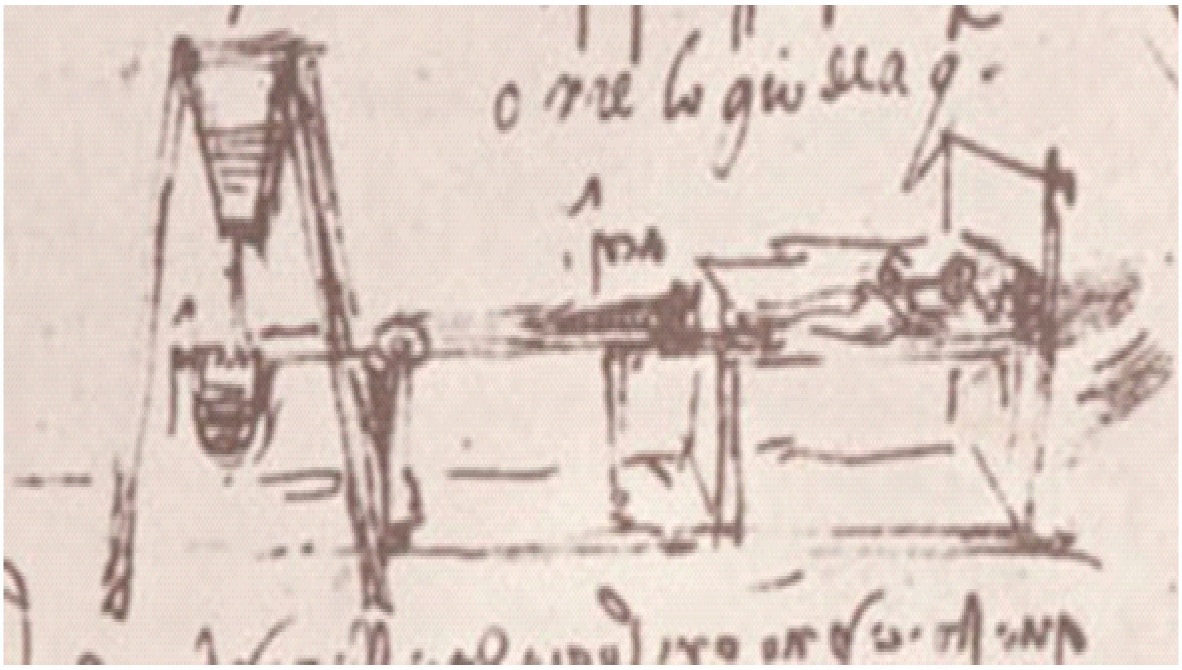
Shreeram Shankar Abhyankar: A Leading Figure in Algebraic Geometry

Shreeram Shankar Abhyankar (1930–2012) was a prominent mathematician, highly respected for his substantial contributions to the field of algebraic geometry, particularly in the area of singularity resolution. Born in Ujjain, India, Abhyankar began his education in India and later moved to the United States to complete his Ph.D. at Harvard University under the guidance of the renowned mathematician Oscar Zariski. His doctoral research on local uniformization of algebraic surfaces over ground fields of characteristic p became a cornerstone in the study of algebraic geometry. One of Abhyankar's most significant accomplishments was solving the problem of singularity resolution in positive characteristics, a challenge that had baffled many before him. His innovative work expanded the boundaries of algebraic geometry and introduced new methodologies for tackling complex mathematical challenges. Beyond his work on singularities, he also made valuable contributions to group theory, valuation theory, and the Jacobian conjecture. In addition to his research, Abhyankar was a passionate teacher and mentor. He spent the majority of his academic career at Purdue University, where he held the position of Marshall Distinguished Professor of Mathematics. Throughout his tenure, he mentored numerous students, nurturing the development of future mathematicians. Abhyankar maintained close ties with India and played a crucial role in promoting mathematical research and education in his home country. He was a fellow of the Indian National Science Academy and received the prestigious Chauvenet Prize in 1978. Shreeram Abhyankar's impact on mathematics continues to be felt within the academic world, cementing his legacy as a trailblazer in algebraic geometry. His work remains highly influential, shaping modern mathematical thought and inspiring further research in the field.

Source: en.wikipedia.org

Abhishikta Sahoo
Dept. of EE

The Leonardo da Vinci Alarm Clock



Leonardo da Vinci (1452–1519) was a prolific inventor and unquestionably one of the greatest innovative minds of all time. His scientific forays spanned diverse domains, including aerodynamics, hydraulics, human anatomy, and weapons systems. One of his more esoteric inventions had to do with his somewhat unusual and distinctive sleep pattern, called polyphasic sleeping. It is believed that he took power naps of around 20-30 minutes duration, followed by 3-4 hours of staying awake; this resulted in a total sleep time of around 2-3 hours per day.

To allow himself to remain awake and continue with his rigorous routine, he designed an alarm clock. The concept was based on falling water and included controlled amounts of water that were dropped into a reservoir, which when full would activate several levers that propped his legs upwards, waking him up. The fluid flow was carefully monitored in this invention. One has to remember that this clock was at least a hundred and fifty years prior to the invention of an accurate timekeeper in the form of a pendulum clock, in the mid-1600s. His plans and sketches for this device can be found in his Codex Madrid, a portion of which is shown above.

Content excerpted from web sources, including https://www.davincilife.com/alarm-clock.html

Prof. Jaideep Talukdar
Dept. of BSH

Study and Analysis of Maize Diseases Using Image Processing Techniques

It's challenging to envision life without agriculture, akin to trying to breathe without air. Agriculture isn't solely about providing sustenance but also profoundly influences employment opportunities and income worldwide. The agricultural sector stands as a cornerstone of economic progress for numerous nations. Countries like India, China, the United States, Brazil, Mexico, Russia, Japan, Germany, and France all rely heavily on agriculture for economic development. Plant disease detection, identification, and classification are vital aspects of modern agricultural practices, serving as linchpins for food security, resource optimization, and meeting the needs of a growing global population. Accurate plant disease identification empowers farmers and agribusinesses to make well-informed decisions regarding crop management, such as planting strategies, irrigation needs, and pest control methods. Leveraging image processing techniques enables precise identification of various disease varieties, empowering farmers to tailor their approaches according to each crop type's unique requirements and attributes. This boosts overall agricultural efficiency and fosters sustainability by reducing the use of wrong pesticides and mitigating environmental impact.

Maize, commonly referred to as corn, serves as a staple food for humans and animals, while also serving as a key ingredient in various industrial products. However, meeting the dietary needs of the world's expanding population presents significant challenges due to climate fluctuations, limited resources, and the prevalence of various diseases affecting maize plants, including fungi, bacteria, viruses, and mollicutes. Detecting plant infections swiftly is crucial to minimize the risk of crop failure and financial losses. Leveraging image processing techniques with computer intelligence is integral to early disease detection, enabling farmers to apply targeted treatments and avoid inadvertent harm to plants through misapplied pesticides. Machine learning and deep learning-based classification methods are increasingly prominent in research, offering advantages in feature extraction and automated learning. These methodologies are being applied not only to identify and classify diseases affecting maize plants but also to address similar challenges in other plant species.

This thesis presents a thorough and innovative examination of cutting-edge technologies within the agricultural sector. Here, we introduce the stage by underlining the pivotal role of automatic disease detection in modern farming, with a focus on precision, efficiency, and sustainability. The thesis concentrates on the detection of leaf diseases in maize crops, leveraging state-of-the-art image processing techniques to develop robust algorithms for the precise detection of defective maize leaves. Extending these methods to classify different maize leaf diseases, with the design of a hybrid-optimized classification model. This thesis also focuses on the automated classification of stem diseases in maize crops by using a deep feature extraction-based model, further enhancing agricultural efficiency. To this end, this thesis synthesizes the collective findings, elucidating the overall impact on the field, highlighting agricultural implications, and suggesting avenues for future research.

Dr. Arabinda Dash
Dept. of CSE

Accurate Structure Prediction of Biomolecular Interactions with AlphaFold 3

Introduction

The revolutionary capabilities of AlphaFold have transformed the field of protein structure prediction, dramatically improving our understanding of biomolecular interactions. The latest iteration, AlphaFold 3 (AF3), builds on the success of its predecessors by incorporating a novel diffusion-based architecture. This article discusses the advancements of AF3 in predicting the joint structure of complex biomolecules such as proteins, nucleic acids, small molecules, and modified residues, along with its performance in diverse biomolecular interactions.

The Evolution of AlphaFold

Since its introduction, AlphaFold has achieved significant breakthroughs in predicting protein structures with remarkable accuracy. AlphaFold 2 (AF2) expanded on this by providing reliable predictions for protein interactions through structural modeling techniques. AF3 takes a further leap by addressing more complex interactions within a unified deep-learning framework. The model now integrates the ability to handle a wider range of biomolecular types, moving beyond proteins to nucleic acids and ligands.

Key Advancements in AlphaFold 3

AlphaFold 3 introduces several technical enhancements that allow it to model more complex biological systems. The shift to a diffusion-based architecture replaces the previous focus on protein frames and torsion angles with direct prediction of atomic coordinates. This change enables AF3 to handle arbitrary chemical components and predict high-accuracy structures across various biomolecular categories, including protein-ligand, protein-nucleic acid, and antibody-antigen interactions.

One of the significant changes is the reduction in the complexity of multiple-sequence alignments (MSA). By replacing the AF2 former module with a performer module, AF3 reduces the need for extensive MSA processing, resulting in greater flexibility and generalization across different biomolecular complexes.

Performance Across Biomolecular Complexes

AF3 delivers substantial improvements in accuracy compared to specialized tools that focus on specific types of biomolecular interactions. For protein-ligand interactions, AF3 outperforms traditional docking tools such as AutoDock Vina, while in the realm of protein-nucleic acid interactions, it surpasses nucleic acid-specific predictors. AF3 also demonstrates higher accuracy in predicting antibody-antigen complexes than AlphaFold Multimer v.2.3. These advancements enable AF3 to offer reliable predictions for a wider range of biomolecular interactions.

Network Architecture and Training

Af3 maintains the overall structure of AF2 but with critical enhancements. The pair former module takes center stage in processing pairwise interactions within the chemical complex, while the diffusion module handles direct predictions of atomic positions. The integration of these modules allows AF3 to accommodate diverse biomolecular structures while minimizing the need for stereochemical losses or specialized handling.

Training AF3 involves the use of generative diffusion techniques, which generate a distribution of possible structures. This approach provides local structure accuracy even when the overall model may be uncertain about certain positions. The model is trained to predict 'noised' atomic coordinates, and the iterative denoising process generates highly accurate structural predictions. This architecture is shown in Fig. 1 [a-d].

Challenges and Future Directions

Despite the impressive capabilities of AF3, the model still faces challenges. One significant issue is the tendency for 'hallucinations,' where the model may predict plausible but incorrect structures in unstructured regions. To mitigate this, AF3 uses cross-distillation training, incorporating data from AlphaFold Multimer predictions to reduce hallucinations.

AF3's generative approach also presents limitations in predicting dynamic structures, as the model typically produces static structures. Moreover, while AF3 excels in handling diverse biomolecular types, some targets-such as antibody-antigen complexes-still require a large number of predictions and computational resources to achieve the highest accuracy.

Conclusion

AlphaFold 3 represents a significant step forward in biomolecular modeling, demonstrating that a deep-learning framework can predict highly accurate structures across a wide range of complex interactions. By integrating more chemical diversity and improving overall model efficiency, AF3 opens new doors for the rational design of therapeutics and our understanding of cellular functions. Continued advancements in experimental techniques and deep learning are expected to further enhance the capabilities of models like AF3, driving the field of structural biology toward new frontiers.

Source:https://doi.org/10.1038/s41586-024-07487-w

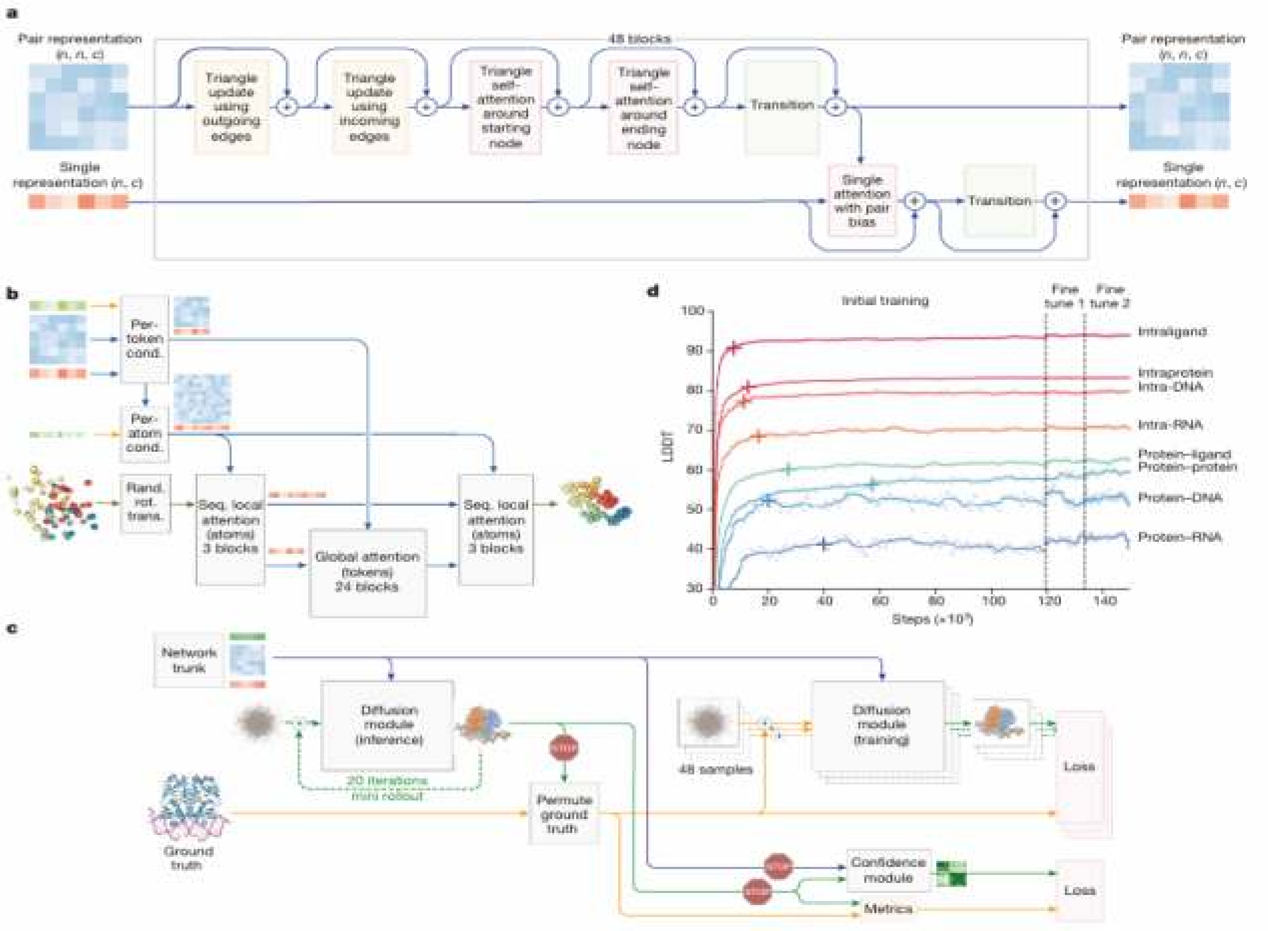

a
Pair representation (n, n, c)
48 blocks
Triangle update using outgoing edges
Triangle update using incoming edges
Triangle self-attention around starting node
Triangle self-attention around ending node
Transition
Pair representation (n, n, c)
Single representation (n, c)
Single attention with pair bias
Transition
Single representation (n, c)
b
Per-token cond.
Per-atom cond.
Rand. rot. trans.
Seq. local attention (atoms) 3 blocks
Global attention (tokens) 24 blocks
Seq. local attention (atoms) 3 blocks
d
Initial training
Fine tune 1
Fine tune 2
LDDT
Steps (×10³)
Intraligand
Intraprotein
Intra-DNA
Intra-RNA
Protein–ligand
Protein–protein
Protein–DNA
Protein–RNA
c
Network trunk
Diffusion module (inference)
20 iterations mini rollout
STOP
Permute ground truth
Ground truth
48 samples
Diffusion module (training)
Loss
Confidence module
Metrics
Loss

Fig. 1 Architectural and training details. a, The pairformer module. Input and output: pair representation with dimension (n, n, c) and single representation with dimension (n, c). n is the number of tokens (polymer residues and atoms); c is the number of channels (128 for the pair representation, 384 for the single representation). Each of the 48 blocks has an independent set of trainable parameters. **b**, The diffusion module. Input: coarse arrays depict per-token representations (green, inputs; blue, pair; red, single). Fine arrays depict per-atom representations. The coloured balls represent physical atom coordinates. Cond., conditioning; rand. rot. trans., random rotation and translation; seq., sequence. **c**, The training set-up (distogram head omitted) starting from the end of the network trunk. The coloured arrays show activations from the network trunk (green, inputs; blue, pair; red, single). The blue arrows show abstract activation arrays. The yellow arrows show ground-truth data. The green arrows show predicted data. The stop sign represents stopping of the gradient. Both depicted diffusion modules share weights. **d**, Training curves for initial training and fine-tuning stages, showing the LDDT on our evaluation set as a function of optimizer steps. The scatter plot shows the raw datapoints and the lines show the smoothed performance using a median filter with a kernel width of nine datapoints. The crosses mark the point at which the smoothed performance reaches 97% of its initial training maximum.

Adarsh Amrit
Dept. of EE

Traffic Signal Recognition using Deep Learning

Abstract *– Traffic signal recognition is essential for the advancement of intelligent transportation systems and autonomous vehicles since it improves road safety and guarantees adherence to traffic regulations. This study aims to utilize deep learning methodologies to effectively identify and categorize traffic lights in practical settings. We use Convolutional neural networks (CNNs) to extract spatial characteristics and achieve robust performance in a variety of scenarios, such as complex backgrounds, obstruction, and illumination variations. This type of neural network takes input from an image, extracts features from an image and provides learnable parameters efficiently for classification, detection, and many more tasks. We intend to provide recognition of traffic signs using convolutional neural networks. The delicacy of the model designed in this paper on the German Traffic Sign Recognition Benchmark (GTSRB) dataset. By using basic convolution and pooling techniques, the network can recognize traffic signs, and it has been tested on the GTSRB (German Traffic Sign Recognition Benchmark) dataset.*

Keywords *– Traffic sign, Computer vision, Classification, TensorFlow, CNN.*

I. Introduction

Traffic sign recognition in intelligent driving systems such as automatic driving and assisted driving plays an important role. Traffic sign recognition methods are split into two categories: manual feature methods and deep learning methods. In the past, traditional recognition methods required manual labelling feature extraction and other feature recognition methods which greatly reduced the speed of system operation. Manual labelling not only increased the workload but also the accuracy rate was difficult to guarantee. Artificial feature learning methods generally use SVM and random forest, but this method contains CNN which is suitable for recognizing images with blurred feature boundaries.

In recent years, the fast development of deep learning has changed detection methods. The research of neural networks has gradually become a liked research field for researchers. The birth and quick application of neural networks can get rid of the laborious manual annotation, and the constructed network can automatically extract the features of the input image. Especially for complex images, the network can get different features, and finally, these features are used for target classification. Thanks to the development of deep learning, the recognition of traffic signs is also developing rapidly.

II. Background

In recent years, CNN has become one of the research hotspots and many students are devoted to this field. Therefore, CNN has gradually become the most common image classification model in computer vision. Usually, a complete CNN includes three basic parts: convolutional layer, pooling layer, and fully-connected layer. The convolutional layer is a very important part of CNN. For example, Bouti et al. [1] modified the LeNet architecture to achieve a good classification effect on the GSTRB dataset. Stallkamp et al. [2] use hinge loss function and loss of function with a network neural network to achieve the recognition of traffic signs. Wu et al. [3] used the two branch networks to enhance the learning target classification capabilities, and the classification effect is better. The convolution kernel convolves the corresponding region of the image with a specified step size and outputs a two-dimensional feature map the image develops from low-dimensional to high-dimensional, and then obtains high-dimensional features of the image. In the article [4] a traffic sign detection and identification method on account of image processing is proposed, which is combined with a convolutional neural network (CNN) to sort traffic signs.

M.T. Islam [5] proposed a system that is able to detect and classify a set of 28 traffic signs in different environments. Compared with traditional machine learning methods, adding convolutional layers [6] can recklessly extract features at different levels in the image, and has translation invariance to the input image. In addition, the convolution mask in the convolution layer is a parameter of the shared, which greatly reduces the size of the parameters. The pooling layer in the convolution process can reduce the image dimension, retain the ability of key information, and speed up the network training process. Common pool methods include maximum pool, average pool, and random pool. No matter which pooling method is used, its main purpose is to reduce the spatial feature dimension, reduce the system load, and speed up the network training speed. At the end of the neural network, there is usually one, or more fully-connected layer. Its function is to be extended to a one-dimensional feature map, use the extracted high-dimensional feature information to classify the image, and use the last fully connected layer as the output layer, and then the network outputs the classification result. In addition, the classification activation function can convert the feature information of the image into the (0, 1) interval, which reduces the computer performance consumed during the training process.

III. Methodology

The desired neural network constructed in this paper is trained on the training set to verify the recognition accuracy of the network on the validation set. According to the results of the validation set, the training is kept on the training set. the accuracy of the network on the test set is tested later.

A.Data Enhancement and Processing

Fig. 1 shows the distribution of 43 classes of the GTSRB dataset. The horizontal coordinate is 43 categories, and the vertical coordinate is the frequency per category. We can see that the distribution of the image dataset is uneven, which makes it easy for the network to classify certain categories accurately, while for other categories the classification effect is inaccurate, so this paper uses data enhancement [4] methods to expand the dataset. The generalization

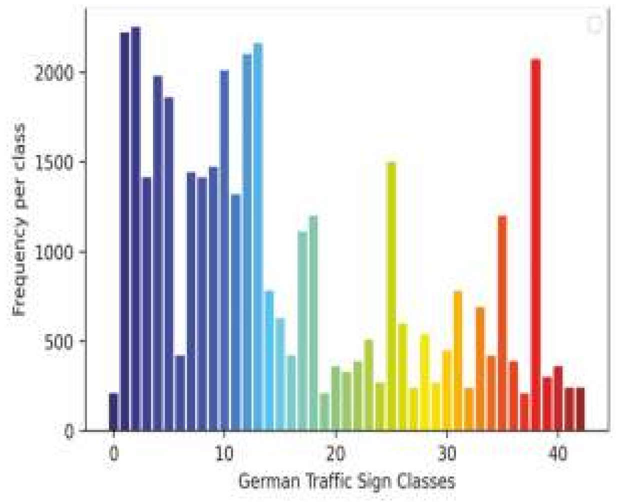


Fig. 1. Dataset distribution

ability of the network and the classification ability of different shooting angles are improved.

We used the Keras library to load each layer. So, we installed TensorFlow before importing deep learning layers. We used the OS module that helps iterate all the images with their respective classes and labels. Pandas is used for data analysis tasks in Python. NumPy is mostly used for working with Numerical values as it makes it easy to apply mathematical functions.

We loaded all the images in a single list in the form of an array. The given list describes the pixels of the image. We made another list containing labels or classes of the corresponding image. To put the image data into the CNN model, we first need to convert the data into a NumPy array.

The training dataset contains the labels or classes from 0 to 42. Thus, with the help of this module, we iterated through each class folder and appended the image to the data list and the respective label to the labels list. We have the CSV files that contain the valid label category name.

B. Network Structure

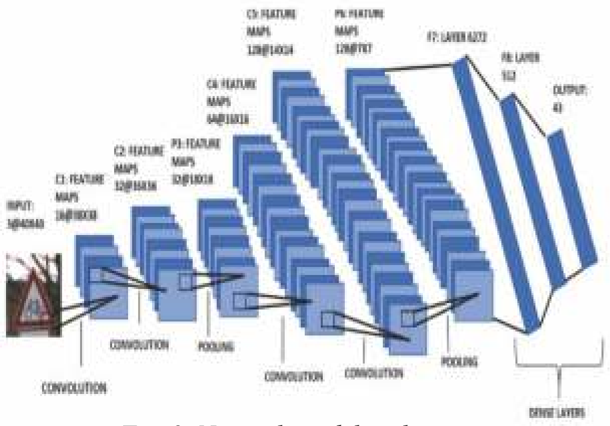


Fig. 2. Network model architecture

A total of 11 layers of convolutional neural networks are constructed in this paper, we call it a three-tier self-interpretable Deep convolutional network(TS-CNN) [6], as shown in Fig. 2. The network is mainly formed of convolutional and pooling layers. By gradually increasing the convolutional layer, extracting the feature map from the input image, using max pooling to reduce the dimensionality of the feature map, and obtaining features at different feature scales by fusing more layers. Finally, the fully connected layer performs a dimensional transformation on the input features and uses soft extremum functions to classify the traffic signs.

C. *Network Details*

1. Dropout

We place Dropout technology in the construction of the network. In the process of forward propagation, this method randomly inactivates neurons with a certain probability P to reduce the scale of parameters and improve the generalization ability of the model. Fig. 3 is a before and after diagram.

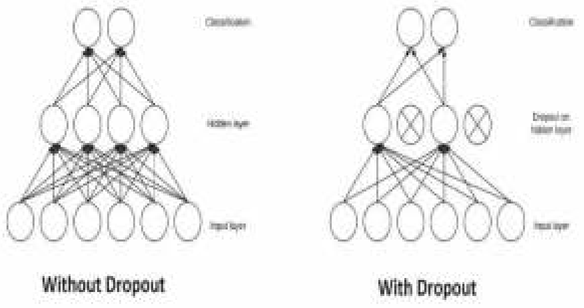


Fig. 3. Without and with Dropout

The left part of Fig. 3 does not use Dropout, and the right part of Fig. 3 uses Dropout, it can be seen that the complexity of the network structure after using Dropout is reduced, which is helpful to improve the network training efficiency and generalization ability.

2. Activation Function

This work uses the common ReLU function and Softmax function. The use of ReLU helps to prevent the exponential growth in the computation required to operate the neural network. If the CNN scales in size, the computational cost of adding extra ReLUs increases linearly. The expression and figure of the function are as follows.

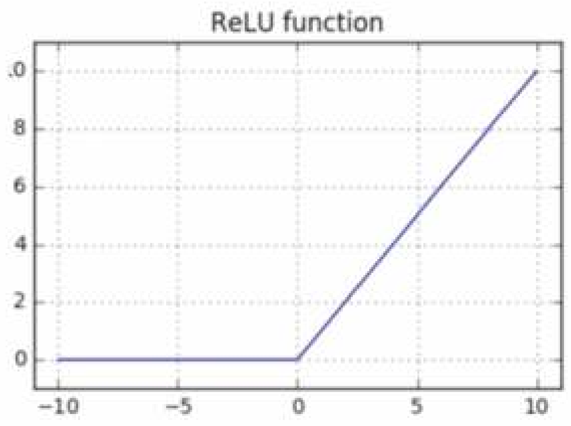


Fig. 4. ReLU function

$$R(z) = \begin{cases} z, & z > 0 \\ 0, & z <= 0 \end{cases} \quad (1)$$

Softmax is the final layer in CNN architecture and gives the probability distribution of classes. The class having the highest probability will be selected as the predicted class. The output of the softmax gives us the likelihood of a particular image belonging to a certain class.

IV. Experiments

A. *Datasets and Setups*

In this paper, the dataset implements the German traffic sign recognition benchmark (GTSRB). It was randomly obtained from the camera in a real-time scene [7] and was initially provided by the International Joint Conference on Neural Networks (IJCNN) in 2011. The dataset includes 51839 images in 43 classes, of which training set images have 39,209 samples testing images have 12,630 samples, and each image resolution is dynamically changed from 15 × 15 to 250 × 250.

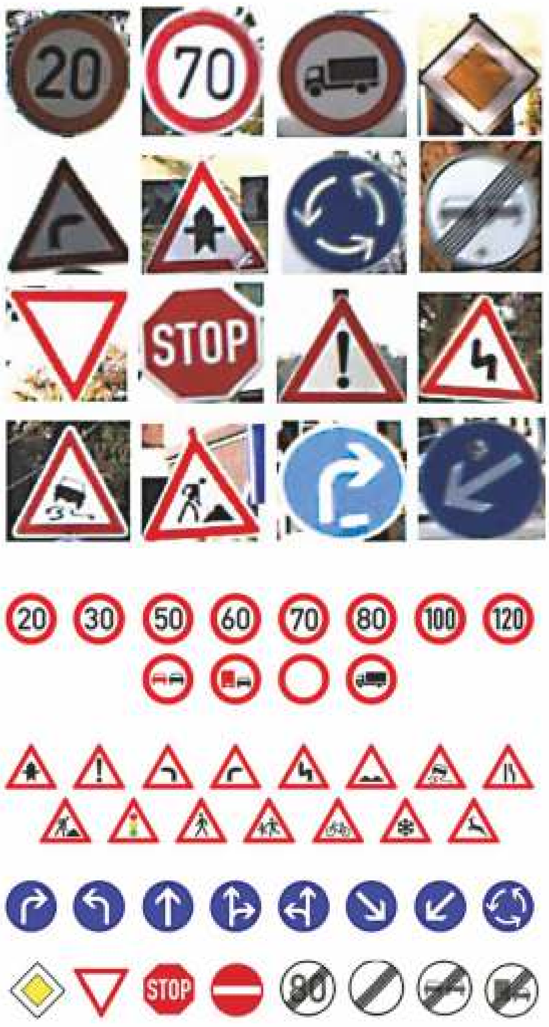


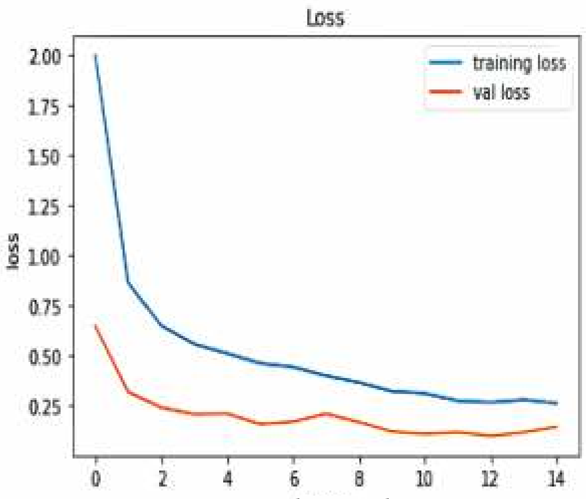
Fig. 5 An example of the image dataset in GTSRB

The image with constant change in scales increases the quality of the data and can improve the fitting effect of the network. Fig. 5 is the type of all datasets, which contains 43 road sign categories. The evaluation of the neural network is performed on the following configuration, and the experiment is performed according to the division of the GTSRB dataset.

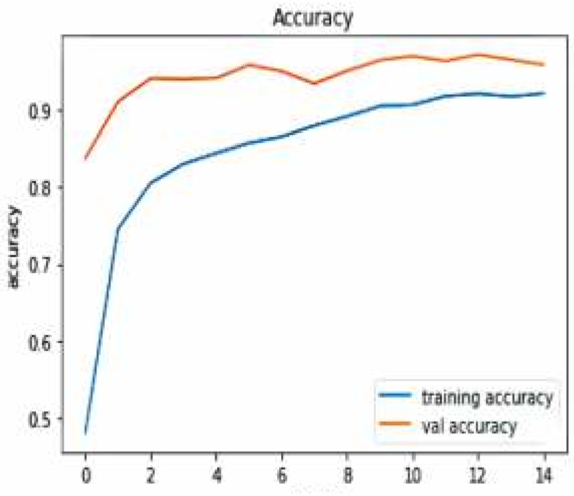
1. CPU: Intel (R) I3-7020U, 2.50GHz
2. Memory: 2GB DDR4
3. Disk capacity: 1T
4. Operating system: WINDOWS 10

B. Evaluation and Analysis

We trained the network and stopped the training when the loss of the training fluctuated stably, so iterated 15 rounds in total to get the trained model. Figure 6 shows the loss and accuracy of images during the training process.



a. Loss value epochs



b. Accuracy value epochs

Fig. 6. Network iteration graph of loss and accuracy

Fig. 6(a) shows that the loss value gradually approaches zero, but the accuracy rate at around 95 in Fig. 6(b). We think this is a normal phenomenon, because the use of data enhancement, the image after each batch of data enhancement is different. In other words, the image data is not fixed every time, so the accuracy may not achieve a high effect, which also prevents the network from overfitting, thereby showing a better effect on the test dataset.

V. Conclusion

In this paper we propose a convolutional neural network suitable for traffic sign recognition classification. The network achieves the recognition of traffic signs through simple convolution and pooling operations and is verified on the GTSRB data. The disadvantage of this network is its processing time, which is not faster than the detection speed of current algorithms, but it has a simple architecture and strong scalability. In future work, we consider recognizing traffic signs under severe weather and conducting experiments on more benchmarks.

References

[1] Bouti, A., Mahraz, M.A., Riffi, J. and Tairi, H., 2019. A robust system for road sign detection and classification using LeNet architecture based on convolutional neural network. Soft Computing, pp.1-13.

[2] Stallkamp, J., Schlipsing, M., Salmen, J. and Igel, C., 2011, July. The German traffic sign recognition benchmark: a multi-class classification competition. In The 2011 international joint conference on neural networks (pp. 1453-1460). IEEE.

[3] Wu, Y., Liu, Y., Li, J., Liu, H. and Hu, X., 2013, August. Traffic sign detection based on convolutional neural networks. In The 2013 international joint conference on neural networks (IJCNN) (pp. 1-7). IEEE.

[4] Z. He, Z. Xiao and Z. Yan, "Traffic Sign Recognition Based on Convolutional Neural Network Model," 2020 Chinese Automation Congress (CAC), Shanghai, China, 2020, pp. 155-158, doi: 10.1109/CAC51589.2020.9327830.

[5] M. T. Islam, "Traffic sign detection and recognition based on convolutional neural networks," 2019 International Conference on Advances in Computing, Communication and Control (ICAC3), Mumbai, India, 2019, pp. 1-6,

doi: 10.1109/ICAC347590.2019.9036784.

[6] S. Xu, D. Niu, B. Tao and G. Li, "Convolutional Neural Network Based Traffic Sign Recognition System," 2018 5th International Conference on Systems and Informatics (ICSAI), Nanjing, China, 2018, pp. 957-961, doi: 10.1109/ICSAI.2018.8599471.

[7] L. Shangzheng, "A Traffic Sign Image Recognition and Classification Approach Based on Convolutional Neural Network," 2019 11th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), Qiqihar, China, 2019, pp. 408-411, doi: 10.1109/ICMTMA.2019.00096.

Anshu Abhipsa Sahoo, Sudhanshu Ranjan Kar, Lipun Patri, Satyajit Swain
7th Sem., Dept. of EE

Home Assistant's Voice Preview Edition is a little box with big privacy powers

Smart home technology, such as voice-activated assistants and automated lighting has been in the market for quite some time and the technology is rapidly expanding. All big players like Samsung, Google, Apple & Amazon offer home assistance apps. Further, many independent players also offer smart home automation. Home automation is all the more crucial since it can be adapted to assist the elderly, enhancing their safety and independence at home. In this regard, a major player in home automation technology i.e. Home Assistant announced today the availability of the Voice Preview Edition, its own design of a living-room-friendly box to offer voice assistance with home automation. Using its privacy-minded Nabu Casa cloud—or your capable computer—to handle the processing, the Voice Preview Edition (VPE) has the rough footprint of a modern Apple TV but is thinner. It works similarly to an Amazon Echo, Google Assistant, or Apple Siri device, but with a more focused goal. Start with a wake word—the default, and most well-trained version, is "Okay, Nabu," but "Hey, Jarvis" and "Hey, Mycroft" are available. Follow that with a command, typically something that targets a smart home device: "Turn on living room lights," "Set the thermostat to 68," and "Activate TV time." The “Thing” is used for primarily controlling devices, scenes, and automation around your home, set up in Home Assistant. That means you have to have assigned them a name or alias that you can remember. Coming up with naming schemes is something you end up doing in big-tech smart home systems, too, but it's a bit more important with the VPE. Home Assistant has good "bridge" options built into it for connecting all the devices. The VPE box can run timers (with neat LED ring progress indicators), and with a little bit of settings tweaking, you can connect it to Home Assistant's built-in shopping lists and task lists or most any other plug-in or extension of the system. It can be messed with LLMs(Large Language Models)—like ChatGPT or Google's Gemini—locally or through cloud subscriptions.

Source: https://arstechnica.com

Minamata Incident

The Minamata Incident marked one of the worst cases of water pollution. In 1932, a factory in Minamata City, Japan, began dumping its industrial effluent, Methylmercury, into the surrounding bay and the sea. Methylmercury is incredibly toxic to humans and animals alike, causing a wide range of neurological disorders.

Its ill effects were not immediately noticeable. However, this all changed, as Methylmercury started to bio-accumulate in shellfish and fish in Minamata Bay. These affected organisms were then caught and consumed by the local population. Soon, the ill effects of Methylmercury were becoming apparent. Initially, animals such as cats and dogs were affected by this. The city's cats would often convulse and make strange noises before dying – hence, the term "dancing cat disease" was coined.

Soon, the same symptoms were observed in people, though the cause was not apparent at the time. Other affected people showed symptoms of acute mercury poisoning such as ataxia, muscle weakness, loss of coordination, damage to speech and hearing, etc. In severe cases, paralysis occurred, which was followed by coma and death. These diseases and deaths continued for almost 36 years before they could be officially acknowledged by the government and the organization.

Since then, the government of Japan has adopted various control measures for water pollution to curb future environmental disasters and should serve as a wake-up call for industries discharging wastes in waterbodies.

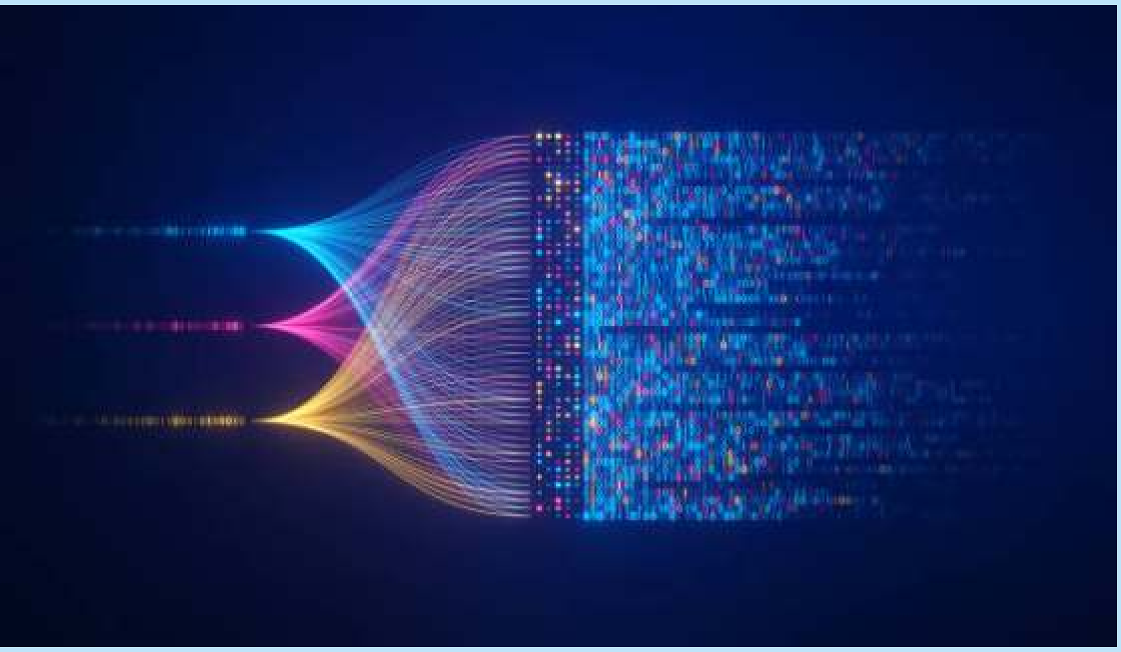
Source: *https://byjus.com/biology/water-pollution-control/*

Publication Cell
Tel: 9937289499/8260333609
Email: publication@silicon.ac.in

www.silicon.ac.in

The Science & Technology Magazine

Digital Digest



Silicon University, Odisha

Contents

Editorial 2
Special Feature 3
Technology Updates 9
DD Feature 11
Profile of a Scientist 17
Historical Tidbits 18
PhD Synopsis 19
Breakthrough in Bio-Informatics 20
Environmental Awareness & Concerns 27

Editorial Team
Dr. Jaideep Talukdar
Dr. Lopamudra Mitra
Dr. Pragyan Paramita Das

Members
Dr. Nalini Singh
Dr. Priyanka Kar
Dr. Amiya Bhusan Sahoo
Mr. Chittaranjan Mohapatra

Student Members
Adarsh Amrit
Abhishikta Sahoo

Media Services
G. Madhusudan

Circulation
Sujit Kumar Jena

Make your submissions to:
publication@silicon.ac.in