International Journal of Computer (IJC)
ISSN 2307-4523 (Print & Online)
https://ijcjournal.org/index.php/InternationalJournalOfComputer/index

Application of Text Summarization on Text-Based Generative Adversarial Networks

Muhammad Alli-Balogun*

University of Salford
Email: muhammadallibalogun@gmail.com

Abstract

In this project, we wish to convert long textual inputs into summarised text chunks and generate images describing the summarized text. This project aims to cultivate a model that can generate true-to-life images from summarized textual input using GAN. GANs aim to estimate and recreate the possible spread of real-world data samples and produce new pictures based on this distribution. This project offers an automated summarised text-to-image synthesis for creating images from written descriptions. The written descriptions serve as the GAN generator's conditional intake. The first step in this synthesis is the use of Natural Language Processing to bring out keywords for summarizing. BART transformers are employed. This is then fed to the GAN network consisting of a generator and discriminator. This project used a pre-trained DALL-E mini model as the GAN architecture.

Keywords: Generative adversarial networks; Text Summarization; BART transformer; DALL-E mini; generator; discriminator.

This work was submitted to the International Journal of Computer on the 17th of January 2024. This work was fully self-funded and had no external financial support of any kind.

1. Introduction

Significant progress has been achieved by Artificial Intelligence in reducing the disparity between human and machine capabilities. Researchers and amateurs alike have exerted effort in several aspects of the discipline to reach astounding achievements. Computer vision is one of the numerous fields in this category.The primary objective of artificial intelligence (machine intelligence and the emulation of intelligent behaviour) is primarily to aid computers to complete intellectual tasks such as managerial, problem-solving, awareness, and accurately interpreting information (in any language and translating them). This is now accomplished by means such as picture and video recognition, image processing and classification, media reproduction, recommender systems, natural language processing, etc.

Received: 11/17/2023
Accepted: 1/17/2024
Published: 1/27/2024

---

* Corresponding author.

However, even with the rapid developments in Artificial Intelligence, the problem of obtaining bigger, high-quality training dataset persists. Acquiring labelled big datasets for training can be problematic, the second limitation is training these datasets, as they require big and expensive computing stations. They also require time, the time commitment required for large datasets can range from weeks to months, as such it makes it difficult for small companies and private individuals to train datasets on their personal hardware. One of the constraints of this paper, comprehending the meaning behind the visual output as the outputs lack interpretability. On the social side, the created output may not necessarily align with societal or public standards, leading to misunderstanding or a distortion of the intended message. These must all be taken into considerations when using AI text-to-image generators.

GANs, or Generative Adversarial Networks, utilise techniques in deep learning, more specifically, convolutional neural networks. GANs consist of generators and discriminators and are taught via adversarial learning. GANs aim to predict the possible spread in real-world data instances and generate unique specimens based on these instances.

Two components comprise a generative adversarial network (GAN):

The *generator* can improve on its generated data over many epochs. The discriminator uses the produced images as negative examples for training. The *discriminator* absorbs how to distinguish between false and genuine data created by the generator. The discriminator penalises the generator if it creates unlikely outcomes.

As training commences, the generator provides seemingly false information, which the discriminator swiftly identifies. As training advances, the generator becomes increasingly accurate at providing an output which will mislead the discriminator. Lastly, if generator training is successful, the discriminator gets less successful at differentiating between real and false. It begins to identify false data as genuine, and its accuracy declines. The optimization of GANs is an issue of minimax optimization. This occurs at a saddle point that creates a minimum with regard to the generator and a maximum with regard to the discriminator, and optimization concludes. That is, the GAN optimization objective is getting to the Nash equilibrium[8]. In that instance, the generator is assumed to hold the distribution of real-world instances properly. GANs have been extensively investigated since their inception because of their immense potential for applications, which include image and vision computing and voice and language processing[18].

1.1 Problem

GANs are a relatively new concept in machine-level space. They were first conceptualized and implemented by Ian Goodfellow and his colleagues in 2014. GAN is still in its infancy stage with many versions of it still under trial and testing. This has inevitably resulted in a lack of performance ratings for different GAN flavours. This has led to one of the biggest hurdles for businesses and researchers face when deciding the ideal GAN framework to use to solve problems.

Lastly, Deep learning is a very CPU and GPU intensive model to run and as such require a lot of computing power to train even the simplest models. Minimum recommended system requirements are Quadcore Intel i7 Skylake

processors with at least 2.5ghz or later, 16gb of RAM or higher (8-12gb may be okay but will take slightly more time), M.2 PCIe or regular PCIe SSD with a minimum of 256GB storage, though 512GB is preferred for use and a premium graphics card like Nvidia-P100 GPU with at least 4gb on it.

1.2 Background

Machine learning tackles the problem of constructing computers that better themselves impulsively through learning. This is an expanding field of computing algorithms intended to simulate human intelligence by gaining information from their surroundings. They are regarded as the workhorses of the new era of so-called big data. In addition to pattern recognition, computer vision, natural language processing, genetic sequence analysis, robotics, compiler optimization, semantic web, computer security, software engineering, finance, and computational biology, machine learning algorithms have been effectively implemented in biomedical and medical procedures.

1.2.1 Deep Learning and Neural Networks

Neural networks, which are also regarded as artificial neural networks or simulated neural networks, are a subfield of machine learning and the cornerstone of deep learning techniques. This term and arrangement are drawn from the human brain, simulating the communication between biological neurons. [4]. In neural networks, the node layers include an input layer, one or more concealed layers, and an output layer. In general, the more hidden layers a neural network has, the more precise it will be. Each node, or artificial neuron, is connected to another and is accompanied by a weight and threshold. These nodes accept input from the nodes of the preceding layer, multiply it by a weight, and then introduce a bias. If the output of a nod goes beyond a threshold value, that node is active, and data is sent to the next network layer.

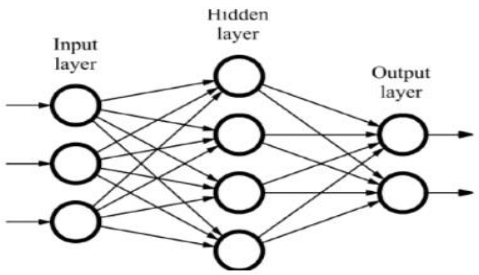

Figure 1: classic neural network architecture, where lines connecting neurons are also shown

The arrows connecting the dots illustrate how all the neurons are linked and how data moves from the input layer through the hidden layers before going to the output layer. Each link is assigned a numerical value known as weight. Each node functions as its own linear regression framework, with input data, weights, a bias (or threshold), and an output.

To calculate the outputs of each neuron in Hidden Layer would look like this

$$Z1 = W1*In1 + W2*In2 + W3*In3 + W4*In4 + W5*In5 + B\_Neuron1 \quad (1)$$

Where W represents weight and in represents the Input value. These weights help to determining the importance of any given variable, with greater weights more significant more to the output than other inputs. All inputs are then multiplied by their corresponding weight before being added together. B_neuron1 is the bias for that particular node, by applying a constant (i.e., the specified bias) to the input, bias enables you to adjust the activation function. Bias in Neural Networks can be compared to the function of a constant in a linear function, in which the line is inverted by the constant value. The output Z1 which is the summation is then processed by an activation function, sigmoid, which provides nonlinearity. Since neural netwoks perform comparably to decision trees by cascading data from node to node, having x values between 0 and 1 reduces the effect of a single variable's change on the output of any given node, and hence the output of the neural network.

The formula could similarly be represented this:

$$\sum wixi + bias = w1x1 + w2x2 + w3x3 + bias \quad (2)$$

Where w1 represents the weight that lives in the link between the input or previous neuron and Neuron 1. Similarly, w2 represents the weight between neuron1 and neuron 2.

$$output = f(x) = 1 \text{ if } \sum w1x1 + b >= 0;\ 0 \text{ if } \sum w1x1 + b < 0 \quad (3)$$

The output is then passed through an activation function and subsequently decides the output. If the output of a node exceeds a predetermined threshold, it "fires" (or activates) the node, delivering data to the subsequent network layer. This causes the output of one node to become the input of the subsequent node. This neural network is a feedforward network since data is moved from one layer to the subsequent layer.

The above equation can be further condensed and generalized as;

$$h_i = \sigma\left(\sum_{j=1}^{N} V_{ij} x_j + T_i^{hid}\right) \quad (4)$$

where σ is called activation (or transfer) function, N the number of input neurons, Vij the weights, $x\,j$ inputs to the input neurons, and T_i^{hid} the threshold terms of the hidden neurons. The mathematical representation of a sigmoid function is shown below.

$$\sigma = \frac{1}{1+ e^{-x}} \quad (5)$$

Deep Neural Network or Deep Net techniques can be applied to neural networks with three or more input and output layers. A neural network with just wo layers is considered rudimentary. Deep nets process data in complex ways by employing sophisticated math modelling. Deep neural networks boost the performance and accuracy of a model. They permit a model to receive a collection of inputs and generate an output.

1.2.2 Convolutional Neural Networks

Convolutional neural network (CNN), a kind of artificial neural networks that has grown dominant in a number

of computer vision and image classification tasks, is gaining interest in other image fields, including radiology. Convolutional Neural Networks (ConvNet/CNN) are a subset of Deep Learning that can take an input image, assign importance (learnable weights and biases), and distinguish between multiple image features/objects. ConvNet requires far less pre-processing than other classification approaches while delivering superior results. This is because, Convnets can capture with the aid of filters, the spatial and temporal dependencies in images. This helps the network understand the sophistication of images better without missing important features that are critical to a good prediction. CNNs are typically made of three layers or three building blocks namely: convolution, pooling, and fully connected layers.

The convolutional layer is the central component of a CNN and the location of the bulk of computation. It requires several elements, including input data, a filter/kernel, and a feature map. This input is an array of numbers called a tensor. The feature map, activation map or convolved picture is gotten after an element-wise multiplication which is essentially the dot product between each element of the kernel and the input tensor is computed at each location of the tensor and added to obtain the output value in the corresponding position of the output tensor. This technique is repeated using several kernels to generate an arbitrary number of feature maps, each of which represents a unique property of the input tensors; various kernels may therefore be regarded as distinct feature extractors.

Pooling layers, sometimes referred to as down sampling, accomplishes dimensionality reduction by decreasing the number of input parameters. In a manner comparable to the convolutional layer, the pooling process sweeps a weightless filter across the whole input. Instead, the kernel applies an aggregation function on the values in the receptive field to populate the output array.

Fully Connected Layer: The term of the fully interconnected layer is descriptive. In partly linked layers, as previously discussed, the pixel values of the input picture are not directly connected to the output layer. In contrast, in the fully connected layer, each node in the output layer is directly linked to a node in the layer behind it.

1.3 Generative Adversarial Networks

GANs, or Generative Adversarial Networks, employ deep learning techniques such as convolutional neural networks. Ian Goodfellow and his colleagues created generative adversarial networks (GANs) in 2014 [6]. Generative Adversarial Networks belong to the set of generative models, meaning that they can produce new content. Two neural networks compete against one another in a zero-sum game, in which one agent's gain matches the other agent's loss. The main objective of GANs is to approximate the likely spread of real data instances and produce novel instances from the original distribution.

Two components make up a generative adversarial network (GAN): Generator and Discriminator. Critically, the generator does not have direct access to actual images; it can only learn through engagement with the discriminator. The discriminator has availability of both synthetic samples and samples obtained from an image stack. The discriminator receives an error signal based on the basic ground truth of whether the picture originated from the actual stack or the generator. Through the discriminator, the same error signal may be utilized to teach

the generator, enabling it to create forgeries of higher quality. Similarly, the discriminator network, D, may be defined as a function that translates picture data to the likelihood that the image is from the real data distribution as opposed to the generator distribution. *(D):(D,x)→ (0,1)*

The optimization of GANs is a minimax optimization problem. At a saddle point that creates a minimum with regard to the generator and a maximum with respect to the discriminator, at this point the optimization is concluded. Thus, the point of GAN optimization is to achieve Nash equilibrium. [8]. At that time, the generator can be judged to have caught the distribution of real-world instances properly. GANs have been extensively investigated since their inception because to their immense potential for applications, like figures and vision computing and voice and speech processing.

1.4 Text Summarization

People are overloaded by the vast quantity of online information and documents due to the Internet's explosive expansion. Summarization is the act of condensing a piece of text into a shorter version, hence reducing the size of the original text while preserving the content's meaning and essential informational elements. Since manual text summarising is often time-consuming and laborious, the automation of the work is gaining popularity and is thus a key push for academic research. Text summarizing has vital applications in a variety of NLP-related activities, including text categorization, question answering, legal text summarization, news summary, and headline generation. In addition, the development of summaries may be incorporated into these systems as an intermediary step that helps minimise the document's length.

As per [15], a summary is "a text that is derived from one or more texts, that conveys crucial information in the original text(s), and that is no longer than half of the original text(s) and, in most cases, substantially less than that." Automatic text summarization is the process of creating a succinct and grammatically correct summary of a text document without the assistance of a person while keeping the original content's meaning. In recent years, multiple systems for the automated summarization of text have been developed and broadly used across a variety of areas. For instance, search engines produce previews of documents comprised of snippets. Other instances are news websites that offer condensed summaries of news subjects, normally in the form of headlines, to aid browsing and knowledge extraction techniques.

There are several methods for extracting information from unprocessed text input and incorporating it into a summarization model, which may be classed as *Extractive* and *Abstractive*.

I. Extractive Text Summarization

This is the conventional approach that was created initially. The primary purpose is to extract the most important sentences from the text and include them into the summary. It is essential to highlight that the derived summary comprises precise quotes from the original text. Most of the current summary research focuses on extractive summarization, as it is simpler and generates naturally grammatical summaries with minimal language examination. In addition, extractive summaries include the most significant sentences from the input, which may be a single text or numerous documents.

II. Abstractive Text Summarization

It is a more sophisticated approach, with regular new developments. The strategy is to find the key pieces, analyze the context, and rewrite them. This guarantees that the essential information is presented in the smallest text feasible. Note that in this case, the summary sentences are produced and not just pulled from the original text. In reality, a good abstractive summary includes the input's key points and is grammatically sound. Abstract techniques capitalise on current advances in deep learning. Given that it may be viewed as a sequence mapping task in which the source text must be transferred to the target summary. These models comprise of an encoder and a decoder, with a neural network that reads the text, encodes it, and then generates the target text.

Even though most summaries written by humans are not extractive, extractive summarization has been the primary area of summarising research in modern times. Pure extractive summaries are frequently superior to abstractive summaries generated automatically. This is because abstractive summarization approaches address more challenging difficulties, such as semantic representation, inference, and natural language creation, than data-driven methods like sentence extraction. There isn’t a truly abstract summarization method available today. Existing abstractive summarizers frequently depend on an extractive pre-processing element to generate the text's abstract.

2. Literature Review

2.1 Generative Adversarial Networks

Game theory's zero-sum game served as the basis for GAN. The definition of a zero-sum game, a non-cooperative contest, is a game where the gains of one party are guaranteed to bring losses to the other side, and the wins and losses of both participants amount to zero. The discriminator evaluates the samples produced by the generator in GAN. More realistic the images produced by the generator, the more challenging it is for the discriminator to determine the authenticity and untruth of the images. Similarly, during the early stages of training, poor-quality samples may be easily detected as bogus samples. They will eventually achieve a point of harmony in the game known as the Nash equilibrium. Nash equilibrium describes the method followed by both players in a game trying to maximise their respective interests. [21] In training program, the generator attempts to create sufficient samples to deceive the discriminator, which attempts to determine the validity of each sample. The process of mutual gaming is a feature and distinguishing property of generative adversarial networks.

Since Goodfellow's introduction of GAN [6], other GAN versions have been developed. Improvements to the model's structure, theoretical expansion, fresh applications, etc., are among the most significant innovative elements. Figure 3 depicts the computing techniques and architectures of many GAN variations.

GANs may be utilised in producing examples with similar statistical spread as actual data, for instance,e to create photorealistic photographs. GANs are also able to address the issue of inadequate training instances for supervised or semi-supervised learning. Additionally, GANs are utilised for voice and lingua processing, like conversation generation. For this chapter, we cover the spectrum of applications for Generative Adversarial Networks.

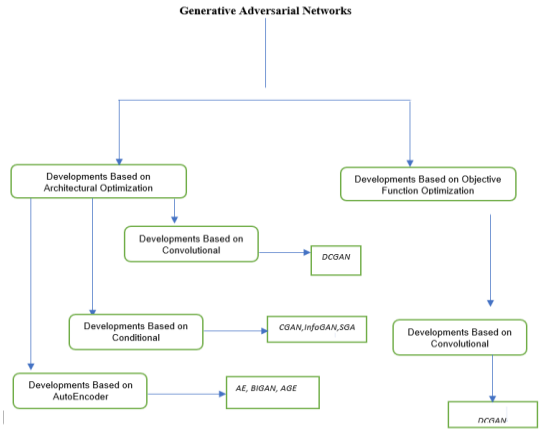

Figure 2: classifications of developments in GAN

2.1.1 Developments Based on Convolutional

The typical neural network consists of just three layers: the input layer, the hidden layer, and the output layer. By activation function, neurons with weights are transferred to the next layer inside each layer. This method of accepting input from the output of the preceding layer is also known as full connection. The problem of this method is that it is impacted by many variables such as weight, bias, input etc. hence, training convergence is too slow, and the generalization effect is poor. DCGAN eliminates the hidden layer of full connection, employing batch normalization during model convergence to prevent generator collapse and deeper gradient propagation. The activation function selects ReLU function in the generator, while Leaky ReLU function is chosen in the discriminator to prevent gradient sparse selection. DCGAN's inception enabled unsupervised feature extraction and image synthesis.

2.1.2 Developments Based on Conditional

Reference [23] proposed CGAN due to the twofold randomness of GAN (random noise and random samples). Adding y-tag as an input, he expanded the (2-D) GAN architecture to the conditional situation by making both the generator and discriminator networks class-conditional. With the addition of y, the input of the generator is transformed into noise and label, whilst the input of the discriminator is transformed into a true sample and creates sample and label. The production of CGAN is governed by a condition, which implements generator supervision and may create distinct samples based on various values. "Due to the discriminator's y input, we may also choose the appropriate type when comparing the sample created by the generator to the actual sample. Not only can CGAN create samples of certain labels, but it can also enhance the quality of samples generated." [24] used conditional GANs to remove rain streaks from images. In the GAN optimization framework, he introduced a new, revised loss function for increased training stability. In addition, a multi-scale discriminator is proposed to use characteristics from multiple scales to assess if a de-rained image is authentic or fake.

2.1.3 Developments Based on Autoencoders

VAE-GANs are considered an expansion of the VAE architecture. It must be emphasized, however, that the

outstanding quality of the created pictures is the result of training as both a VAE decoder and a GAN generator. The authors, [25], suggest substituting the VAE reconstruction error term (expected log likelihood) with the GAN discriminator-expressed reconstruction error. Given that both the decoder of a VAE and the generator of a GAN work on the latent space z to create the image x, a decoder is utilised rather than a generator. This makes the approach more of a combination of VAE and GAN, or one might consider it as an extension of GAN.

Reference [26] suggested a technique for studying the semantics of data distribution and its inverse mapping, employing the learned feature representations to project data back into the latent coordinates. "In addition to the basic GAN framework's generator G, BiGAN also contains an encoder E that transfers input x to latent representations z. The BiGAN discriminator D shows disparity not only in the data space (x vs G(z)), but also in the data and latent spaces (tuples (x;E(x)) versus (G(z); z), where the latent value is either encoder output E(x) or generator input z. With respect to the goal of GANs, the BiGAN encoder E should learn to invert the generator G in this situation."

2.1.4 Developments Based on Objective Function Optimization

Unrolled Generative Adversarial Networks (GANs) are techniques for stabilizing GANs by describing the generator goal in terms of an "unrolled optimization of the discriminator." [27] This method addresses the problem of mode collapse, stabilizes training of GANs with complicated recurrent generators, and boosts the variety and coverage of the generator's data distribution. However, it has a trade-off between more accurate approximation of the true-generator loss and the exponentially increasing computing cost of each training step.

Wasserstein GAN (WGAN) addresses the vanishing gradient issue when training GANs with gradient descent by substituting the Jensen-Shannon divergence with the Earth-Mover distance for measuring the distribution gap between actual instances and artificial ones. They employ a critic function based on the Lipschitz constraint to symbolise the discriminator D.

Reference [28] explored four approaches for estimating the 1-Wasserstein distance between two measures: weight clipping (WC), gradient penalty (GP), c-transform, and (c,)-transform. They found that (c,)-transform and c-transform are more accurate than gradient penalty and weight clipping approaches for computing the minibatch distance and estimating the batch distance.

Energy Based Generative Adversarial Networks (EBGANs) propose linking GANs and auto-encoders and revisiting the GAN framework from the standpoint of alternative energy. The discriminator's energy function can be considered as a trainable cost function for the generator, assigning low energy values to regions with dense data and higher energy values outside of these regions. EBGANs have a superior convergence pattern and scalability for producing high-resolution pictures.

2.1.5 Developments for Computer Applications in Pictures and Vision

GANs are capable of producing picture samples with the same distribution as actual images. [11] defined a deep residual network, SR Res-Net, that establishes a novel state of the art when compared to public benchmark datasets

using the prevalent PSNR metric. However, they identified several drawbacks of this PSNR-focused picture super-resolution and presented SRGAN as an alternative for picture super-resolution. VGG network is used as the discriminator, whereas the residual network is used as the generator. They showed that SRGAN reconstructions for large upscaling factors are four times much more realistic than reconstructions achieved using state-of -the-art reference algorithms.

For training auto-encoder-based GANs, [1] suggest BEGAN, a novel equilibrium enforcing approach combined with a loss calculated from the Wasserstein distance. While conventional GANs attempt to directly equate data distributions, BEGAN outputs similar auto-encoder loss distributions. This technique gives a new approximate convergence measure and balances the generator and discriminator during training. It also gives at least partial answers to various unresolved GAN issues, such as monitoring convergence, regulating distributional diversity, and preserving the equilibrium between the discriminator and the generator. In addition, the authors provide a method for managing the compromise between picture variety and graphic performance. On the picture creation assignment, they set a novel graphic quality benchmark.

Reference [16] presented *comma.ai's* preliminary study on learning a driving simulator shortly after [5] introduced GANs. They sought to construct driving scenarios using GANs. They examined autoencoder and RNN-based video prediction algorithms. In lieu of learning the entire system from ground up, they initially taught the autoencoder with generative adversarial network-based cost functions to produce pictures of the road that appeared realistic. then teaching an RNN transition model in the embedded space followed. The autoencoder and transition model yield realistic results. GANs can create true-to life pictures of the pathway and hence be used in driverless cars for unsupervised or semi-supervised learning, according to the findings.

Simulated + Unsupervised Learning was proposed by [7] to provide realism to the simulator while keeping the annotated synthetic pictures. They discussed Sim-GAN, a method for S+U learning that employs an adversarial network and produced cutting edge outcomes without using any labelled real-world data. Their methodology successfully bridged the gap between artificial and real picture collections.

Image-to-image interpretation is a category of vision techniques whose objective discovering the relationship between an incoming piture and an outgoing picture. By concurrently detecting global structures and local details, [9] proposed "Two-Pathway GAN (TP-GAN) for photorealistic frontal view synthesis with just one face picture". The identity-preserving synthetic picture can be utilized for applications such as facial recognition. TP-training GAN's phase necessitates matched instances of 'identity-preserving frontal view images and face images in a different position'. However, matched training data are unavailable for several jobs. Consequently, [22] introduced CycleGAN for 'learning to translate a picture from a source domain to a target domain without paired instances. Their approach is applicable to a variety of picture-to-picture translation applications, such as style transfer, attribute transmission, and picture augmentation.

In their study, [20] suggested an "Attentional Generative Adversarial Network (AttnGAN) for text-to-image synthesis with fine-grained resolution. First, they construct a unique attentional generating network for the AttnGAN in order to create high-quality images using a multi-step procedure. Then, present a deep attentional

multimodal similarity model to compute the fine-grained image-text matching loss necessary for training the AttnGAN's generator". AttnGAN substantially surpasses prior leading-edge GAN models, increasing the best validation inception result on the CUB dataset by 14.14 percent and on the more difficult COCO dataset by 170.25 percent. Extensive experimental findings confirm the efficacy of the suggested attention mechanism in the AttnGAN, which is particularly important for text-to-image creation for complicated scenarios.Reference [2] presented a unique approach Vector Quantized Generative Adversarial Networks, (VQGAN) for both Generating and modifying pictures from open domain text prompts. Their system can create pictures of good visual fidelity from text cues of large semantic intricacies with no training by employing a multimodal encoder to direct image production. On a range of exercises, we demonstrate that utilizes Contrasting Image-Language Pretraining (CLIP) which main purpose is for judging how well an input matches the text prompt to guide VQGAN generates greater visual quality outputs than previous, less flexible techniques such as DALL-E mini, notwithstanding not having been trained for the responsibilities assigned. They are separate models that work in tandem. This interaction helps the image generator create more precise pictures.

Table 1: Summary of some GAN Architectures

S/No	Paper	Methodology	Performance	Dataset
1	Wasserstein generative adversarial networks	Replacing Jensen-Shannon divergence with the Earth-Mover distance	Improved stability of learning Low quality, converging issues	LSUN-Bedrooms dataset
2	Improved Training of Wasserstein GANs WGAN	Using penalty term in the critic loss instead of clipped weights	Improved stability over clipped weights WGAN, faster convergence	LSUN-Bedrooms dataset
3	Conditional Generative Adversarial Networks CGAN	Addition of y tag as input to th GAN generator and discriminator	Produces samples with specific labels, improvement on the quality of generated samples.	MNIST dataset, ImageNet
4	semi-supervised GAN (SGAN)	Addition of label to discriminator output	Clearer consistent images than traditional GAN	MNIST dataset
5	auxiliary classifier GAN (ACGAN)	Use of labels and noise as inputs to the Generator	Improved quality of generated samples	ImageNet
6	Deep Convolution GAN	Batch normalization used for convergence, uses ReLu activation function and Leaky ReLu is used in the Discriminator	Allows for unsupervised feature extraction	ImageNet

8	Sequence Generated Adversarial Networks (SeqGAN)	Use of sequence generation for training GANs	Strong performance for generating creative audio sequences	Nottingham Dataset
9	Boundary Equilibrium Generative Adversarial Networks BEGAN	Uses auto-encoders as the discriminators	Fast and stable training, quick convergence	360*K* celebrity faces images
10	Information Maximizing Generative Adversarial Networks (InfoGAN)	Suitable for unsupervised learning	Easy to train	MNIST dataset
11	Speech Enhancement GAN	Suitable for audio files	Produces clean enhanced audio	(Valentini-Botinhao and his colleagues., 2016) dataset
12	StackGAN	Uses conditioning augmentation for image synthesis, stage1 GAN draws a rough sketch and Stage2GAN improves on the results of stage 1 GAN	Higher resolution images (256X256)	CUB, Oxford-102, and MS-COCO datasets
12	MedGAN	Utilizes a generative adversarial framework to discover the distribution of actual EHRs.	Impressive results for binary and count variables	HER datasets
13	STGAN	utilizes a discriminator to normalise a generator with a loss function	Good performance for chess games	N/A
14	MalGAN	employs a replacement detector to complement the black-box malware detection method		N/A
15	SRGAN	use VGG network as the discriminator and residual network as the generator	Truer to life photos than state of the art methods	BSD100
16	CycleGAN	Translates an image from source domain to target domain without paired examples unlike TP-GAN	General purpose hence can be used on an extensive variety range of applications	landscape photographs downloaded from Flickr and WikiArt.

2.2 Automatic Text Summarization

In this section, I will cover several extraction and abstraction techniques. Then, compare the findings of each method's benefits and limitations.

2.2.1 Text Summarization using genism with TextRank

Reference [13] developed TextRank, a graph-based sorting engine for textual analysis, based on Google's PageRank [19] algorithm, that identifies the most pertinent phrases within a document. In 1998, PageRank was the original algorithm used by Google to sort online pages. TextRank is an approach for extractive summarization. It is premised on the belief that often-occurring words are noteworthy. Therefore, phrases containing common terms are significant. First, the entire text is divided into sentences, and then the algorithm constructs a network consisting of sentences as nodes and overlapping words as linkages. PageRank finally selects the most significant nodes in this network of phrases. With their research, they demonstrated that TextRank's accuracy in these applications is comparable to that of previously suggested cutting-edge algorithms. TextRank is extremely portable to various domains, genres, or languages since it doesn't involve linguistic expertise or domain- or language-specific marked corpora.

2.2.2 Text Summarization with Seq2Seq

Sequence to Sequence (commonly abbreviated seq2seq) model is a subdivision of Recurrent Neural Network architectures that have generally (but not exclusively) been used to handle complicated Language issues like Machine Translation, Question replying, building Chatbots, Text Summarization, etc. Typically, they enter a sequence from one domain (e.g. text vocabulary) and output a sequence from another domain (i.e. summary vocabulary). Typically, Seq2Seq models exhibit the following characteristics:

- Sequences of the same length as the corpus: the texts are packed into sequences of equal length in order to create a feature matrix.

- Word Embedding mechanism: feature learning approaches in which texts from the lexicon are plotted to vectors of actual values determined in reference to the probability distribution for each word coming before or after another.

- Encoder-Decoder structure: the encoder examines the input sequence and delivers its own internal states as background for the decoder, which estimates the next word of the target sequence based on the preceding words.

- Model for training and Model for predictions: the training model is not directly utilised for prediction. In fact, we will build two neural networks (both having an Encoder-Decoder structure), one for training and the second (named the "Inference Model") to create predictions by reusing a portion of the trained model's layers.

Reference [14] applied the attentional encoder-decoder to the job of abstractive summarization with highly promising results, greatly surpassing state-of-the-art outcomes on two independent datasets. Each of the unique models addresses a distinct challenge in abstractive summarization, resulting in significant performance enhancement. In addition, we offer a novel multi-sentence summarization data set and build benchmarks on it. In future research, they intended to concentrate on this data and develop further robust models for multi-sentence summaries.

2.2.3 Text Summarization using pre-trained with BART transformers.

Recent advancements in model design and model pretraining have contributed to the development of natural language processing. Transformer topologies have simplified the construction of higher-powered models, and pretraining these models have made it feasible to exploit this capacity successfully for an extensive range of activities. Transformers is an open-source collection developed by [17]whose purpose is to make these advancements accessible to the greater machine learning community. Under a common API, the library consists of carefully designed, cutting-edge Transformer designs.

Using Transformers shown that sequence models (such as LSTM) may be entirely substituted for Attention mechanisms, resulting in significantly greater results. These language frameworks may handle any NLP job by simultaneously analysing sequences and mapping relationships between words, irrespective of the distance between them in the text. As a result, in their text embedding, the same word can produce multiple vectors based on its context. Google's BERT [3] and OpenAI's GPT are the most well-known language models, with billions of parameters to train.

Facebook's Bidirectional Auto-Regressive Transformer (BART) developed by [12] employs a regular Seq2Seq bidirectional encoder (identical to BERT) with a left-to-right autoregressive decoder (like GPT). BART = BERT + GPT.

3. Model Architecture

3.1 Model Description

3.1.1 GANs

Generative adversarial neural networks are a robust group of neural networks that employ an efficient method towards unsupervised learning. Through automated study of the core framework and learning of the current structures and features of the real data, GANs can create instances that are highly comparable to the real data distribution.

The GAN architecture adopts a game-theoretic approach naturally. Typically, GANs consist of 2 neural networks that are trained and compete against one another: a generator and a discriminator.

Generator. Model utilised to produce new possible instances from the issue area.

Discriminator. Model utilised for categorising samples as genuine (from the domain) or fictitious (generated).

The name "adversarial" was chosen for GAN because these two networks are in continual confrontation during the training procedure. Adversarial machine learning is a minimax challenge in which a defence searches the parameter space for the parameters that reduce the cost of the classifier. The attacker examines the model inputs at the same time to minimise the cost. These two networks are comparable to a counterfeiter (generator) and law enforcement (discriminator). The generator seeks to develop a type of currency that resembles real-world currency by studying the most recent techniques to mislead the police, i.e., the discriminator. In contrast, the police must continually refresh their knowledge to detect phoney currency. Both networks are continuously upgrading their expertise and receiving evaluations on the success of their modifications. This conflict continues until the authorities are unable to differentiate between actual and false data; this indicates that the pretender is producing a valid specimen.

Here are the steps taken by a GAN:

I. The generator accepts numbers at random and outputs a picture.
II. This produced picture is delivered into the discriminator alongside an image stream extracted from the real, ground-truth dataset.
III. The discriminator accepts the authentic and counterfeit photos and provides probabilities, a value within 0 and 1, with 1 signifying a prediction of authenticity and 0 denoting counterfeit.

So there are two feedback loops:

I. The discriminator is connected to the known ground truth of the pictures through a feedback loop.
II. The generator is coupled to the discriminator through a feedback loop.

CNNs, or Convolutional Neural Networks, are often employed in GANs as the generator and discrimination models. This may be because the initial explanation of the algorithm occurred in the domain of computer vision and utilised CNNs and image data, as well as the substantial improvement made in recent times utilising CNNs more widely to produce novel results on a diverse range of computer missions, such as object detection and face recognition.

3.1.2 Text Summarization

Lewis and his colleagues 2019 introduced the BART model: "Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension." BART is a trans-former encoder-encoder (seq2seq) model that combines a bidirectional (BERT-like) encoder with an autoregressive (GPT-like) decoder. It is a sequence-to-sequence model that employs a bidirectional encoder on distorted textual data and a left-to-right autoregressive decoder. BART is a noise removal autoencoder constructed with a sequence-to-sequence model that is suitable to a vast array of final tasks. Pretraining consists of two phases. First, (1) text is corrupted using random noising, and then secondly, a sequence-to-sequence model is learnt to recover the original text. The pre-training task involves randomly altering the order of the original phrases and using a novel in-filling technique

that substitutes text spans with an unique mask token. It utilises a traditional Transformer-based neural machine architecture design which, despite being straightforward, generalises BERT (thanks to the bidirectional encoder), GPT (with the left-to-right decoder), and many other more current pretraining styles.

3.2 Model Analysis

3.2.1 Generative Advesarial Networks Analysis

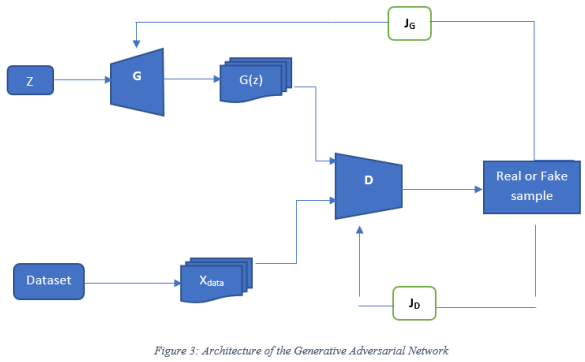

Figure 3: Architecture of the Generative Adversarial Network

Figure 3: Architecture of the Generative Adversarial Network

The GAN architecture is depicted in Figure above. X_{data} and G(z) are the true samples from the training dataset and the manufactured samples generated by generator G, correspondingly. Discriminator D determines the chance that the input image is actual or faked. In GAN, the generator accepts as input a fixed-length noise vector z (the uniformly distributed random vector). The generator then generates fresh data G(z) based on conventional signal distributions X_{data}. To acquire a better grasp of the situation, producing an image may need a vector of random values rather than a picture as input. The points in this multidimensional vector are paired with the points in the issue domain following training, producing a condensed presentation of the data spread.

The discriminator functions as a binary classifier and distinguishes between fraudulent G(z) samples and authentic Xdata samples. The discriminator is trained to optimize the chance of correctly labelling authentic and fabricated data. Invariably, if the input consists of authentic Xdata data, the discriminator labels it as authentic data and outputs a number near to 1. Else, if the feed consists of data created by the generator, the discriminator labels it as bogus data and delivers a number near zero. The generator and discriminator may consist of neural networks, convolutional NNs, recurrent NNs, or autoencoders. In order to update the networks, the discriminator needs the loss function J_D as well as the generator which requires the loss function J_G. Only via backpropagation signals of the faux output does the generator change its settings. In comparison, the discriminator receives more data and adjusts its weights using both simulated and actual output.

$$If\ X = X_{data} \rightarrow D(X) \rightarrow 1 \Longrightarrow \quad max\ D\ V(D.G) = E_{x \approx Pdata(x)}\ [log(D(x))] \tag{6}$$

$$If\ X = G(Z) \rightarrow \{\ D(X) \rightarrow 0;\ for\ D \Longrightarrow \quad maxD\ V(D.G) = E_{z \approx Pz}[log(1\text{-}D(G(z)))] \tag{7}$$

$$If\ X = G(Z) \rightarrow \{ D(X) \rightarrow 1; for\ G \Longrightarrow \quad minG\ V(D.G) = E_{z \approx Pdata}[log(D(x))] \qquad (8)$$

p_{data} represents the spread of real data while p_z stands for random noise spread. D(x) is the possiblity that true data point x is real according to the discriminator's approximation of. E_x is known as the intended value across all real data samples. G(z) is the generator's result when fed noise z. D(G(z)) represents the discriminator's approximation that the likelihood that an artificial sample is real.E_z is the estimated figure across all random inputs to the generator (essentially, the expected value over all generated false cases G(z)). Since the generator cannot directly influence the log(D(x)) component in the function, reducing the loss for the generator is equal to minimising log(1 - D(G(z))

The two parts of equation 2 can be summed up to be.

$$If\ X = G(Z) \rightarrow \ minG\ maxD\ V(D.G) = E_{z \approx Pz}[log(1\text{-}D(G(z)))] + E_{z \approx Pdata}[log(D(x))] \qquad (9)$$

Equation 9 shows the GAN optimization approach to the minimax issue.

As per Equation 6, if X=Xdata, the discriminator should output a value close to 1 if X=Xdata. Thus, X is was anticipated to convey actual data and maximize V (G.D). According to Equation 7, if X=G(Z), it will result in two unique discriminator orientations that satisfy the problem's first criteria. The discriminator is intended to be able to determine that the produced specimen is bogus and to output a result closer to zero. V(G.D) must also be maximized in these conditions. It illustrates the succeeding requirement of the problem from the standpoint of a generator. In this instance, it is optimal for the generator can deceive the discriminator, i.e., the output should be near to 1. Put differently, the generator is taught to deceive the discriminator by reducing V(G.D) and achieving a true data spread. From a mathematical standpoint, Eqn 9 depicts a two-player minimax contest with value function V(G.D).

3.2.2 BART Text Summarization Analysis

BART employs the typical sequence-to-sequence Transformer design in [17], with the exception that ReLU activation functions are changed to GeLUs and parameters are initialised from N. (0, 0.02). Every layer in the decoder further conducts cross-attention across the last hidden layer of the encoder and BERT employs an extra feed-forward network prior to word prediction, but BART does not. BART has around 10% more variables than the similarly sized BERT model.

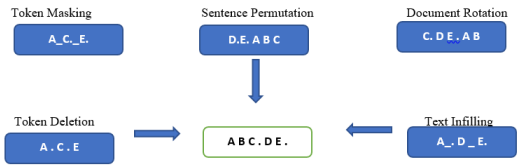

Figure 4: BART Model

Transformations for noising the input that we experiment with. These transformations can be composed

Token Masking; After BERT is applied , random tokens are sampled and [MASK] elements are substituted for them.

Token Deletion; Random tokens from the input are removed. Unlike token masking, the model must determine which input places are absent.

Text Infilling; A group of text spans are sampled using a Poisson distribution (lambda = 3) to determine span lengths. Replace each span with a single [MASK] token. Text infilling teaches the model to forecast the number of tokens missing from a span.

Sentence Permuation; A document is broken into sentences based on periods, and the arrangement of these phrases is randomised.

Document Rotation; A uniformly random token is selected, and the document is flipped such that it begins with that token. This exercise teaches the model to recognise the document's beginning.

4. Experimental Results

4.1 Methodology

As stated earlier in this paper, we intend to evaluate the performance of images produced by our Generative Adversarial Network architecture when it is paired with a text summarization algorithm. How well would the network interpret large chunks of texts compared to abstractly summarized texts compressed by recurrent neural networks before being fed into the GAN structure.

In this chapter, the design choices of the datasets, algorithm types and chosen text environments will be discussed briefly. In the latter parts of this chapter, the results and analysis of the results will be examined.Since traditional GAN structures require lots of computing resources and time to train and fine-tune models. I decided to use pretrained models by DALL E Mini based on Hugging Face[1]. This mode is a transformer-based text-to-image generation model, which was 'Developed by Boris Dayma, Suraj Patil, Pedro Cuenca, Khalid Saifullah, Tanishq Abraham, Phúc Lê, Luke, Luke Melas, Ritobrata Ghosh.' It's a model for generating visuals from textual cues. The model creators said in the DALLE mini project report, "OpenAI has the first excellent model for picture generation using DALLE. DALLE mini is an effort to reproduce these outcomes using an open-source methodology." The model is based off the VQGAN which uses a BART[2] as encoder. DALL E was developed using three datasets:

- The Conceptual Captions Dataset[3] comprises three million pairings of images and captions.
- Conceptual 12M[4] comprises 12 million pairings of images and captions.
- The OpenAI subset[5] of YFCC100M[6], which contains around 15 million photos and was reduced to 2 million images owing to storage constraints. They employed both the title and the description as the caption and eliminated HTML elements, new lines, and additional spaces.

In their[7] technical report on DALLE Mini, the model's researchers give comparisons between DALLE Mini's findings and those of DALLE-pytorch[8], OpenAI's DALLE, and models consisting of a generator paired with the CLIP neural network model[9].

For the text summarizer, a similar route was employed. This time, pre-trained BART transformers fine-tuned on the CNN daily Mail[10] developed by [12]. BART is a sequence-to-sequence (seq2seq) transformer encoder-encoder model featuring a bidirectional (BERT-like) encoder and an auto - regressive (GPT-like) decoder. BART is pre-trained by (1) distorting text with random noise and (2) developing a model to recover the original text.

BART is more successful when fine-tuned for text production (e.g., summary, translation), but it is equally useful for comprehension tasks (e.g. text classification, question answering). CNN Daily Mail, a big collection of text-summary pairings, has optimised this specific checkpoint.

For ease of computing power and other resources, Google Colab Pro was chosen as the test environment. The Colab Pro environment uses Nvidia GPU P100 or T4, has up to 32gb allowance and limits session up to 24 hours.

For ease of viewing and brevity, snippets of the code used can be found in the appendix section. However, for the full copy of the code can be found on the github[11] profile.

4.2 Results

To assess the visual quality of photos created, we compare subjectively the DALLE mini with a BART transformer and one without a text summarizer. We would refer to the model with the text summarizer as model A and the one without the text summarizer as model B. For Model B, the texts are first passed through the BART text summarizer to condense the words to a smaller piece while retaining the key aspects of the texts necessary to produce a near accurate image. This is then passed through the DALL E mini GAN structure which outputs a pictorial image.When passing text of word lengths of less than 200 across the two groups, images produced were similar. Both models produced samples that were consistent with the descriptions of the input text. For example, when the text of an African wild dog[12] from Britannica's page which contains 177 words.

Figure 5: showing Model A's depiction of the African wild dog

Below is the sample produced by model B

Figure 6: Model B's depiction of the African wild dog

For the above texts Model A consistently produced samples in less time than model B. Model A averaged 190s to run while model B averaged 193s

As the size of the text increases, we see a more interesting variance in the output of the pictures produced. When the text about the savannah elephant from worldwidelife[13] page was used model B produced more samples with more elephants in the picture than model A. this could be as a result of the extra features the text summarization might have omitted. As usual, model A completed the run-in lesser time with 211s while model B finished in about 218s.

Figure 7: model A's output when fed with the elephant text

Figure 8: model B output when fed with text about elephants

However, for texts with more than 600-word count, it was observed that while the model B without the text summarizer was able to keep up somewhat with the sister model A with the text summarizer, it needed more time to process the large number of words and glean as much information. This was especially evident in the Amazon text[14] which has over 1600 words, this consequently took the model B *221* seconds to process the images. For large texts, the texts size was broken up into chunks of about 200 words to help the GAN model process samples more specifically to those chunks. This was done to produce less generalized samples.

The images produced showed some consistencies with words used in the text although some images were very similar with very little differences as shown in the images below

Figure 9: model A's sample of the Amazon forest

Figure 10: model B's sample output from the amazon text

Figure 11: Examples of images with identical samples produced

The above example highlights that both models were able to still produce very similar samples with those described in the text.

4.3 Analysis

The results from the experiment above show that model A that has the text summarizer couple to the GAN architecture shows better performance as the length of the text increases. This is most evident in the run times of the models.

For much shorter texts, text below the two hundred word mark, both models perform similarly and are identical in performance. They both averagely take about 190s to complete the run and produce an output. As the word count of the texts increase we start to see more details. Model B tended to show more diverse samples since it had more descriptions to work with.. Although this also means an increase in runtimes of about 5 seconds extra.Laslty for texts with lenghts of between above the 600 mark, we chose to split the text in model A into shorter chunks of 200 words since the average number of words in a paragraph is 200. This will enable model A to produce images for specific paragraphs leading to less general samples. This could also help reduce the amount of information it had to process, leading to shorter run times of about 7 seconds. B tended to show more diverse samples since it had more descriptions to work with. Overall images from both models were not so dissimilar as expected.

Table 2: shows the time taken to run each sample text on both models

Num of words	Method	Time to run(seconds)
<200	Text Summarizer + GAN	190 ± 3
	GAN	193 ±3
400-500	Text Summarizer + GAN	211± 3
	GAN	218 ±3
>600	Text Summarizer + GAN	215 ±3
	GAN	221 ±3

Although the majority of research on Generative Adversarial Networks has focused on training, fine-tuning, and general improvements to the model, practically no attention has been paid to how well the GAN performs when it is given vast volumes of text to process. This report focuses on the lenght of the runtimes and the quality of samples produced when large chunks of text are fed into the GAN architecture. As a consequence of this, it will be challenging to evaluate our model in relation to others. As a result, there will be a need for more study in order to give additional comparative analyses.

5. Conclusion

5.1 Summary

In this report , we incorporated a BART text summarizer to a pre-trained Generative Adversarial Network Architecture. The suggested technique reduces the size of the input text using abstractive methods before the summarised text is fed into the DALL E mini GAN framework. The degree to which the summarization procedure is carried out can be altered, and this change will take effect before the code is executed. Extensive quantitative and qualitative results illustrate the effectiveness of this method for very large size of texts especially if time is limited or if computing power is low.

5.2 Discussion

Text summarization has become an increasingly demanding feature in the modern world with large volumes of text becoming available. This method provides an insight as to how large volumes of text from articles, webpages, novels, and other materials can be quickly turned into descriptive images which encapsulate the main idea in the story. This is mainly possible by splitting larger chunks of text into smaller and summarized bits that are easier for the GAN architecture to digest. In future, this idea can be expanded into producing moving pictures in the form of video form simple text like novels and documentaries.

5.3 Future Work

As mentioned earlier, Generative Adversarial Networks are a relatively new field, and with the increasing power of computers, training of models is much faster. While this provides a basic framework as to how large chunks of

text can be illustrated, more work is needed in this field to develop interesting topics like.

- Text-to-video synthesis
- Movie making
- Audio-to-video synthesis

References

[1] D. Berthelot, T. Schumm, and L. Metz, "Began: Boundary equilibrium generative adversarial networks," 2017, pp. 8.

[2] K. Crowson, S. Biderman, D. Kornis, D. Stander, E. Hallahan, L. Castricato, and E. Raff, "Vqgan-clip: Open domain image generation and editing with natural language guidance," 2022, pp. 16.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018.

[4] I. C. Education, "What are Neural Networks? " Internet: https://www.ibm.com/cloud/learn/neural-networks, 2020 [July 2022].

[5] I. Goodfellow, " Generative adversarial networks," presented at NIPS on GANs, 2016.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in Neural Information Processing Systems (NIPS)*, Springer New York, 2014.

[7] C. Gou, Y. Wu, K. Wang, F.-Y. Wang, and Q. Ji, "Learning-by-synthesis for accurate eye detection," *23rd International Conference on Pattern Recognition (ICPR)*. 2016.

[8] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," *IEEE Transactions on Knowledge and Data Engineering* pp.3313-3332, vol. 35 2020.

[9] R. Huang, S. Zhang, T. Li, and R. He, "Beyond face rotation: Global and local perception GAN for photorealistic and identity preserving frontal view synthesis," *Proc. of the IEEE International Conference on Computer Vision (ICCV).* 2017

[10] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *International Conference on Machine Learning*, 2015.

[11] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, and Z. Wang, "Photo-realistic single image super-resolution using a generative adversarial network," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[12] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," 2019.

[13] R. Mihalcea and P. Tarau, "Textrank: Bringing order into text," *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004.

[14] R. Nallapati, B. Zhou, C. Gulcehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence RNNs and beyond," 2016.

[15] D. Radev, E. Hovy, and K. McKeown, "Introduction to the special issue on summarization,"

Computational Linguistics, vol. 28, no. 4, pp. 399-408, 2002.

[16] E. Santana and G. Hotz, "Learning a driving simulator," 2016.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[18] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, and F.-Y. Wang, "Generative adversarial networks: introduction and outlook," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 4, pp. 588-598, 2017.

[19] Wikipedia, "PageRank," PageRank. [Online]. Internet: https://en.wikipedia.org/wiki/PageRank. 16 August 2014 [2022]

[20] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "Attngan: Fine-grained text to image generation with attentional generative adversarial networks," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[21] J. Zheng, Y. Cai, Y. Wu, and X. Shen, "Dynamic computation offloading for mobile cloud computing: A stochastic game-theoretic approach," *IEEE Transactions on Mobile Computing*, vol. 18, no. 4, pp. 771-786, 2018.

[22] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *Proc. IEEE International Conference on Computer Vision*, 2017.

[23] Mirza, M., & Osindero, S. "Conditional generative adversarial nets", 2014.

[24] Zhang, H., Sindagi, V., & Patel, V. M. "Image de-raining using a conditional generative adversarial network". *IEEE transactions on circuits and systems for video technology, 2019.*

[25] Larsen, A. B. L., Sønderby, S. K., Larochelle, H., & Winther, O. "Autoencoding beyond pixels using a learned similarity metric". *International conference on machine learning, 2016.*

[26] Donahue, J., Krähenbühl, P., & Darrell, T. "Adversarial feature learning." 2016.

[27] Metz, L., Poole, B., Pfau, D., & Sohl-Dickstein, J. "Unrolled generative adversarial networks", 2016.

[28] Mallasto, A., Montúfar, G., & Gerolin, A. "How well do WGANs estimate the Wasserstein", 2019.

Footnotes

[1] Pretrained models based on Hugging Face : https://huggingface.co/dalle-mini/dalle-mini
[2] Facebook BART: https://arxiv.org/abs/1910.13461
[3] Conceptual Captions Dataset: https://aclanthology.org/P18-1238/
[4] Conceptual 12M Dataset: https://arxiv.org/abs/2102.08981
[5] OpenAI Dataset : https://github.com/openai/CLIP/blob/main/data/yfcc100m.md
[6] YFCC100M : https://multimediacommons.wordpress.com/yfcc100m-core-dataset/
[7] WandB DALLE mini report: https://wandb.ai/dalle-mini/dalle-mini/reports/DALL-E-Mini-Explained-with-Demo--Vmlldzo4NjIxODA#the-results-of-our-dall-e-experiment
[8] DALL E pytorch Github repo: https://github.com/lucidrains/DALLE-pytorch
[9] CLIP neural network model: https://openai.com/blog/clip/
[10] CNN Daily Mail: https://huggingface.co/datasets/cnn_dailymail
[11] GAN code on github profile: https://github.com/Embashosho/Embashosho
[12] African Wild Dog text: https://www.britannica.com/animal/African-hunting-dog
[13] WorldwideLife Elephant Text: https://www.worldwildlife.org/species/elephant#:~:text=Elephants%20are%20the%20largest%20land,or%20bathing%2C%20among%20other%20uses.
[14] Article on Amazon Rain Forest: https://studycorgi.com/the-amazon-rainforest-an-integral-component-of-the-environment/