Implementation of Image Generative models

using imageGPT

Payel DuttaKaustuv Kunal *Student – DSML Principal Data Scientist  
 PES University Great Learning* Bangalore, India Bangalore, India  
 payel1191@gmail.com kaustuv.kunal@greatlearning.in

*Abstract*—***Motivated by advancements in unsupervised representation learning for natural language, this paper sought to investigate whether comparable models could also generate useful representations for images. Using a sequence Transformer, I trained the model to autonomously predict pixels, without any awareness of the 2D structure of the input. Despite being trained on low-resolution ImageNet without labels, the imageGPT model achieved impressive results, displaying robust image representations, as evidenced by linear probing, fine-tuning, and classification with low-data. To achieve the desired outcome, I implemented a two-stage approach, involving pre-training and fine-tuning. During the pre-training stage, I experimented with both auto-regressive and GPT objectives. Based on findings, it appears that generative image modeling remains a promising avenue for acquiring high-quality, unsupervised image representations. This study will come in useful whenever there is a need for image completion. Given any random half image, the model will try to predict completed version of the image in multiple forms, which can be extended later to facial recognition, object identification as well as it might find its use in healthcare domain.***

Keywords— image completion, image prediction, GPT-2, iGPT, MNIST, NLP

# Introduction

In today’s world, human beings are very fond of clicking pictures, otherwise known as images. This is specially attributed to the improvement and affordability of technology that every family owns a smartphone or a camera. People want to capture every moment and almost 1/10th of the world’s population is addicted to taking selfies. This results in a huge amount of data when it comes to preserving the images.

The rate of crime is also increasing nowadays, and given a good resolution of image, it is easier for the cops to nab the criminal. But, oftentimes, the images might be half visible, might be partially dark, might not be clear etc. In those case, facial recognition can prove to be a daunting task. This paper aims to explore the areas wherein ImageGPT (iGPT) can help in image completion, provided it has already been trained by a large number of image datasets[[1](#One)].

These days people are very social and often gather in large happening events where chances of losing individual items are very high. Those items might be very expensive or might be of value to them and they will try to put in all efforts to get it back, but often times, the picture of the exact lost item might not be available, but they might get an object which is quite similar to it. In those cases, given the newer object as an input, the model will try to predict similar images to the lost item, that will give the searchers an idea of what the item looked like. This method could also be useful for identifying missing persons.

Nowadays, due to the immense volume of images, it is difficult to effectively store all the data, and it is very likely that some images might get deleted or the quality might be compromised. It will be very difficult for the professionals, to correctly identify person or things due to lack of relevant data. ImageGPT, launched in the year 2020, was built on top of GPT-2 and has shown good performance in both image classification as well as image completion. In this paper, I wanted to explore the areas of image completion, where a model will be able to predict complete images when provided with half-cut images, or low-quality images. The pretrained model will predict multiple similar images, and then the professional will just have to pick the image that best matches the description. This will save lot of time in manually imagining/painting pictures of criminals, objects, or places as it will be readily available to them.

As generative pre-training methods for images celebrate their tenth anniversary and continue to exert a substantial impact in NLP, it is time to revisit this category of methods and compare them with recent progress in self-supervised methods. This study aims to re-evaluate generative pre-training on images and assess its competitiveness relative to other self-supervised approaches. It is demonstrated that with a flexible architecture, an efficient likelihood-based training objective, and considerable computational resources (2048 TPU cores), generative pre-training can hold its own against other self-supervised methods. The findings indicate that generative pre-training is capable of learning representations that significantly enhance the state of the art in low-resolution unsupervised representation learning scenarios.

iGPT is a generative model that uses a transformer architecture for image completion tasks. The model works by predicting the missing pixels in an image based on the surrounding pixels. Input image is preprocessed by normalizing pixel values and resizing it to a fixed size. The preprocessed image is then fed into the ImageGPT model, which uses a series of transformer layers to encode the input image into a high-dimensional vector representation. A portion of the encoded image is then masked out, representing the missing pixels that need to be completed. The model then generates the missing pixels by using an autoregressive generation process[[2](#Two)]. Starting from the leftmost missing pixel, the model predicts the value of the pixel based on the surrounding pixels that have already been generated. The process is then repeated until all the missing pixels have been generated. The completed image is then obtained by adding the generated pixels back into the input image.

In this paper, I tried to leverage ImageGPT for image completion. The existing papers implemented iGPT with TensorFlow. Since TensorFlow is not supported by browsers, so I implemented the model with Pytorch and to get better accuracy of the model, I augmented the training data, to generate more samples. A common approach to evaluate the quality of representations is to fine-tune them for image classification. Pre-training can then be seen as a valuable initialization or regularizer when combined with early stopping. The dataset used in this study is MNIST dataset, but this can also be extended to other datasets in the future.

***Literature Review***

The history of ImageGPT (iGPT) can be traced back to the development of the GPT (Generative Pre-trained Transformer) model by OpenAI. The GPT model was designed to generate coherent and natural language text by training a large-scale neural network on a massive corpus of text data. The success of GPT inspired the development of the iGPT model, which was trained to generate images from textual descriptions using a similar approach. The iGPT model was introduced in 2021[[3](#Three)], and it builds on previous work on generative models for images such as the Generative Adversarial Network (GAN) and Variational Autoencoder (VAE).

iGPT is based on the transformer architecture[[4](#Five)], which is a type of neural network that has proven to be highly effective in natural language processing. The model is trained on large datasets of image-caption pairs, allowing it to learn the relationship between textual descriptions and the corresponding visual content. The iGPT model has shown impressive results in generating high-quality images that match textual descriptions, and it has the potential to be used in a wide range of applications, including design, art, and gaming. As with other deep learning models, ongoing research and development will likely continue to improve and expand the capabilities of iGPT in the future.

# Materials and Methods

## About the Dataset

The MNIST database is a widely used collection of handwritten digits that has become a benchmark for evaluating image processing systems. Created by remixing samples from NIST's original datasets, the MNIST dataset contains 60K training and 10K testing images of handwritten digits, each of which has been normalized to fit within a 28x28 pixel bounding box and anti-aliased to produce grayscale levels. The creators of the dataset felt that the original training and testing datasets, which were sourced from American Census Bureau employees and American high school students respectively, were not ideal for machine learning experiments, hence the remixing and normalization. Half of the training and test sets are derived from NIST's original datasets, while the other half comes from NIST's testing dataset. The MNIST dataset has been used to evaluate a variety of machine learning methods, and the original creators keep a record of some of the most effective approaches that have been tested on the dataset.

Bottom of Form

## Model Architecture

The input image is first preprocessed by resizing it to a fixed size and normalizing the pixel values. The preprocessed image is then fed into the ImageGPT model, which uses a series of transformer layers to encode the input image into a high-dimensional vector representation[[5](#Four)]. A portion of the encoded image is then masked out, representing the missing pixels that need to be completed. The model then generates the missing pixels by using an autoregressive generation process. Starting from the leftmost missing pixel, the model predicts the value of the pixel based on the surrounding pixels that have already been generated. The process is then repeated until all the missing pixels have been generated. The completed image is then obtained by adding the generated pixels back into the input image. The key to the success of ImageGPT is the use of transformer layers. These layers allow the model to capture long-range dependencies in the input image, which is essential for generating high-quality completed images. Additionally, the autoregressive generation process ensures that the completed image is visually consistent with the input image.

## Training strategy used for the model:

The MNIST dataset is downloaded from OpenAI blob store. The data is sent for augmentation (Random Crop, Horizontal Flip, etc). The images are resized to 28x28 pixels. The centroid is calculated using MiniBatchKMeans. The centroid is saved in Numpy File. The centroid and config file to ImageGPT (GPT-2) model. Using DataLoader, create train, validation, test data set. The train, validation, and test data is set for training (Pytorch lightning library is used to speed up training process). Once training completes, send random cropped image from MNIST dataset. Model should be able to predict full context images.

## Evaluation based on metrics

### **Confusion Matrix :**

### A confusion matrix is a visual representation of the different outcomes of a classification problem, showing the predicted and actual results in a table format. This tool is particularly useful for understanding the nuances of a model's performance, especially in cases where class imbalances may be present.

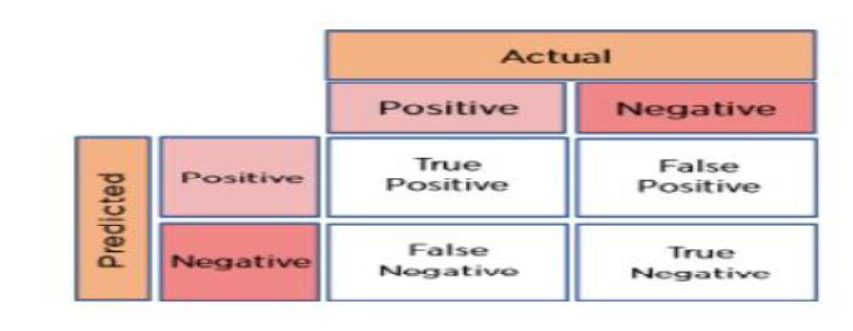


Figure 1: Confusion Matrix

**1. True Positive (TP)**: It refers to the number of instances in which the model correctly predicts a positive outcome when the actual value is also positive.

**2. False Positive (FP)**: It refers to counting the instances in which the model predicted a positive outcome, but the actual value was negative.

**3.True Negative (TN)**: It measures the number of instances in which the model correctly predicted a negative outcome when the actual value was also negative.

**4. False Negative (FN)**: It counts the instances in which the model predicted a negative outcome, but the actual value was positive.

It can be challenging to determine the performance of our model based solely on the confusion matrix. To gain a clearer understanding of our model's accuracy, we rely on a set of metrics, including:

**Accuracy:** It is a commonly used metric in evaluating the performance of a classification model. It measures the proportion of correctly classified instances, which indicates how often the model makes the correct prediction.

**Precision**: Precision is a commonly used evaluation metric in classification tasks that measures the model's ability to correctly identify positive instances out of the total predicted positive instances.

**Recall:** It is used to calculate the model's ability to predict positive values.

**F1-Score:** It is a widely used evaluation metric in classification tasks that combines both precision and recall into a single score.

### **Kappa Score :**

The Kappa score, also known as the Cohen's Kappa coefficient, is a statistical measure used to evaluate the inter-rater agreement between two raters. It is particularly useful when evaluating the performance of classification or prediction models where two or more raters are involved. The Kappa score ranges from -1 to 1, where a score of -1 indicates perfect disagreement between the raters, a score of 0 indicates that the agreement is no better than chance, and a score of 1 indicates perfect agreement between the raters. The Kappa score is calculated by comparing the observed agreement between the raters with the expected agreement that would occur by chance. It is computed using the formula:

where p0 is the observed proportion of agreement between the raters and pe is the proportion of agreement that would be expected by chance.

A Kappa score of 0.8 or higher is generally considered to indicate excellent agreement between the raters, while a score between 0.6 and 0.8 indicates good agreement, and a score below 0.6 indicates poor agreement. The Kappa score is commonly used in fields such as psychology, medicine, and social sciences to evaluate the inter-rater reliability of diagnostic tests or assessments[[6](#Six)]. It is also used in machine learning and natural language processing to evaluate the performance of classification or prediction models.

## Methodology:

The MNIST dataset is a widely used benchmark dataset in machine learning and computer vision. It consists of a set of 70,000 handwritten digits that have been normalized and centered, each digit being a grayscale image of size 28x28 pixels. The training set is used to train machine learning models, while the test set is used to evaluate their performance. For this project we used torchvision.

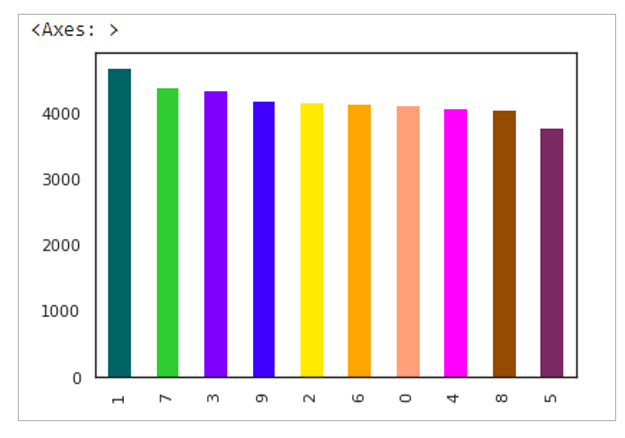
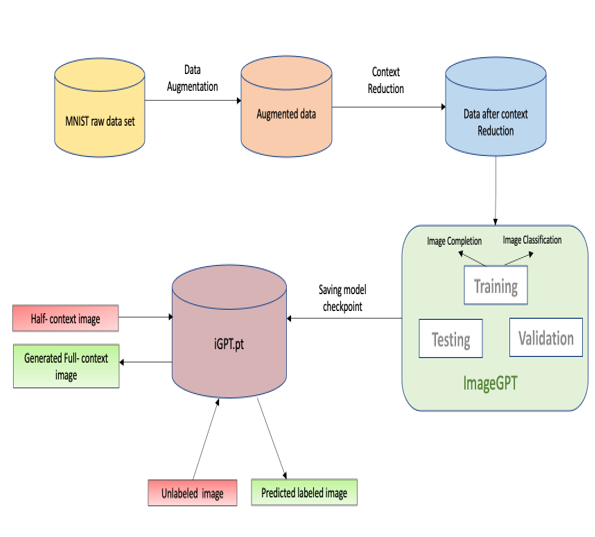


Figure 2: Count of each label of MNIST dataset for training

Torchvision is a PyTorch library that provides 3 kinds of modules - a collection of datasets, transforms, and models, specifically designed for computer vision tasks. It is used to load and preprocess image data, and to define and train deep neural networks for image classification, segmentation, and detection[[7](#Seven)]. The “torchvision.datasets” module provides a set of standard datasets, including MNIST, CIFAR-10, CIFAR-100, and ImageNet. I downloaded the MNIST dataset and loaded into PyTorch. 60000 training images were not enough to train the imageGPT model. To overcome that issue , image augmentation had been performed on training images.

The “torchvision.transforms” module provides a set of standard image transformations that can be applied to images, such as normalization, resizing, cropping, and flipping. These transformations are often used for data augmentation to improve the performance of deep neural networks. The transformed module from torchvision was used for image augmentation.

Context reduction was performed as the resolution of the image was high. After that image were sent for training, testing and validation to imageGPT model. Once the training completed the model was saved into disk in .pt file for future downstream task. Now if any half context input image is passed to the model it will generate the full image.

## Implementation Details:

Pytorch has diverse set of libraries to support machine learning and deep learning algorithm, torchvision library is one of them. Torchvision library has dataset module. This module allows to download the MNIST data in binary format.

“torchvision.transforms” library was used for augmentation for MNITS dataset. As MNIST datasets was consists of grayscale images so few transformations were required for augmentations. *RandomCrop* with padding 4 pixel and *RandomHorizontalFlip* was used for that transformation. After transformations the train images were saved into disk. Before transformation, the size of the binary training images was 9.4 Mb. After augmentation size increased to 45 Mb.

In this Image GPT model, the input to the network is a sequence of image patches, where each patch is a small square image of size 28x28 pixels. The patches are flattened

and concatenated into a long sequence, which is then processed by a multi-layer transformer network to generate a sequence of output patches, which are combined to form the final image.

For each label between 0-9, 16 clusters were taken for each of them and then centroid was calculated for each of the clusters. The GPT2 class initializes the start of sequence (sos) token, the token embeddings, and the position embeddings. It then applies the transformer block multiple times to the input sequence, concatenating the sos token and adding positional embeddings to the input before each transformer block. The imageGPT was trained against three different configuration which were small, medium, and large. While training checkpoints were created on 131000, 262000, 524000 and saved during training. After successfully training the saved model checkpoint was used for image generation and classification.

Top of Form

# Results

This model has the capability to generate completed images as well as predict the label of the images also known as classification.

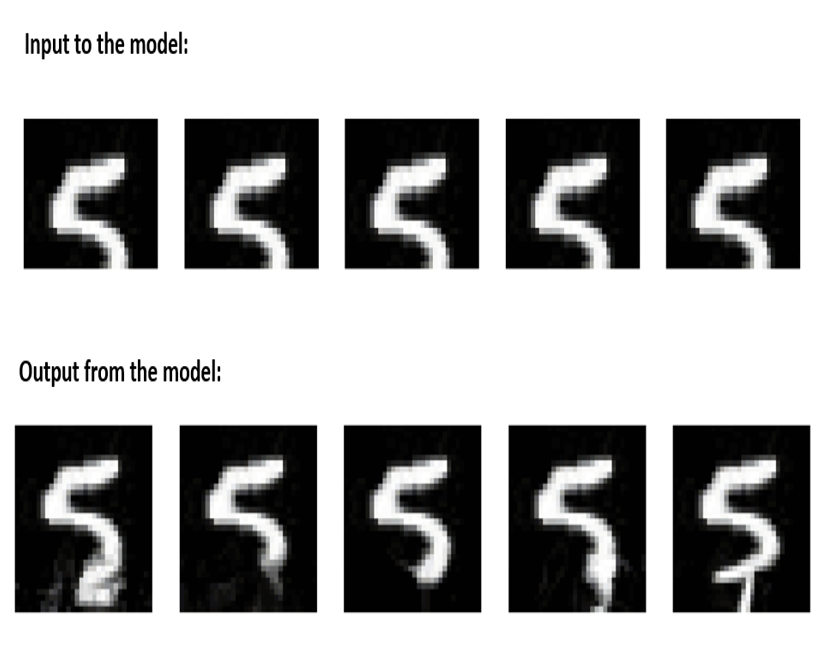


Figure 4: Full context image prediction by the model

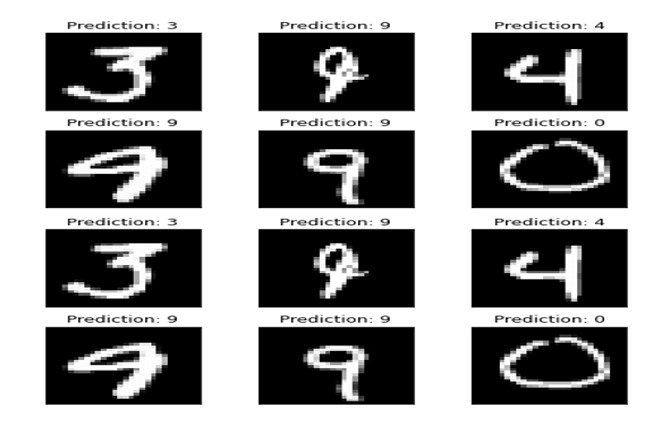


Figure 5: Label prediction by the model

To verify the the correctness with which the labels are predicted and generated I chose to use confusion matrix as a metric for this classification problem as can be seen below:

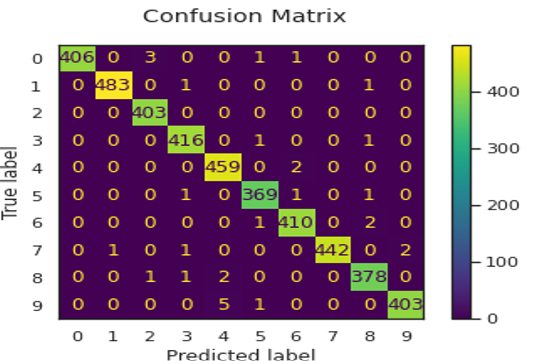


Figure 7: Confusion matrix generated on test data

From figure 7, we see that the majority of the numbers lies in the diagonal of the matrix, stating that the model predicts correctly in most of the cases. To further validate, I decided to plot a graph of actual vs predicted label where, it was again verified that the majority of the points lie in the diagonal, proving that the deviation from actual to predicted is less.

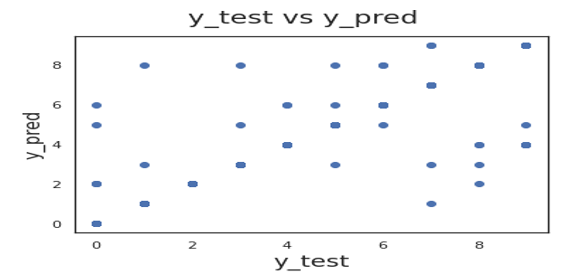


Figure 8 : Plot of Actual vs predicted label

I calculated kappa score which came out to be 0.992. As the kappa score for the full model (with cut-off probability 0.5) is more than 0.99, it can be assumed that there is substantial agreement between the actual and predicted values.

**Model comparison:**

On plotting the image completion loss between iGPT-S, iGPT-M and iGPT-L it was clearly visible that the loss of iGPT-L was the least, as the number of parameters are higher compared to the other models.

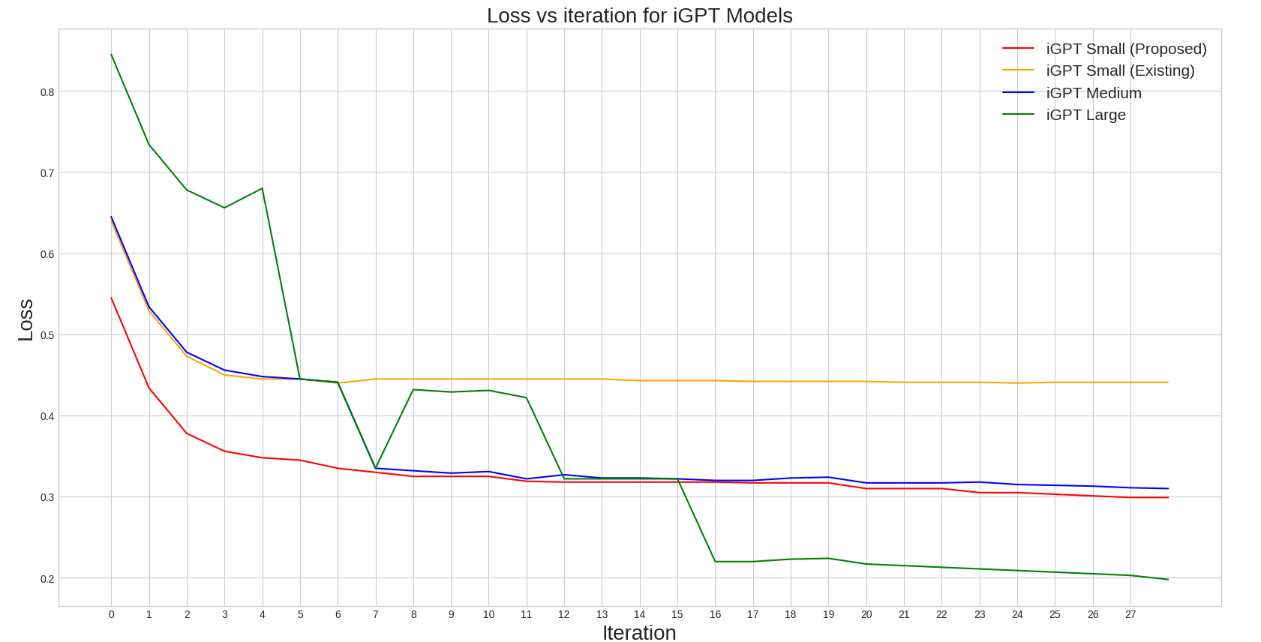
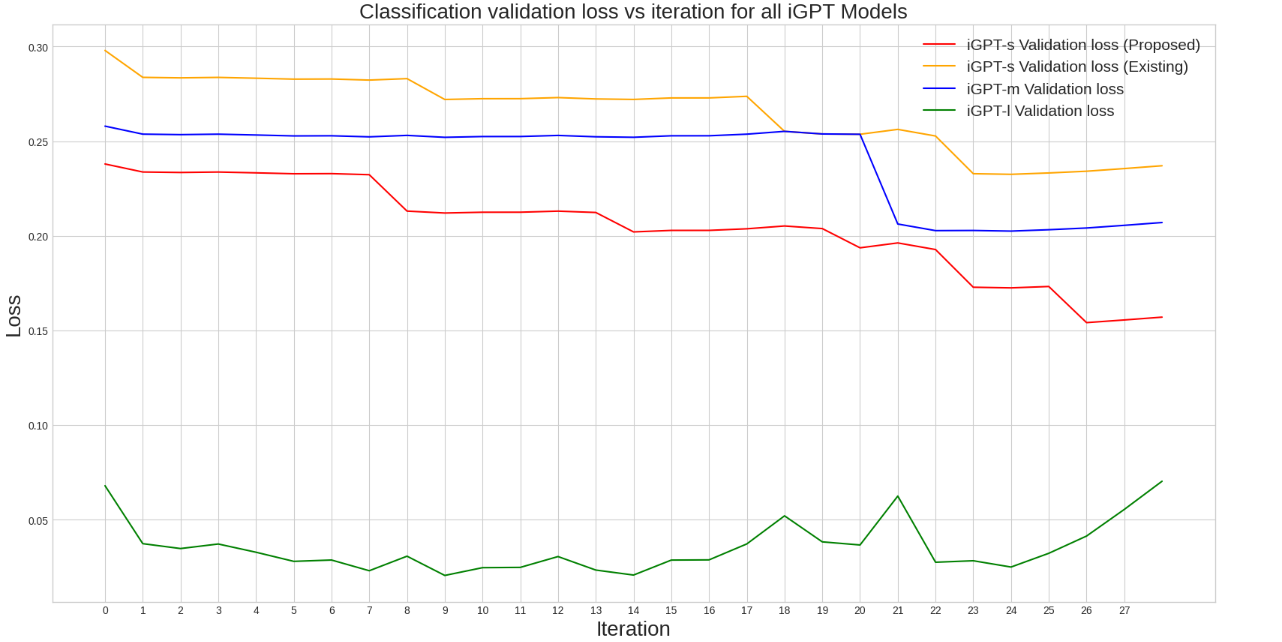
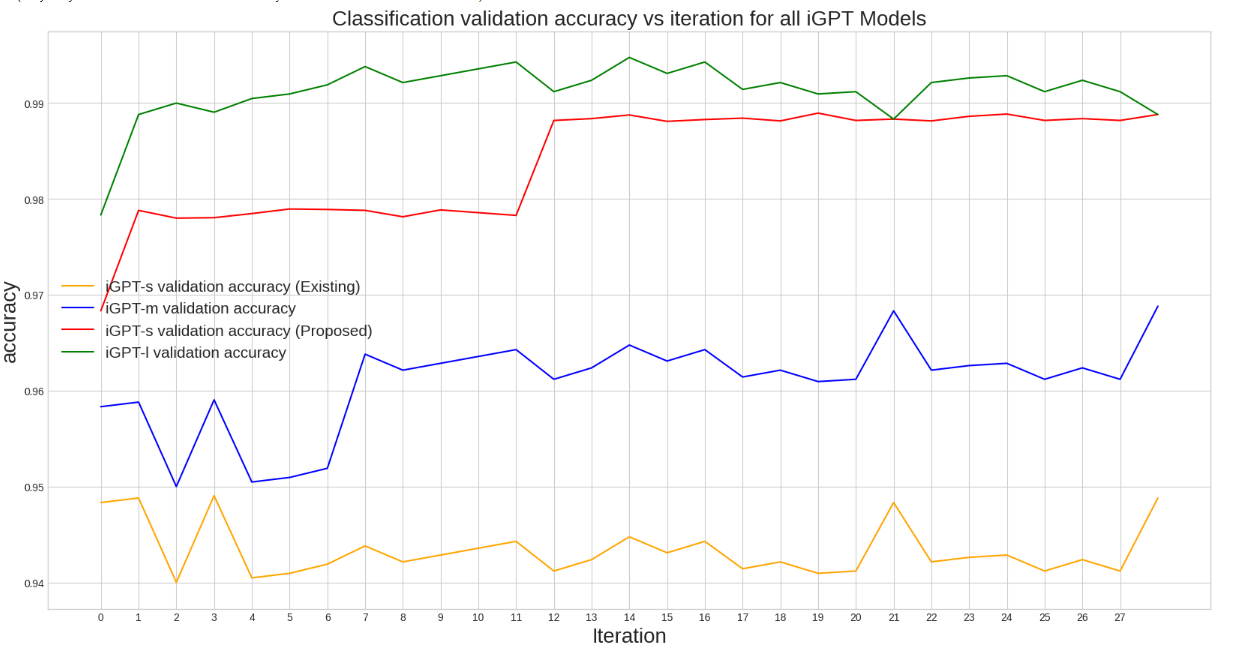


Figure 9 : Image completion loss for iGPT models

Since this model is also capable to perform image classification, so I decided to do a comparative analysis on the iGPT models.



From the above figures we can conclude that iGPT-large model has the highest accuracy and the lowest loss among all other iGPT models, this is because iGPT-large has the highest number of parameters and trained on more epochs. As a result, we got the best prediction for both image completion as well as classification using iGPT-Large models.

# Discussions and conclusions

Image Completion is still a challenging task for deep learning NLP, especially when it comes to domain-specific corpora like MNIST dataset. Image completion, also known as image inpainting, is a computer vision task that involves filling in missing parts of an image. This technique can be used for a variety of applications, including photo editing, restoration, and reconstruction. The imageGPT model has been trained for two tasks. One is image completion and second one is image classification. The image completion task for Image GPT involves generating plausible and coherent image completions for partial input images. Image classification is a computer vision task that involves predicting the class or category of an image based on its content. In the context of Image GPT, the image classification task involves training a neural network model to classify images into one of several predefined categories based on their visual features.

Image GPT models have shown great promise in a variety of applications, including image captioning, image classification, and image synthesis. For example, the Image GPT model has been shown to generate highly accurate and natural language descriptions of images. These models can be

used to automatically generate captions for images in large datasets, making it easier for humans to search and browse through these datasets. In addition to image captioning, Image GPT models can also be used for image classification tasks. For example, Image GPT models have been used to classify images of different types of animals, plants, and even human emotions. These models can help us better understand and categorize visual data, which can have applications in fields such as healthcare, robotics, and autonomous vehicles. Finally, Image GPT models can also be used for image synthesis. This involves generating new images based on textual input, such as a written description or a list of attributes. For example, an Image GPT model could be trained to generate realistic images of birds based on a textual description of their physical characteristics. This could have applications in fields such as fashion, architecture, and graphic design.

The future scope of Image GPT models is vast and exciting. One area of future development is in the generation of more colored images. The same model could be extended to include image completion as well as classification of colored images. The potential of iGPT-XL can also be evaluated with the help of higher computing resources. It has a scope to perform better than iGPT-L. In conclusion, Image GPT models represent a major advancement in the field of computer vision. These models have shown great promise in a variety of applications, and the future scope of these models is vast and exciting. As Image GPT models continue to develop and improve, we can expect to see even more innovative and useful applications of this technology.

Top of Form

# References

1. Chen, Mark and Radford, Alec and Child, Rewon and Wu, Jeffrey and Jun, Heewoo and Luan, David and Sutskever, Ilya. *Generative pretraining from pixels.* International conference on machine learning. Pages 1691-1703, PMLR, 2020.
2. Luo, Renqian, Sun, Liai , Xia, Yingce , Qin, Tao Zhang, Sheng , Poon, Hoifung, Liu and Tie-Yan. *BioGPT: generative pre-trained transformer for biomedical text generation and mining.* Briefings in Bioinformatics, Oxford Academic, 2022.
3. Su, Nigel, Yixuan and Collier. *Contrastive search is what you need for neural text generation.* arXiv preprint arXiv:2210.14140, 2022
4. Chang, Ernie and Shen, Xiaoyu and Zhu, Dawei and Demberg, Vera and Su, Hui. *Neural data-to-text generation with lm-based text augmentation.* arXiv preprint arXiv:2102.03556, 2021

1. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention is all you need.* corr abs/1706.03762 (2017). 2017.
2. Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. *Language*

*models are unsupervised multitask learners.* 2018. URL- *https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf*

1. Yang Liu and Mirella Lapata. *Text summarization with pretrained encoders.* arXiv preprint arXiv:1908.08345, 2019.
2. Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. *Exploring the limits of transfer learning with a unified text-to-text*

*transformer,* 2019.

1. Zhen Huang, Shiyi Xu, Minghao Hu, Xinyi Wang, Jinyan Qiu, Yongquan Fu, Yuncai Zhao, Yuxing Peng, and Changjian Wang*., "Recent trends in deep learning based open-domain textual question answering systems.,"* 2020.
2. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *Bert: Pre-training of deep*

*bidirectional transformers for language understanding,* 2018.

1. Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi,

Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew*. Huggingface’s*

*transformers: State-of-the-art natural language processing,* 2019.

1. Derek Miller. *Leveraging bert for extractive text summarization on lectures*, 2019.
2. Dima Suleiman and Arafat Awajan. *Deep learning based abstractive text summarization: approaches, datasets, evaluation measures, and challenges. Mathematical problems in engineering,* 2020.
3. Philipp Hartl and Udo Kruschwitz. *Applying automatic text summarization for fake news detection.* arXiv preprint arXiv:2204.01841, pages 1–8, 2022.
4. Abigail See, Peter J Liu, and Christopher D Manning. *Get to the point: Summarization with pointer-generator networks.* arXiv preprint arXiv:1704.04368, 2017
5. Dmitrii Aksenov, Julián Moreno-Schneider, Peter Bourgonje, Robert Schwarzenberg, Leonhard Hennig, and Georg Rehm. *Abstractive text summarization based on language model conditioning and locality modeling.* arXiv preprint arXiv:2003.13027, 2020.
6. Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. *Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter*, 2019
7. Shakshi Sharma, Ekanshi Agrawal, Rajesh Sharma, and Anwitaman Datta. *Facov: Covid-19 viral news and rumors fact-check articles dataset. In Proceedings of the International AAAI Conference on Web and Social Media,* volume 16, pages 1312–1321, 2022.
8. Ruixuan Zhang, Zhuoyu Wei, Yu Shi, and Yining Chen. Bert-al: Bert for arbitrarily long document understanding. 2019.
9. Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Darrin Eide, Kathryn Funk, Rodney Kinney, Ziyang Liu, William Merrill, Paul Mooney, Dewey Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex D. Wade, Kuansan Wang, Chris Wilhelm, Boya Xie, Douglas Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. *Cord-19: The covid-19 open research dataset,* 2020