IMAGE DEBLURRING EXPERIMENTS WITH TRANSFORMER BASED MODELS AND AUTOENCODERS (CMP719 SPRING2023 PROJECT)

Tahir Buyukbasaran
N22142989

ABSTRACT

Image restoration, a demanding task in computer vision research, is consistently required in various industries. In this project, a specific focus was given to image deblurring experiments as a subcategory of restoration. The Restormer Zamir et al. (2022) model served as the initial framework, which underwent modifications to enhance its performance. Additionally, two novel autoencoder models were introduced. All experiments were conducted within the context of motion deblurring. The performance of these models was evaluated and compared to determine their effectiveness.

1 INTRODUCTION

Convolutional neural networks (CNNs) O'Shea & Nash (2015) have been widely utilized for image restoration tasks, but recent advancements have shown that Transformers Vaswani et al. (2017) offer improved performance in certain areas. However, Transformers suffer from computational complexity issues that limit their application to high-resolution image restoration tasks. This project focuses on enhancing the performance of the Restormer model, a Transformer-based image restoration model.

Restormer was originally designed to address the drawbacks of CNNs, such as their limited receptive field and inadaptability to input content. Nevertheless, the quadratic complexity issue of Transformers poses challenges, particularly when dealing with high-resolution images.

The project involves implementing modifications to enhance the quality of the restored images produced by Restormer. The performance of the modified Restormer model is evaluated and compared with the original Restormer model and two other novel autoencoder models that were developed from scratch. Specifically, the focus is on motion deblurring tasks, and the evaluation metrics used include Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM).

The results of the experiments demonstrate that the modified Restormer model yields promising outputs with competitive PSNR and SSIM values. Additionally, one of the implemented autoencoder models exhibits satisfactory performance when compared to the complex Restormer and Restormer-based model.

In summary, this project first to attempt to improve the performance of the Restormer model which is Transformer based. Furthermore, the project explores two other autoencoder models and compares their performances with the Restormer and Restormer based models, providing valuable insights into the effectiveness of different approaches in image restoration tasks.

The codes and sample images of the project can be find at `https://github.com/tahirbb/CMP719`

2 MODELS

2.1 RESTORMER BASED MODEL (MODEL1)

As a Unet shaped transformer-based model called **Restormer** Zamir et al. (2022) is proposed for image restoration, which captures global connectivity and is applicable to large images. The model introduces the Multi-Dconv Head 'Transposed' Attention (MDTA) block as a replacement for the vanilla multi-head self-attention mechanism Vaswani et al. (2017). The MDTA block performs self-attention across feature dimensions, computing cross-covariance across feature channels to obtain attention maps. Local context mixing is performed prior to covariance computation using pixel-wise and channel-wise aggregations, achieved through 1x1 convolutions and depth-wise convolutions, respectively.

Furthermore, the model incorporates a Feed-Forward Network (FN) consisting of two fully connected layers with a gating mechanism. The first linear transformation layer of the regular FN is reformulated with a gating layer, which is the element-wise product of two linear projection layers activated by the GELU Hendrycks & Gimpel (2023) non-linearity. The Gated-Dconv FN (GDFN) also adopts local context mixing, similar to the MDTA module, to equally emphasize spatial context. The gating mechanism in GDFN facilitates the flow of complementary features, enabling subsequent layers in the network hierarchy to focus on more refined image attributes and produce high-quality outputs.

In this project, as showed in Figure 1 some modifications implemented to enhance the Restormer's Zamir et al. (2022) performance by introducing a new branches to the network. The purpose of this modification is to allow information to flow to the output prior to the skip connections of the U-shaped networks decoder side . Towards the end of the network, the new branches are incorporated, and it combines the input image with the outputs of the original U-shaped network using a **softmax temprature** . Additionally, components such as the transformer and upsampling blocks from Restormer were utilized in this project.

In the original Restormer network, three layers of encoding and decoding blocks are utilized, with each decoding block receiving skip connections from the corresponding encoding layer. To enhance the model's performance, a key improvement was made by upsampling the output of the L4 and L3 transformer blocks until it matched the dimension of the L1 transformer input. Subsequently, the refined transformer block of Restormer was applied to the upsampled output.

With this modification, the new network incorporates three distinct sources of information: the output generated by the original Restormer, the information from the L4 branch extended network, and the information from the L3 branch extended network. To combine these sources effectively, a softmax temprature is employed, which normalizes the outputs using softmax and computes a weighted sum. Furthermore, the input image is added to the aggregated output, providing a more comprehensive representation. This approach allows the model to benefit from a balanced combination of information and achieve improved performance.

Experimental results demonstrate that the proposed model surpasses the performance of the original Restormer network, highlighting its effectiveness in image restoration tasks.

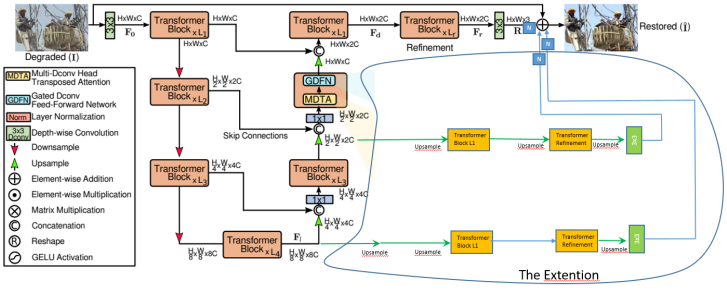


Figure 1: The Model1

2.2 FIRST AUTOENCODER BASED MODEL (MODEL2)

The proposed **Model2**, depicted in Figure 2, comprises two distinct encoder-decoder pairs (Modules) for image deblurring. These modules are equipped with various convolutional layers to capture different information from the input images. The Restormer based FeedForward network serves as the latent layers, incorporating convolutional layers and GELU-activated element-wise multiplication operations to refine the learned representations.

Both Module1 and Module2 encoder-decoder pairs exhibit gradual reduction and subsequent increase in spatial dimensions are achieved through the use of transposed convolutions. The encoder components employ convolutional layers followed by activation functions, including ReLU and GELU, while dropout layers are incorporated for regularization purposes. Max-pooling layers facilitate downsampling of the feature maps.

The decoder component is responsible for reconstructing the images from the latent space representation. It employs transposed convolutional layers, also known as deconvolution, to upsample the feature maps and gradually restore the original image size. Activation functions such as GELU and ReLU introduce non-linearity to the decoding process. The final layer utilizes the Sigmoid function.

In the output of Model2, the reconstructed output is combined with the blurred input image, and the resulting image is clamped between 0 and 1 using the `torch.clamp` function. This mechanism ensures that the pixel values remain within a valid range.

A notable characteristic of this model is the adoption of a two-times autoencoder architecture. This approach involves performing the encoding and decoding processes twice, enabling the model to capture finer details and achieve more refined image reconstruction. Additionally, a clamp mechanism is employed to restrict the pixel values and maintain the output images within the appropriate range.

In detail the differences of the Model1 Encoder and Decoder structures of them are:

Encoders:

- **Architecture:** The Module2 consists of simpler convolutional layers (Conv2d) with ReLU activation functions and max pooling (MaxPool2d) operations. In contrast, Module1 employs more complex convolutional layers, including groups and different kernel sizes, along with GELU activation functions and additional dropout (Dropout) layers.
- **Input Processing:** The Module2 processes the input image by gradually reducing its spatial dimensions while increasing the number of channels. Module1 performs similar operations but with a different set of convolutional operations and downsample operations.
- **Capturing Features:** The Module2 aims to extract general features from the input image. In contrast, Module1 is designed to capture more complex and abstract features by leveraging a different set of convolutional operations. Module1 learns richer representations compared to the regular encoder.

Decoders:

- **Architecture:** The Module2 consists of transpose convolutional layers (ConvTranspose2d) with ReLU activation functions and dropout (Dropout) layers. It aims to reverse the operations of the encoder, gradually increasing the spatial dimensions while reducing the number of channels. On the other hand, Module1 employs transpose convolutional layers with GELU activation functions and dropout layers.
- **Upsampling and Reconstruction:** The Module2 uses transpose convolutional layers to perform upsampling operations and reconstruct the spatial details lost during encoding. Module1 employs similar operations but with the use of GELU activation functions instead of ReLU.
- **Output Generation:** Both the Module2 and Module1 generate the reconstructed image by adjusting pixel values using activation functions (e.g., Sigmoid for the decoder and GELU for Decoder1).

In summary, the differences between Module1 and Module2 lie in the architecture, types of convolutional layers, activation functions, and the complexity of feature extraction. Module1 Encoder and

Decoder are designed to capture and reconstruct more complex and abstract features compared to the Module2. These differences allow the Autoencoder model to learn and represent different levels of abstraction in the input data.

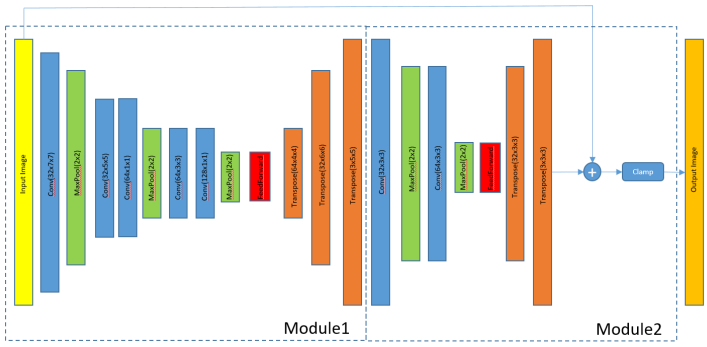


Figure 2: The Model2

2.3 Second Autoencoder Based Model (Model3)

As an alternative autoencoder model, the more complex **Model3** consists of three interconnected autoencoder modules, aiming to increase the model's capacity for fine-tuning. However, the results obtained from this model are inferior to those of **Model1** and **Model2**.

The first module preserves the spatial size of the input data, deviating from the standard autoencoder structure. The remaining modules follow a gradual decrease and increase in size. The encoder sides of the third and second modules utilize convolution filters with varying sizes, progressively reducing in size. The latent layer of this model is derived from the Restormer FeedForward network.

The outputs of each module are added to the subsequent module's output, with a **weighted** addition approach that differs from **Model2**. Clamping is not implemented to preserve the original input information.

In summary, **Model3** is a more complex model designed to extract additional information from the input image.

In detail the Encoder-Decoder parts of the the Module comperative descriptions;

Encoder Branches:

- **Module1:** The encoder in the proposed model consists of multiple convolutional layers with different filter sizes, activation functions, and max-pooling operations. It aims to extract and encode important features from the input image, enabling subsequent decoding and reconstruction processes. The specific architecture includes 3 convolutional layers with 256, 128, and 64 filters respectively, followed by PReLU He et al. (2015) activation functions and max-pooling to downsample the feature maps. The encoder outputs a compressed representation of the image with 10 channels.
- **Module2's:** The encoder module in the model consists of a sequence of convolutional layers with varying kernel sizes, strides, and padding. It begins with a convolutional layer that takes 3 input channels and produces 32 output channels. The subsequent layers follow a similar pattern, including convolutional layers with different configurations, PReLU activation functions, dropout layers, and max-pooling operations for downsampling. The first convolutional layer has a kernel size of 7. The subsequent layers include convolutional layers with kernel sizes of 5, 1, and 3, each followed by appropriate activation functions, dropout layers, and max-pooling operations. The concept of groups is applied to some of these layers, where the input channels are divided into groups for independent convolution operations.

Decoder Branches:

- **Module1:** The decoder module in the model comprises a sequence of transpose convolutional layers responsible for reconstructing the original image. It begins with a transpose convolutional layer that takes 10 input channels and produces 128 output channels. The subsequent layers follow a similar pattern, including transpose convolutional layers with different configurations, PReLU He et al. (2015) activation functions, and dropout layers for regularization. The first transpose convolutional layer has a kernel size of 3.The following layers include transpose convolutional layers with different kernel sizes and configurations, each followed by a PReLU activation function and a dropout layer with a dropout probability of 0.5. The last layer in the decoder module is a transpose convolutional layer with 32 input channels, 3 output channels, a kernel size of 4, and a stride value of 2 for upsampling the feature maps. A padding of 1 is applied to ensure the output size matches the original image dimensions. The Sigmoid activation function is then applied to normalize the pixel values of the output image between 0 and 1.

- **Module2's:** The decoder module in the model uses a series of transpose convolutional layers for upsampling and reconstructing the image. It starts with a transpose convolutional layer that takes an input of 128 channels and produces an output of 64 channels. The layer has a kernel size of 4. The output is then passed through a PReLU activation function.A dropout layer with a dropout probability of 0.5 follows, providing regularization. Next, a transpose convolutional layer takes an input of 64 channels and generates an output of 32 channels. The layer has a kernel size of 6, stride of 2, padding of 1, and output padding of 1. Again, a PReLU activation function is applied, followed by another dropout layer with a probability of 0.5. The final transpose convolutional layer takes an input of 32 channels and produces an output of 3 channels. It has a kernel size of 5, stride of 2, padding of 1, and output padding of 1. The output is then passed through a Sigmoid activation function, ensuring that the pixel values of the reconstructed image are normalized between 0 and 1.

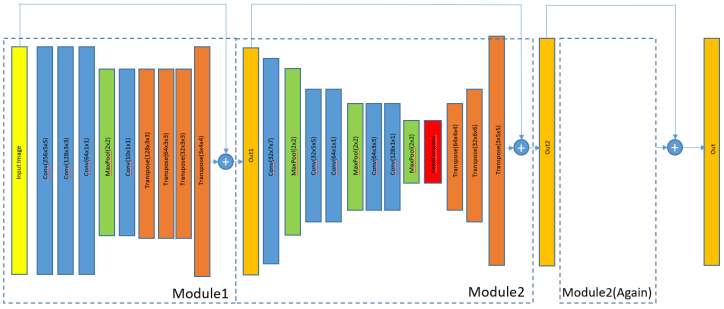


Figure 3: The Model3

3 EXPERIMENTS

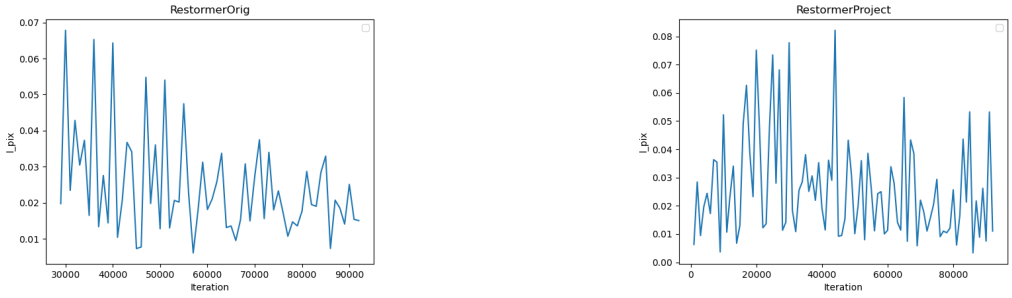
The experiments conducted in this project can be divided into two sections. Firstly, the performance of the Restormer-based model was assessed through a comparison study. For this comparison, both the original Restormer Zamir et al. (2022) and the Restormer-based **Model1** were trained using the GoPro dataset Nah et al. (2017) to ensure a fair evaluation. Each model underwent training in the same environment to enable proper comparison.

In the second section, a motion-blurred text dataset called BMVC Hradiš et al. (2015) was utilized for performance comparison. In this stage, the performances of the Original Restormer, Model1-2, and Model3 were compared. The focus was on analyzing how the structural differences between Model2 and Model3 impacted the output of the models. Detailed discussions were conducted to gain insights into the observed variations in performance.

3.1 TRAINING

3.1.1 MODEL1 TRAINING

In this project motion deblurring task has been focused. For the training task GoPro motion blurred datased has been used and both original Restormer and the **Model1** has been trained on the same environment. Both of the models has been trained for 92.000 iterations. Since the complexity of the Model1 has been increased 92.000 iterations corresponds to the 5 epoch on the other hand the same number of iterations corresponds 8 epoch for the original Restormer. As the training metric l_pix value has been observed which can be considered as pixel-wise loss or pixel-level loss in image processing and computer vision tasks. It represents the discrepancy or difference between the predicted output and the ground truth at the pixel level. The l_pix value provides a quantitative measure of the pixel-level dissimilarity between the predicted and ground truth images.



(a) Original Restormer

(b) Model1

Figure 4: Training Graphs for Original Restormer and Model1

As observed in Figure 4a, a gradual decrease in the l_pix values of the Restormer is evident. On the other hand, as seen in in the Figure 4b the l_pix values of **Model1** exhibit more fluctuations, albeit with an overall decreasing trend. Notably, after the 50,000th iteration, the l_pix values of Model1 consistently fall below 0.01. This trend suggests that as the number of epochs increases, Model1 is expected to provide more accurate results compared to Restormer.

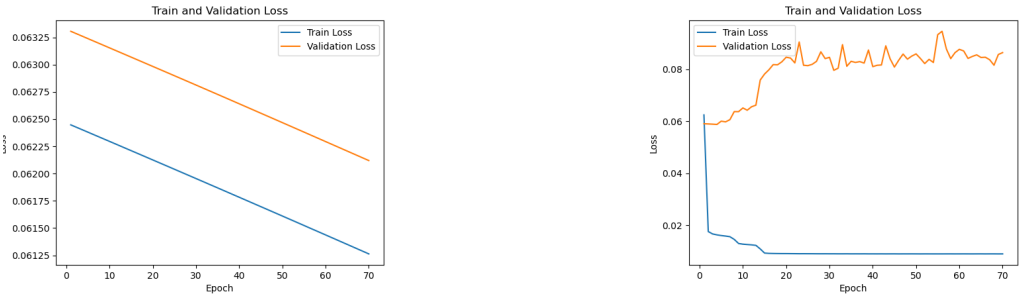
3.1.2 MODEL2 AND MODEL3 TRAINING

For the purpose of result comparison in motion blurred image restoration, particularly in the restoration of motion blurred text images, the BMVC dataset Hradiš et al. (2015) was employed for training. Due to limitations in hardware resources, a small subset of the dataset was selected for both training and testing. A total of 10,000 images from the BMVC dataset were utilized, with 70% allocated for training, 10% for validation, and 20% for testing.

In **Model2**, data normalization was applied, setting the mean and standard deviation to 0.5. The Mean Squared Error (MSE) was employed for the loss , and a learning rate of 0.0001 and weight decay of 0.001 (penalty) were utilized with the SGD optimizer to prevent overfitting. Dropout was also incorporated in the model to further mitigate overfitting.

As for **Model3**, data normalization was not applied. Similar to Model2, MSE loss calculation was employed, but the Adam optimizer was used instead. The learning rate was set to 0.0001, and a weight decay of 0.001 was employed. Dropout was also utilized to prevent overfitting.

As observed in Figure 5, both models were trained for a total of 70 epochs, as indicated by the train-validation loss graphs. Notably, **Model2** exhibited a consistent decreasing trend, suggesting improved performance over time. Conversely, despite its complexity and low learning rate, **Model3** displayed signs of overfitting, indicating a mismatch between its performance on the training data and its generalization ability.



(a) Model2 Train-Validation Loss

(b) Model3 Train-Validation Loss

Figure 5: Training Graphs for Model2 and Model3

3.1.3 Qualitative Results

As a qualitative comparison the firstly the output of the Restormer and the **Model1** output has been showed in Figure 6. And for the comparison for all model the text blurred restoration results are showed in Figure 7.

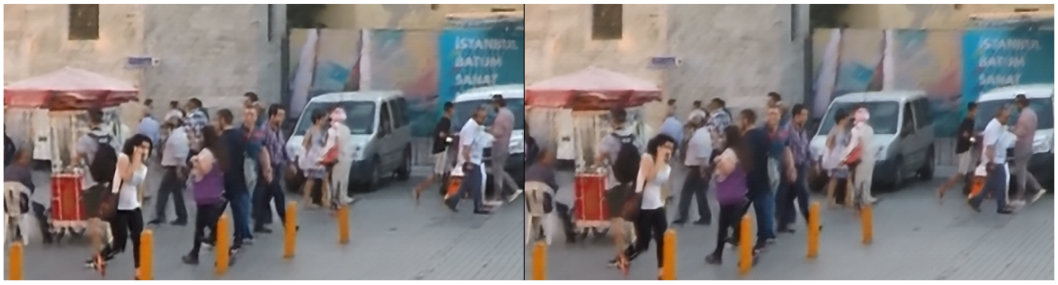


Figure 6: Left **Model1**-Right Restormer

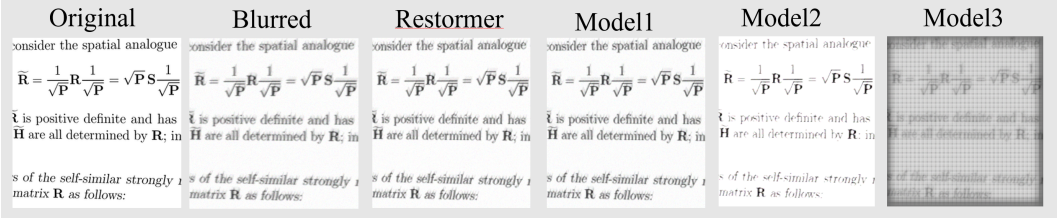


Figure 7: Sample Outputs from BMVC

As observed in Figure 6, it can be seen that the output generated by **Model1** exhibits a clearer result, particularly in the areas with text. The differences between the models become more apparent in these texted areas.

In Figure 7, the outputs of the original Restormer and **Model1** are found to be very similar. However, the output of **Model2** does not closely resemble the original image, but effectively suppresses the blurred areas while maintaining the black regions. The output can be regarded as a filtration of the blurred image, with an increased contrast. It appears to be more suitable for text extraction operations.

On the other hand, despite its complexity, **Model3** produces the poorest output. The presence of black areas around the output image indicates the effects of overlapped padding, and the grid-shaped black dots indicate inefficient positioning of the receptive fields in the convolutional layers. The training process also suggests that **Model3** has not been effectively learned. To improve the per-

formance of Model3, it is recommended to precisely tune the initial convolutions and increase the number of training images for enhanced training.

3.1.4 QUANTITATIVE RESULTS

Since the problem is image restoration PSNR and SSIM values has been compared. PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index Measure) are both metrics used to assess the quality of image or video reconstructions compared to their original versions.

PSNR focuses on the numerical difference between images, while SSIM considers both numerical and perceptual factors to measure similarity between images.

Firstly, the comparison between The Original Restormer and **Model1** was conducted, as presented in Table 1. According to the table, the best SSIM values were achieved by **Model1**, while the highest PSNR value was obtained by Restormer. When considering the same number of epochs, **Model1** exhibited superior performance in terms of both PSNR and SSIM values.

Model	PSNR	SSIM
Restormer (5 epoch-68.000 iterations)	27.6073	0.8968
Restormer (8 epoch-92.000 iterations)	**27.628**	0.8973
Model1 (5 epoch-92.000 iterations)	27.6118	**0.8975**

Table 1: Restormer and Model1 Comparisons with GoPro Dataset

Secondly, the calculation of mean PSNR and SSIM values was performed on a subset of the BMVC dataset. As demonstrated in Table 2, the best PSNR value was achieved by **Model1**, while surprisingly, Restormer attained the highest SSIM score with only 5 epochs. **Model2** yielded competitive results, considering its simplicity, limited training dataset, and number of training cycles. However, **Model3** exhibited the poorest results among all models.

Model	PSNR	SSIM
Model1	**17.8736**	0.7465
Restormer (5 Epoch)	17.8402	**0.7509**
Restormer (8 Epoch)	17.7002	0.7391
Model2	16.4436	0.7401
Model3	11.135	0.2584

Table 2: All Models Comparisons with BMVC subset

4 DISCUSSIONS

In this project, three different approaches were employed for motion deblurring in image restoration. Firstly, an improvement was made to the state-of-the-art (SOTA) Restormer Zamir et al. (2022) model to enhance its performance despite its complexity. Additionally, two autoencoder models were implemented specifically for motion deblurring problem. Throughout the project, the impact of dataset normalization, model complexity, and convolutional receptive filters was observed.

To achieve better results, the complexity of the model was increased. Measures such as weight decay and dropout were applied to prevent overfitting, and a small learning rate was set to obtain well-fitted weights. To address the vanishing gradients problem and enable information flow between modules, the outputs of all modules were added. In the case of **Model3**, a weighted output was also employed.

Based on the experimental results, it was found that achieving a balance between model complexity, a diverse and balanced training dataset, appropriate convolution filter selection, upsampling techniques, learning rate, and regularization is crucial for obtaining optimal results. Various experiments were conducted during the project, including the use of SuperResolution, custom ReLU activations, vanilla autoencoders with linear and symmetric structures, UNet CNN models, and data augmentation techniques. Among these experiments, **Model1** and **Model2** demonstrated promising results, while **Model3** provided valuable insights and lessons learned.

REFERENCES

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus), 2023.

Michal Hradiš, Jan Kotera, Pavel Zemčík, and Filip Šroubek. Convolutional neural networks for direct text deblurring. In *Proceedings of BMVC 2015*. The British Machine Vision Association and Society for Pattern Recognition, 2015. ISBN 1-901725-53-7. URL `http://www.fit.vutbr.cz/research/view_pub.php?id=10922`.

Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *CVPR*, July 2017.

Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks, 2015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration, 2022.