Generalization by Visual Attention

Baptiste Colle
Supervisor: Wendelin Böhmer
EEMCS, Delft University of Technology, The Netherlands

June 18, 2022

A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

Generalization by Visual Attention

Baptiste Colle
Supervisor: Wendelin Böhmer
EEMCS, Delft University of Technology, The Netherlands

Abstract

Most deep learning models fail to generalize in production. Indeed, sometimes data used during training does not completely reflect the deployed environment. The test data is then considered out-of-distribution compared to the training data. In this paper, we focus on out-of-distribution performance for image classification. In fact, transformers, which are a novel neural network architecture compared to the more traditionally used convolutional neural networks (CNN), have been shown to work well for image classification. This is why, in this paper, we firstly explore the different capabilities of both models on out-of-distribution. This is then followed by an in-depth investigation of individual architectural components of the transformer and their impact on the generalization capability of the model.

1 Introduction

Image classification is a fundamental task in computer science and namely computer vision. The goal is to identify the objects present in an image [1].

The most commonly used type of network for this task is convolutional neural networks (CNN), however, they can perform poorly on out-of-distribution images. Out-of-distribution happens when training and test data are not drawn from the same distribution and thus decrease the model generalization capabilities. Generally, we see that there is a drop in accuracy when we change between distributions.

Due to this drop, an alternative to CNNs for image classification could be the visual transformer [2]. This method applies transformers that are typically used in natural language processing (NLP) [3] to the field of computer vision. However, the visual attention paper did not investigate the impact of transformers on out-of-distribution, therefore it is still an open question. In this paper, we will explore the multi-head attention (MHA) mechanism of the transformer as we think this could be the key to better out-of-distribution generalization.

This research investigates the difference in the performance of the two previously mentioned models concerning out-of-distributions images. We believe that such a difference could exist due to the visual attention mechanism of the transformer. This property of the model allows it to focus its awareness on previously learned patterns and ignore the unknown features of the image.

1.1 Motivation

Out-of-distribution generalization is a core concept in AI. In fact, by being able to generalize concepts to unseen data and extract meaningful features, the network can use previously learned features in a novel manner. The network is able to transfer knowledge from training data to new unseen data by identifying previously recognized patterns. However, it is often difficult to have training data that reassembles data where the model would be deployed or when new data types come into the system. This is why investigating the capacity for out-of-distribution generalization of the current most used image classification architecture, and the novel transformer architecture could help create future systems that can better tackle the current shortcomings.

1.2 Research Question

The problem that we researched is whether the transformer's attention mechanism could lead to better out-of-distribution than current methods, namely CNNs. Indeed, we investigated different network configurations to see how one can improve out-of-distribution performance and which elements lead to better results. This leads us to ask the following question:

Research Question: *Which network configurations have the largest impact on out-of-distribution performance in both architectures?*

To answer this question, section 2 will lay down the theoretical foundation of our problem. This is followed by an explanation of our approach to answer our research question in section 3. We will then discuss the experimental results in section 4. To finish our research we will discuss the ethical implication of our work in section 5 and conclude by how our research fits into the current machine learning literature in section 6.

2 Background Information

2.1 Convolutional neural network

CNNs are the most commonly used network architecture for image classification. First introduced by LeNet [4] and then made famous by AlexNet [5], they are widely used in research and industries.

CNNs work by extracting a feature map, the output activation of the convolutional operation, from an image based on their kernel, a set of weights. This allows the network to gradually build a representation of an object's features, such as edge, position, objects, etc.

In the context of a 2-D image, a convolution is the act of placing the kernel in the top left region of the image, computing the weighted sum, and then shifting the kernel to the right. This operation is done row by row until the feature map is complete. This can be seen graphically in figure 1

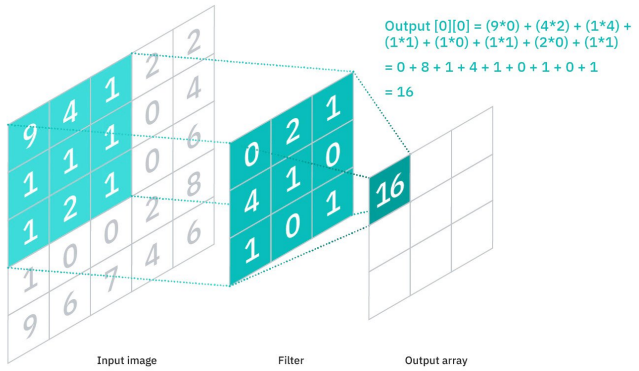

Figure 1: Convolutional operation in CNN [6]

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n) K(m,n) \tag{1}$$

The mathematical convolutional operation is defined as seen above in equation 1, with K representing the weights of the kernel and (i, j) the coordinate of the pixel in the image.

Indeed, even with such a simple operation, convolution neural networks have been shown to perform well on images due to their numerous invariants.

Sparse interactions

When working on images, our input is often composed of many thousands of pixels. Feeding it to a traditional fully connected neural network would require a model with just as many parameters. Additionally, meaningful information is likely found in just a small subset of the image, therefore having thousands of weights set to 0 is very inefficient. The convolutional operation helps speed up computation as only a subset of the image of size k is considered at a time, thus only a subset of pixels or features are connected to each neuron. There could be preoccupations with the convolution that if not all nodes of the network are connected then information could be lost compared to a fully connected layer. Fortunately, when used in *deep* networks with many layers the nodes will *indirectly* interact, allowing for both efficiency and expressiveness.

Parameter Sharing and Equivariant representations

"Parameter sharing" entails having a single set of weights which will then be multiplied by every single input. This saves on storage but also causes the layers to have *equivariance* to translation. Mathematically a function f is equivariant to a function g if $f(g(x)) = g(f(x))$. In the case of images g could shift all images to the right, this confuses traditional neural networks but a convolution is resistant to these kinds of changes. This property is useful as we are not interested in the *precise* location of features but rather their *presence*. The same feature could appear in any position of an image and still be recognized. Furthermore, CNNs often have multiple convolutional layers which allow deeper layers to detect more complex features [4].

2.2 Transformer

The transformers are a new type of architecture first proposed by Vaswani et al. [3]. This led to a revolution in natural language processing (NLP) and allowed for bigger and more complex architecture. Indeed it parallelizes well, allowing for extensive training on sequence data compared to the previous method, such as Long short-term memory (LSTM) neural network.

More recently, it was introduced to the field of computer vision [2] with the visual transformer (ViT) and started to match the performance of CNNs. Indeed its attention mechanism offers more flexibility than CNNs as they can match more distant features together.

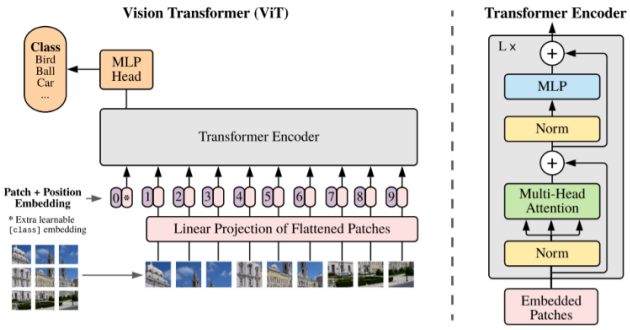

Figure 2: Transformer architecture [2]

Normally a transformer has an encoder and decoder, However, this is not necessary in our case. For image classification, we only need to encode the image representation into a series of tokens and use those to make predictions as seen in figure 2. This contrast with NLP where we want to recreate a new sequence of tokens.

Patching of images

The original ViT uses patches of images as seen in figure 2. The image is cut into multiple patches of the same length and fed into the transformer, this is comparable to the words of a sentence in NLP.

However, the CNN processes images in a different way, the kernel is rolled over the images shifting by the striding distance, often 1. By unfolding the image, we can create the same behavior in the transformer, namely having the same pixel in multiple patches, so that we can compare the behavior of the two models more closely

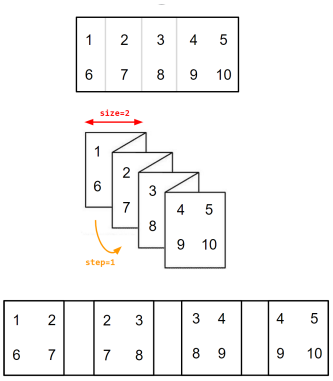

Figure 3: Unfold operation [7]

Indeed as seen in figure 3, we can see that with "unfold" we get a sliding effect over the image, for example, the number two is repeated in the first and second block, this is what we want to be as close to CNN module as possible. This contrast with figure 2 as, for example, the right-most tree is only available in one patch.

2.3 Attention Mechanism

The attention mechanism is the primary part of the transformer. Indeed, this allows the transformer to focus its attention on key features of the images. In order to do so, the transformer architecture has three-parameter learnable weights: the key K_w, the query Q_w and the value V_w. These are respectively multiplied with the tokens, which in our case is the image patches. The resulting matrix, K, Q, V are then fed to the attention mechanism to get a set of attention output and weights. These matrices serve as a linear projection. The size of the K, Q, V are important as they determine the embedding dimension, this is analog to the $out_{channel}$ of a convolutional network. We will use this property to make an analogous block to a convolutional neural network as discussed in section 3.1

In order to obtain our attention score, we use the following formula:

$$Attention(Q, K, V) = softmax(\frac{Q * K^\top}{\sqrt{d_k}}) * V \qquad (2)$$

We use softmax and the embedded dimension of the key d_k, which is determined by the shape of our Q_w, to normalize our result. This gives us the attention of each element of our sequence. We call this, the attention head.

Multi-head attention

Furthermore, the attention mechanism is duplicated to create multiple focal points in an image or sequence of tokens in the case of NLP. We create duplicates of our attention head. The number of attention heads is controlled by a hyperparameter called the number of heads. Following this, all heads are concatenated to give an output matching the embedding dimension.

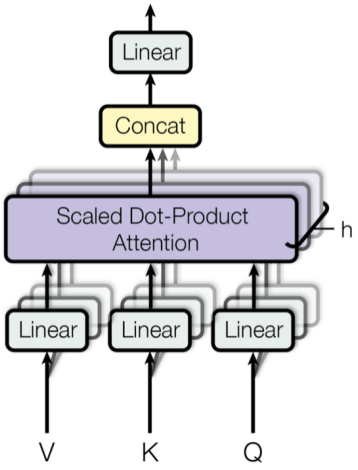

Figure 4: Multi-head attention [3]

Indeed, you can see in figure 4 that the *h* heads can compute the attention of equation 2 in parallel. This leads to a large increase in speed within the performance, with each head being responsible for a subset of the total embedded dimension.

Positional encoding

One of the significant inductive prior of CNN is they are aware of locality, however, a default transformer is not. In order to do so, one must manually encode this positional information into the token. This is done by giving a learnable weight to each patch, so that association between patches can be learned.

Layer norm

Lastly, layer normalization has been proposed as a regulation technique for the visual transformer architecture [3], preventing the weights from becoming too important. Moreover, this layer allows for faster training, as demonstrated in the layer normalization paper [8].

3 Methodology

We studied the effect of different layers and the number of hidden networks on the performance of both architectures. The intention is to see which change produces the most generalization for out-of-distribution images.

This research aims to show that visual transformers have the same robustness when it comes to images. We investigated under which conditions transformers had better accuracy than CNNs. Further tuning of both models was carried out by changing their network architecture. We will see how changing the type of layers and the number of neurons can improve the performance of the task mentioned above.

We will also investigate transformer specific elements such as:

- Positional encoding
- Layer normalisation
- Residual connection
- Number of heads

In addition, we will see the effect of modifying the following aspect of our architecture:

- Network depth
- Max pooling

Indeed the original visual attention paper [2] added multiple elements to the multi-attention head module we will see if adding such elements improve the performance on out-of-distribution.

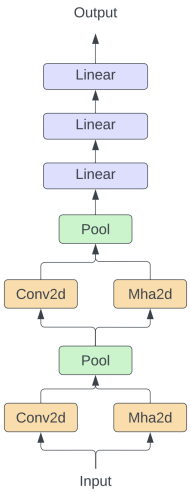

Figure 5: architecture

In order to answer this paper's research question, we will start with a simple network architecture based on the LeNet [4] in both CNN and MHA as seen in figure 5. Secondly, we will extend this architecture by adding components from the transformer architecture. Thirdly, we will investigate the impact of different network configurations on out-of-distribution performance. Lastly, we will see how the number of heads of the transformer impact performance.

Our performance metric is the out-of-distribution accuracy.

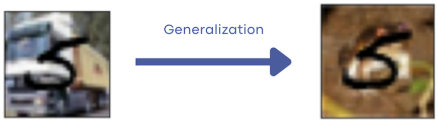

Figure 6: Out-of-distribution images

To stimulate out-of-distribution images, we will use the MNIST and CIFAR10 datasets. We will place the MNIST numbers on top of the background from the CIFAR10 dataset. Furthermore, we will use a varying number of different backgrounds per number during training to see how this impacts the accuracy of both models. As seen in figure 6, we can see that network is trained with the number "5" written on top of an image of a truck, we are now trying to determine if this training allows the network to also recognize the number "5" on top of a flower, for example. We are seeking to answer if the network can generalize its knowledge of numbers to an unknown environment, in our case backgrounds. Furthermore, in our example, all "5" during training are on top of trucks, but for some experiment, we will be varying the number of backgrounds associated with an image from 1 to 64.

Furthermore, each digit is associated with its own background, for example here "5" is associated with trucks but "1" could be for example associated with dogs. So if we decide to have two backgrounds per class (digit), we will have for example trucks and pizza being possible backgrounds for "5" during training and dogs and tomatoes for the number "1".

The goal is that the features learned by our AI models are consistent (invariant) regardless of the environment.

3.1 Conversion from convolution blocks to multi-head attention block

We created a fair comparison of CNN and MHA, as it was denoted by Yutong Bai et al., that comparison of CNNs and transformers are often unequal [9]. In order to do so, we try to make our MHA computation as close to the CNN ones with the only difference being the attention module. This is why we are using unfolding as an operation compared to image to patch as discussed in section 2.2. Indeed, it was critical to embed our image into a sequence of tokens for the multi-head attention to work.

For this project, we relied on the conv2d module from PyTorch and implement our mha2d module that internally relies on pytoch.MultiheadAttention. This module allows for interchangeability between the conv2d and mha2d as the arguments name, dimension, and output dimension are equivalent. Figure 7 illustrated the inner-working of our attention block that can replace convolutional operations.

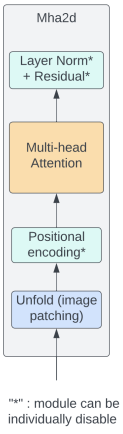

Figure 7: Our custom module equivalent to conv2d from pytorch

As seen in figure 8 below, showing the method signature

of both models, we can easily swap between both. So we can easily compare different network architectures and their performance.

```
CLASS  torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0,
       dilation=1, groups=1, bias=True, padding_mode='zeros', device=None, dtype=None)
```

```
class Mha2d(nn.Module):
    def __init__(self, in_channels, out_channels, kernel_size, stride=1, padding=0,
                 dilation=1, num_head=1, pos_encoding=False, layer_norm=False, residual=False):
```

Figure 8: Comparison of signature of conv2d from pytorch (top) and custom MHA module (bottom)

With the unfolding operation, we can convert our image of size $(32, 32, 3)$ with 3 representing the number of $in_{channel}$ (RGD) into a sequence of token of size $kernel_{size}^2 * in_{channels}$. Furthermore, the embedded dimension of the query, key or value is similar to the $out_{channel}$ of a convolution operation. By making sure that the embedded dimensions of query, key, and value match the $out_{channel}$ we can have an analogous operation to convolutional network as seen in figure 9.

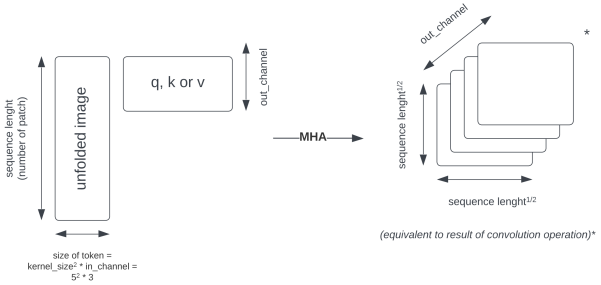

Figure 9: Feeding unfolded image

We can also rely on unfold for adjusting padding, strip, dilation to match the CNN module. So both our methods can be exchanged easily, indeed the only element that still needs to be configured is the hyperparameter of the number of heads. It must be a divisor of embed dimension that is in our case the $out_{channel}$. The MHA module of figure 9 works in the same way as the one explained in the background section equation 2.

3.2 Our hypothesis

Transformers have already been shown to perform well for out-of-distribution text samples in the natural language processing domain [10]. We will investigate if this finding can be replicated for other types of formats, namely images, and if we can draw further conclusions on the transformer properties.

Hypothesis 1: *Transformer should perform better than CNN on out-of-distribution images.*

We believe that due to their attention mechanism, transformers should be able to better generalize better. Indeed, with their attention mechanism, they are not limited to connection inside a kernel to extract a feature but can learn and create their own attention pattern. Furthermore, they should be able to better balance their attention when tested on new unseen data as they can focus their attention on previously learned part of the image

Hypothesis 2: *Increasing the number of backgrounds should increase out-of-distribution performance*

The reasoning behind this hypothesis is that with an increasing amount of background variability, the unique feature of the background becomes less important and more similar to noise. This will make the performance more similar to in-distribution. As generally, in-distribution accuracy is better than out-distribution, we should see an increase in performance.

Hypothesis 3: *Adding layer normalization should improve performance*

Overfitting is a fundamental problem in machine learning, especially for out-of-distribution, as our training and testing data are from two distinct distributions. We suppose that layer normalization, a form of regularisation, should improve performance as it would lead to a simpler model that can more easily generalize.

Hypothesis 4: *Adding positional encoding should improve performance for out-of distribution of MHA*

The MHA lacks some inductive prior of the CNN, adding positional encoding would allow MHA to use positional information of images to make a better prediction, as seen in the visual attention paper [2]. This extra data should help the model make better sense of locality information.

Hypothesis 5: *Deeper model should generalize better*

The network becomes more depth but not wide, it should be able to learn more abstract features and use those features for better generalization. Another advantage of deep networks is that they can learn features at various levels of abstraction. This should help the model have a better understanding of the task.

Hypothesis 6: *Increasing the number of heads should lead to overfitting and a low number of heads should decrease in performance*

With a low number of heads, we cannot reach the best possible accuracy for the model as the image cannot have multiple focal points. We expect this to lead to worst performance, indeed the number of heads is too low for the model to be able to execute the task properly. On the contrary, with a high number of heads, we risk overfitting as the network will be fitted too much on the training data. Indeed, we expect to see a concave graph when plotting the accuracy against the number of heads.

4 Experiment results

4.1 Baseline performance

In order to make sure that our results were statistically significant, we did ten runs of each model to compute the means and standard deviation of all of our experiments.

Furthermore, we use the Adam optimizer [11] to help with our gradient descent. For our classification task, we used cross-entropy as our loss metrics. We can verify our implementation by checking that the model learned to decrease this loss.

The model is trained on a predefined number of backgrounds per digit. We can then measure out-distribution by testing the accuracy of new digits with same distribution of backgrounds. To measure the out-of-distribution, we place the new digits on top of new unseen backgrounds. Coming back to our example of figure 6, the network would be trained with the digit "5" on top of a truck. We can then compute two metrics: in-distribution and out-of-distribution. The in-distribution is when new "5" handwritten digits are on top of trucks and out-of-distribution is when the "5" is on top of a random background.

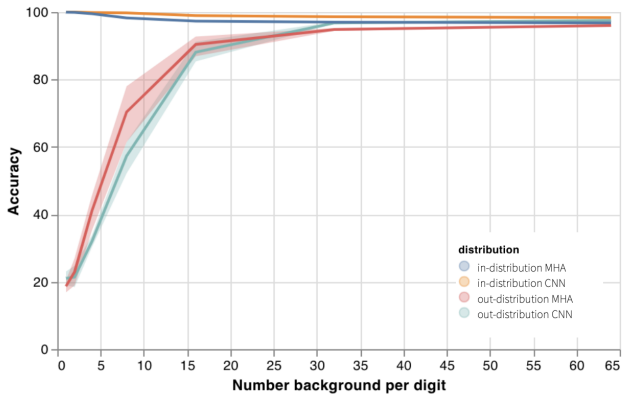

Figure 10: Baseline performance of CNN and MHA

Firstly, we defined a baseline performance for both models as seen in the figure 10. This figure shows the performance of both models for in-distribution and out-distribution while varying the number of backgrounds associated with each image. The figure plots the model's accuracy based on the number of backgrounds, from 1 to 64 backgrounds. Each digit is associated with a number of possible backgrounds that we vary during the experiment, this allows us to test the out-of-distribution performance. MHA performs significantly better for a low number of backgrounds than CNN, validating both of hypotheses 1 and 2.

We can see three notable results: good performance for MHA for low number of backgrounds, better absolute CNN performance in long tail, similar performance in the tail for in and out-distribution for each model,

Firstly, MHA performs relatively better than CNN for a low number of backgrounds. We can therefore accept our hypothesis 1, showing that the visual attention mechanism of the transformer helps us generalize better on out-of-distribution.

Secondly, in the long tail of the distribution the CNN performs better than the MHA. Indeed, this could be explained by the fact that CNNs have a lot of prior inductive making them better for image classification than MHA. We see that CNN for in-distribution performs significantly better than MHA. In fact, in the long tail the out-of-distribution nature of the test set is decreased because the number of associated backgrounds is high so it is more similar to an in-distribution with random backgrounds.

Thirdly, we also see that out-of-distribution and in-distribution reach similar accuracy in the tail of the graph. This is due to the fact that with many backgrounds, the output label and the backgrounds are not strongly correlated and so it is considered as noise by the network. The network will then only rely on the written digit to discriminate the image. This also us to accept our hypothesis 2 that increasing the number of background make the background less critical and more similar to random noise, so it is ignored. Indeed, we see better performance for high number of backgrounds.

4.2 Investigating transformer specific-components

The visual attention paper [2] added extra elements on top of the attention mechanism in an effort to increase the performance of the model. Thus, we investigated positional encoding, layer normalization, residual connection. Finally, we investigated the impact of the number of heads on the transformer performance.

Positional encoding

Positional encoding allows for the embedding of the information about the original location of a patch before feeding it to the MHA module, this allows the attention model to have access to the locality information in an image.

We used learnable weight to encode position as this has been shown to perform better than a simple coordinate-based system [2]. However, as seen in the appendix figure 14, the positional encoding did not lead to significant improvement in performance and on the contrary decreased slightly the model performance. This allowed us to reject our hypothesis 4.

Layer Normalisation

This layer is applied after MHA computation instead of directly using the output of the network, after doing the attention layer, we normalize.

Indeed this replicates the previous findings of [2], and we can see an increase in the performance as seen in appendix figure 16. This leads us to accept our hypothesis 3. Indeed, layer normalization is a regularization technique and thus having it helps with overfitting.

Residual

We further tested the influence of residuals on and off and found that they did not impact the performance in any significant way. This could have been due to the fact that the current network is quite shallow compared to many of the bigger networks where transformers are here. So skipping layers and propagating input further for gradient computation is not necessary.

We tried both with and without layer normalization as they are used in conjunction in the visual transformer architecture. As seen in appendix figure 15, we did not see any significant improvement compared to both the baseline and layer norm versions of our network and on the contrary saw a tendency for decreased performance with this enabled.

Number of heads

Lastly, the number of heads of the multi-head attention is the only hyperparameter unique to the MHA compared to CNN. Therefore we investigated the impact of a varying number of heads. To analyze the performance, we fixed the number of backgrounds to 8, a point of interest in the original baseline, showing a significant discrepancy between MHA and CNN performance. Furthermore, we decide to reuse the original architecture LeNet of figure 5. We then used the 16 out features for both MHA layers and try values until the number of heads is equal to the number of out-channels as this is the upper limit.

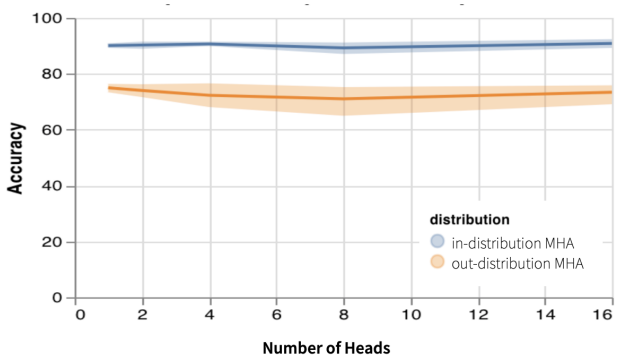

Figure 11: Accuracy of MHA based on the number of heads

As seen in figure 11, the number of heads does not significantly impact the network's performance, which rejected our hypothesis 6. The relative simplicity of our task could explain this. Certainly, MNIST classification is a simple computer vision task, and the input size is relatively small 32 by 32 pixels. This contrast with bigger transformer models such as GPT-3 that translate entire document where multiple heads could have more impact.

However, a smaller amount of heads is desirable, as it reduces computation time to a minimum. Indeed the training was longer with a higher number of heads.

4.3 Max-pooling role in both architecture

We investigated the effect of max-pooling on model performance. By observing the CNN and MHA performance in figure 13 in the appendix, we can see that removing max-pooling significantly decreases the performance of both models, therefore it can be considered a necessary component.

We can see that our experiment confirms common knowledge around max-pooling, as indicated by Andrew-Ng, when he said: "I have to admit, I think the main reason people use max-pooling because it has been found in a lot of experiments to work well... I don't know of anyone fully knows if that is the real underlying reason." [12]

Indeed Max-pooling helps us create small translation and rotation invariance in the images, as neighboring pixels are aggregated together. Another benefit of max-pooling, is that it reduces computation time as it shrinks the input.

4.4 Network depth and out-of-distribution performance

We further tested if deeper networks could improve out-of-distribution performance. We investigated the model's performance when the number of backgrounds is 8, as done with the number of heads. With this number of backgrounds, we could still see a difference between in and out performance and discrepancy between both models. We then plotted both the in and out distribution to see how they perform. Furthermore, we removed the pooling layer from the input this allowed us to test a higher number of layers as the input size is not quickly compressed to a smaller dimension.

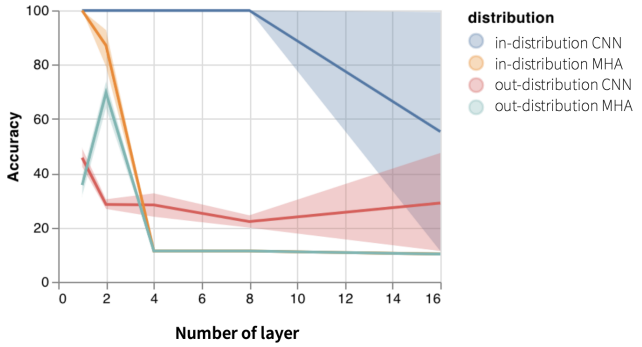

Figure 12: Accuracy of both models according to network depth

The experiment of figure 12 showed that deep networks did not perform well on our task. Indeed the main limitation is the lack of training time and computation. For MHA, if the number of layers is higher than 2, the performance collapses and becomes the same as random guessing, 10% accuracy. Indeed, we could not decrease the loss with such a setup by using 100 epochs and training time of multiple hours. This could be due to vanishing gradient problem as we are not using residual connections. The CNN showed better performance due to being more efficient in computing. Indeed, the performance only started to collapse at 8 layers for CNN compared to 2 in the MHA.

This brings to show that one of the main limitations of MHA is that they require a high amount of training and data. Indeed, they cannot use simple assumptions baked in the model but need to learn them manually. It is important to give a word of caution about this result as our graph shows that there is a learning problem with our models. By stabilizing learning, with regularisation, tuning of the learning step, or residual connections, we could increase the number of layers before the models collapse and see clearer trends and draw stronger conclusions from our experiment. This leads us to mark hypothesis 5 as inconclusive.

5 Responsible Research

This paper is purely algorithmic and theoretical, as it offers a performance improvement compared to previous findings and so does not extend the possible ethical risk of AI [13]. Indeed, artificial intelligence systems can offer a risk in some social aspects, such as the enforcement of social bias. However, we

are not dealing with the collection or usage of sensible data, only objects and numbers so there is no risk of a possible violation.

Furthermore, there is a reproducibility crisis in the scientific community, this is why the source code is fully available on GitHub (cf. appendix 7), with documentation. This allows for more accessible research and reproduction of the work. Indeed, the entire code is platform agnostic. It is based on a Conda environment that uses packages with fixed releases, so future updates to packages will not cause any issues for running our code in the future. The work is under MIT's open license, which allows the work to be fully modified. The original paper is also associated with a release version on GitHub to ensure stability and transparent modification to the original codebase.

Nevertheless, this project aimed to adhere to the CUDO (Communism, Universalism, Disinterestedness, Organized Skepticism) principles [14] and the Netherlands Code of Conduct for Research Integrity [15].

6 Related Work

6.1 Image to patch compared to unfold

For the multi-head attention to work, it was important to embed our image into a sequence of tokens. We tried to apply the image patching method from the ViT's paper [2] and unfolding. Indeed unfolding offers closer computation to CNN than classical image patching. Others tried to apply unfolding to transformer [16], but not to create a module equivalent to CNN 8. Our approach allowed the CNN module to be fully exchanged so that the benefit of both architectures could be compared on an equal footing.

6.2 Out-of-distribution for CNN and transformer

We investigated the performance of CNN on out-of-distribution. We focused on LeNet architecture because it is a simple and classical architecture, and we were able to replicate previous findings about performance. Secondly, we swapped the convolution for the multi-head attention module. Earlier works have found that transformers are better for out-of-distribution on text [17] however, not much research has been done on images. Furthermore, as shown in this paper, the comparison between CNN and transformer performance is not often on an equal footing [9]. However, our approach allows them to be compared fairly, and we have demonstrated that the attention mechanism improves out-of-distribution generalization.

6.3 Positional encoding, Residual connection, Layer norm

We implemented the methods proposed by ViT's paper [2] for positional encoding, Residual connection, and Layer norm. This replication exercise is also used to see the impact of each element individually. Indeed, we compared each element's influence separately to determine the necessity of the components. We were unable to replicate all the findings of the ViT's paper, namely for positional encoding, as we did not see an increase in performance.

6.4 Max-pooling

We compared the impact of max-pooling on both the CNN and MHA architecture. Literature, as well as previous research [18] has shown that max-pooling can help create invariants and improve performance. However, most of the literature on max-pooling is based on CNNs and less research has been conducted on transformers. In this paper, we have shown that transformers can also benefit from max-pooling and that it does offer similar benefits as in CNNs.

6.5 Depth network

We further compared the impact of an increased number of layers on performance. Most models are bigger than ever, showing that depth networks can learn more complex world representations. However, our task is quite simple, and increasing the number of layers does not only increase computation time but decreases performance. Showing that for simple tasks, shallow network can be more advantageous, contributing to the idea of Lei Jimmy Ba et al. [19] that larger networks are not necessarily better. Furthermore, our experiments show that it is easier to train a deep CNN compared to a model with multi-layers of MHA, making a shallower network more important for transformers.

6.6 Number of heads

We decide to investigate the number of heads to determine if changes in this hyperparameter lead to performance changes in the model. Indeed previous research [20] has found that too many heads can lead to overfitting. Overfitting is a critical problem for our experiment as we are dealing with out-of-distribution test data. Furthermore, Paul Michel et al. [21] showed that many heads can be pruned without impacting the performance. We were able to replicate the findings of the second paper with images instead of text. We also show that for simple images (32x32) overfitting with the number of heads is less of a concern.

7 Conclusion

This work reproduced previous findings on the robustness of transformers, namely on out-of-distribution images. We have shown that transformers perform significantly better than CNNs for out-of-distribution images. We also analyze which aspect of the transformer architecture contributes to this ability and see that the attention mechanism is primarily responsible for this. Finally, we determined that deeper transformers are not necessarily better for simple data and that a high number of heads is only essential for more complex data.

More research into out-of-distribution on more complex data, namely bigger images, could improve the reliability of our findings. We also think that further investigation on network depth could be done by stabilizing the learning and researching the impact of residual connection on network depth performance.

References

[1] A. Sarraf, M. Azhdari, and S. Sarraf, "A Comprehensive Review of Deep Learning Architectures for Computer Vision Applications," *Technology, and Sciences (ASRJETS) American Scientific Research Journal for Engineering*, vol. 77, no. 1, 2021, ISSN: 2313-4402.

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," Oct. 2020. [Online]. Available: http://arxiv.org/abs/2010.11929.

[3] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 2017-December, 2017.

[4] Y. LeCun, B. Boser, J. S. Denker, *et al.*, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, no. 4, 1989, ISSN: 0899-7667. DOI: 10.1162/neco.1989.1.4.541.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, 2017, ISSN: 15577317. DOI: 10.1145/3065386.

[6] IBM Cloud Education, *What are Convolutional Neural Networks?* Oct. 20. [Online]. Available: https://www.ibm.com/cloud/learn/convolutional-neural-networks.

[7] iacob, *Unfold operation diagram*, Jun. 2021.

[8] J. L. Ba, J. R. Kiros, and G. E. Hinton, "(LN) Layer Norm," *arXiv:1607.06450v1*, 2015.

[9] Yutong Bai, Jieru Mei, Alan Yuille, and Cihang Xie, "Are Transformers More Robust Than CNNs?," Nov. 2021.

[10] D. Hendrycks, X. Liu, E. Wallace, A. Dziedzic, R. Krishnan, and D. Song, "Pretrained Transformers Improve Out-of-Distribution Robustness," 2020. DOI: 10.18653/v1/2020.acl-main.244.

[11] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.

[12] Y. X. C. (https://stats.stackexchange.com/users/225473/yixiang-chong), *Why is max pooling necessary in convolutional neural networks?* Cross Validated. [Online]. Available: https://stats.stackexchange.com/q/374824.

[13] B. C. Stahl, "Ethical Issues of AI," in 2021. DOI: 10.1007/978-3-030-69978-9{_}4.

[14] M. D. Lund, "Understanding philosophy of science; Theory and reality: An introduction to the philosophy of science," *Journal of the History of the Behavioral Sciences*, vol. 41, no. 3, 2005, ISSN: 0022-5061. DOI: 10.1002/jhbs.20091.

[15] VSNU, KNAW, NFU, NWO, TO2-federatie, and Vereniging Hogescholen, "Netherlands code of conduct for research integrity 2018," Tech. Rep., 2018.

[16] P. Zhang, X. Dai, J. Yang, *et al.*, "Multi-Scale Vision Longformer: A New Vision Transformer for High-Resolution Image Encoding," in *Proceedings of the IEEE International Conference on Computer Vision*, 2021. DOI: 10.1109/ICCV48922.2021.00299.

[17] Chongzhi Zhang, Mingyuan Zhang, and Shanghang Zhang, "Delving Deep into the Generalization of Vision Transformers under Distribution Shifts," Jun. 2021.

[18] C. A. Goodfellow lan Bengio Yoshua, *Deep Learning - Ian Goodfellow, Yoshua Bengio, Aaron Courville*, 2016. [Online]. Available: https://www.deeplearningbook.org/.

[19] L. J. Ba and R. Caruana, "Do deep nets really need to be deep?" In *Advances in Neural Information Processing Systems*, vol. 3, 2014.

[20] Lu Liu, William Hamilton, Guodong Long, Jing Jiang, and Hugo Larochelle, "A Universal Representation Transformer Layer for Few-Shot Image Classification," Jun. 2020.

[21] P. Michel, O. Levy, and G. Neubig, "Are sixteen heads really better than one?" In *Advances in Neural Information Processing Systems*, vol. 32, 2019.

A Appendix

Github Repository Hyperlink.

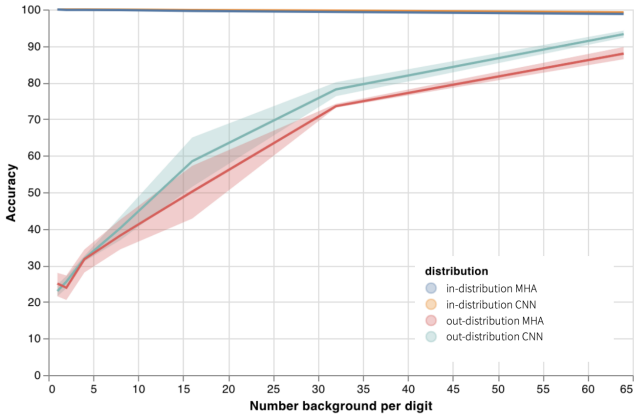

Figure 13: CNN and MHA performance without max-pooling

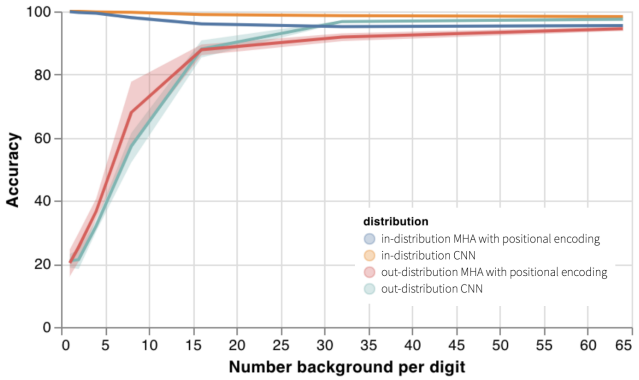

Figure 14: MHA performance with positional encoding

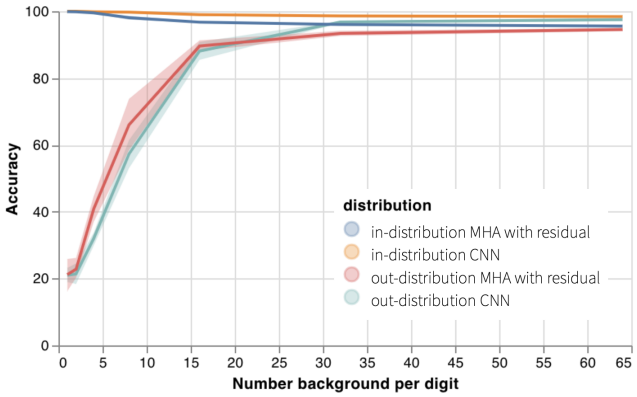

Figure 15: MHA performance with residual connection

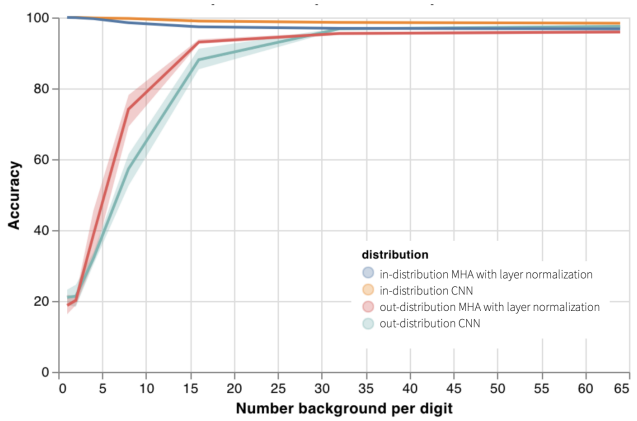

Figure 16: MHA performance with layer norm