Generalization and Data Transformation Invariance of Visual Attention Models

Pepijn de Kruijff
Supervisor(s): Wendelin Böhmer
EEMCS, Delft University of Technology, The Netherlands

24-6-2022

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

Abstract

This paper compares the generalizing capability of multi-head attention (MHA) models with that of convolutional neural networks (CNNs). This is done by comparing their performance on out-of-distribution data. The dataset that is used to train both models is created by coupling digits from the MNIST dataset with a set amount of background images from the CIFAR-10 dataset. An out of distribution sample is generated by using a background not used during training. This paper compares the accuracy of both models on such out-of-distribution samples to indicate the generalizability of both models. Furthermore, the invariance of MHA models towards certain affine data transformations is compared to that of CNNs. The results indicate that MHAs might be slightly better at generalizing to unseen data, but that CNNs are better able to generalize to the data transformations performed in this papers experiments.

1 Introduction

Image recognition is a subcategory in artificial intelligence that handles the interpretation of images. The challenge is to have the neural network 'understand' images so it can perform tasks like classification, tagging, or detecting objects in an image. It is currently a growing field with major applications, from robotics to self-driving vehicles.

One of the classic tasks in computer vision is the classification of the MNIST dataset, which contains grayscale images of handwritten digits and their correct labels (figure 1). Since its introduction by LeCun et al, the dataset has been used as a benchmark for new machine learning models. Early recognition techniques used by LeCun et al had error rates of 7 to 12 percent, with CNNs performing the best. This error rate has steadily decreased over the past 30 years, with the most recent CNN models able to achieve an accuracy of 99.91 percent [1].

Figure 1: Four MNIST digits

However, a challenge that remains is that these models lack generalization capability. Even though they can classify handwritten digits from the MNIST dataset with high accuracy, when they are presented with an image that is unlike their training data it is not able to generalize what it has learned and apply it to this new input. This makes the real-life application of these models hard because it would require them to experience all possible situations during training, which is not realistic. To improve the generalizability of these models multiple methods have been proposed. This paper discusses one of these methods, using self-attention via a multi-head attention layer in a network.

Multi-head attention layers are part of transformers, a model first created for Natural Language Processing (NLP) tasks [10]. Current methods like recurrent neural networks (RNNs) process words sequentially. They are limited by their finite memory or words they have previously seen. Attention attempts to solve this by processing all words simultaneously and drawing a connection between parts in the sentences indicating their importance. This is a drastic difference from previous methods, and led to successful models like Googles BERT model [2].

This mechanism is also applicable to classifying images by splitting them into patches that can be fed to the network similar to words in an NLP context [3]. The attention mechanism then computes how much attention should be applied to specific patches. Because the attention mechanism has multiple heads, the attention is split between multiple parts of the image. This could allow the model to generalize better, as it can shift its focus in an out-of-distribution input. It is currently unclear how such an MHA layer compares to a conventional convolutional neural network on out-of-distribution data.

Aside from accuracy on out-of-distribution data, it also remains unresearched whether MHAs and CNNs share the same data level of transformation invariance. Having a network be invariant to data transformations like translation, rotation or scaling is a desired feature in image recognition because it allows objects to be recognized regardless of their position in the input data. This allows the network to learn features wherever they appear, and generalize them to other positions, rotations, or scales. Current CNNs can be designed to be translation invariant, but they lack other types of transformation invariance.

This paper's contribution to current research into MHA models is two-fold. The first goal is to test a CNN and MHA model on in- and out-of-distribution data to test the hypothesis that MHAs are better able to generalize on out-of-distribution data. The second part is to compare how both models compare on transformed data inputs.

2 Background

This section provides a brief background on Convolutional Neural Networks and Multi-Head Attention layers

2.1 Convolutional Neural Networks

Functionality

At the basis of the Convolutional Neural Network [7] is the mathematical operation "convolution". The arguments of this operation are a function usually referred to as the **input**, the second as **kernel** and the output a **feature map**. As defined in the book "Deep Learning" [5] the equation below is the convolution operation.

$$s(t) = \int x(a)w(t-a)da \tag{1}$$

This operation represents a weighted average of the values of the function x, giving us a more smoothed estimate of the results of x. Though the function w can be replaced with any function, this is the most common operation for machine learning contexts. Additionally, we will usually be working in the discrete case, where our kernel and input are tensors, so multidimensional arrays. The equation for the discrete case is:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a)da \tag{2}$$

The asterisk is the symbol used to denote the convolution between two functions. The convolution will often be utilized with multiple axes at the same time, for example, a 2-dimensional image in our case and therefore also a 2-dimensional kernel. Equation 3 below allows for this and is called “cross-correlation”. Even though in machine learning contexts this term is often interchanged with the term convolution. Figure 2 displays how this operation is applied to an input matrix to generate the output.

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n) \tag{3}$$

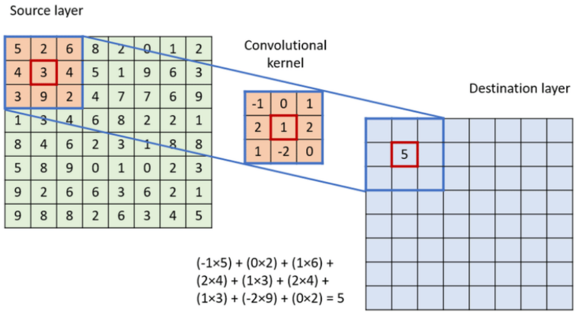

Figure 2: Visual representation of convolution operation [8]

A CNN architecture typically consists of a convolution layer, described by equation 3, followed by a pooling layer. This pattern is repeated multiple times after which the result is flattened and processed by a couple of linear layers. A typical setup is shown in figure 3

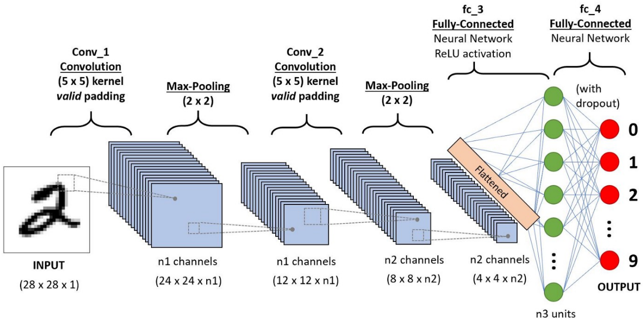

Figure 3: Example CNN [9]

Sparse interactions

When working on images our input will often be composed of many thousands of pixels. Feeding it to a traditional 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, so having thousands of weights set to 0 is very inefficient. A CNN solves this problem by only learning a kernel, and shifting this kernel over the input. This will result in a significant decrease in required parameters.

Translation invariance

One important property of CNNs is that they can be invariant to translation. This means that translating the image does not affect its ability to recognize patterns. There are many hypotheses about the source of this invariance, including the pooling layer and increasing receptive fields in convolution layers [6].The degree of invariance differs based on model architecture and training data. This property is important because this paper will investigate this ability further and compare it with that of MHAs.

2.2 Multi-Head Attention

The visual transformer [3][11] is an extension of the paper [10]. We first start by explaining the ideas from the original paper and see how those can be applied to the field of computer vision.

Transformers were created for NLP tasks. the problem with the previously used methods such as recurrent neural network RNN is their limited memory. RNN process each word sequentially and cannot use the information of all the previously seen or future words efficiently.

Attention solves this problem by, instead of relying on a sequence of processing, drawing a connection between different words in sentences. To do this, for each word i there are 3 components: encoding, the query (q_i), the key (k_i), and the value (v_i). These components are calculated by passing the vector representation of the input word through a linear layer, of which the weights are computed during training:

$$q_i = \Theta_Q x_i, k_i = \Theta_K x_i, v_i = \Theta_V x_i, \tag{4}$$

The importance between word i and another word is calculated by taking the dot product between its query and the other words' key. This output vector is then scaled and normalized, resulting in an attention score for each input word:

$$w = softmax(\frac{q_i^T \cdot k_j}{\sqrt{d_k}}) \tag{5}$$

where d_k is the size of the linear projections q_i and k_i. Softmax is equal to:

$$softmax(\vec{x}) = \frac{e^{\vec{x}}}{\sum_{i=1}^{n} e^{x_i}} \tag{6}$$

To illustrate, consider the following example. For the input sentence 'I live in Delft'. For each word the query, key and value is calculated using equation 4. Inserting the key representation of word 'Delft' and key representation of word 'live' in equation 5, would result in a high value, indicating

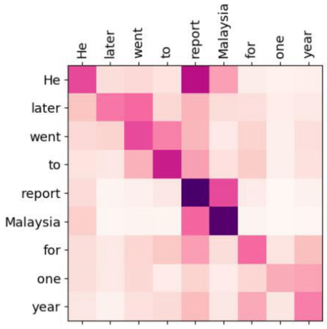

Figure 4: Attention Matrix [4]

that there is a strong connection between the word live and itself and Delft. Inserting 'I' as the key would result in a lower value because it is less important to interpret the word Delft and to understand structure of the sentence.

This can be done for all queries and keys by matrix multiplication using the following formula:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{7}$$

where Q, K, and V are the matrices containing all queries, keys, and values for all n input patches:

$$Q = [q_1, .., q_n], K = [k_1, .., k_n], V = [v_1, .., v_n] \tag{8}$$

The result of this operation is an attention matrix, of which an example is shown in figure 4. High values in this matrix correspond to a high degree of attention between the words. This matrix is then multiplied by the value component. This means that, unlike the CNN, the MHA layer can use information from other patches.

Multi-Head Attention

Multi-head attention is used to be able to pay attention to multiple parts of the sentence at once. To do this, all attention vectors are concatenated together:

$$MultiHead(Q, K, V) = Concat(Head1, ...head_h)W^O \tag{9}$$

where

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{10}$$

With dimensions: $W_i^Q \in R^{d_{model} \times d_k}, W_i^K \in R^{d_{model} \times d_k}, W_i^V \in R^{d_{model} \times d_v}$

Multi-head attention also has the extra advantage that it can enable better parallelization. "multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. With a single attention head, averaging inhibits this." [10]. The full MHA layer is visualized in figure 5.

Finally, in the context of this paper, the objective is to apply transformers to images. This can be done by unfolding the image into a sequence of flattened patches, which are fed to the attention mechanism [3]. This is visualized in the lower part of figure 6. Those patches are similar to the words used in NLP. This allows the network to compute the attention that we should apply to each in addition to the link between the different patches. Multi-head attention means that we can focus on multiple parts of the image at once. In sharp contrast to CNNs, MHAs can use information from completely different parts of an image due to their attention mechanism.

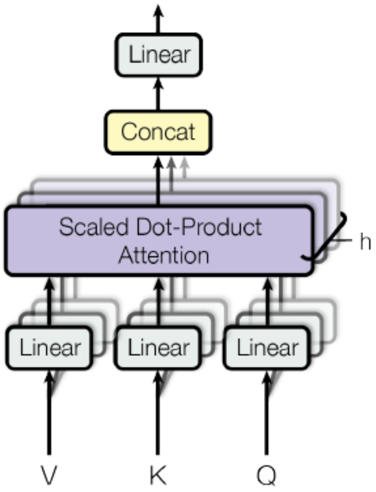

Figure 5: Multi-head attention model [3]

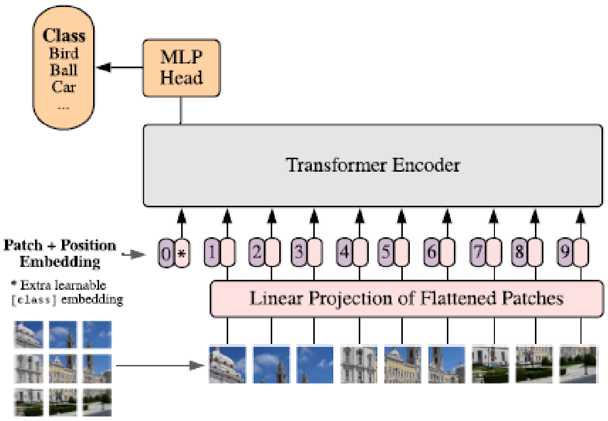

Figure 6: Visual transformer model []

3 Methodology

In this section, the methodology of the techniques used in this paper is explained. The first section goes into detail about the datasets used for the general comparison between CNNs and MHAs. The second section will describe the general model using convolutional layers. The third section describes how the MHA model is implemented. The final subsection introduces the affine data transformations applied to the input data.

3.1 Dataset

To compare whether MHA models can generalize better than CNN's, we construct a dataset where we can generate an out-of-distribution sample to feed both models. This dataset is constructed by overlaying CIFAR-10 images with an MNIST digit, as seen in figure 7. Consider the following range of options in which the MNIST dataset could be combined with backgrounds before training the model on it:

1. The same digits always have the same background.
2. Digits have a set of x amount of backgrounds associated with them.
3. All digits have a random background image.

In the first case, digits always have the same background image. For example, every digit '2' that occurs in the dataset would always have a airplane background and the digit '5' would always have the same image of a horse. In the last case there are no backgrounds assigned to digits. Every digit in the dataset has a random background. This papers model is trained multiple times using the second case. In this scenario, a digit has a variable amount of backgrounds. For example, if x is set to two backgrounds per digit, a '4' could be configured to appear in the dataset sometimes with an image of a boat and sometimes with an image of a lion. Both would appear in the dataset and the digit '4' would never have another image then these two.

The performance of the model is than compared by testing it on a out-of-distribution input. This means this input is a previously unseen digit with a unseen background. The performance of the model should be dependent on x (the amount of backgrounds per digit), because different values of x should lead the model to pay attention to different parts of the image. When x is high, the background is not a good predictor for the digit in the foreground. This would lead the model to primarily train itself using the information on the foreground. Therefore, when an out-of-distribution input with an unseen background is presented to a model trained by a dataset with a high x value, it should be able to correctly classify the digit. The other extreme is when x equals 1, so all background images correspond to a digit. In this case, the model might have trained itself to look at the background to classify the digits in the foreground because background images always correspond. This would lead the model would not performing well on out-of-distribution data. While x is increased, the model should be trained in a way that increasingly shifts the focus from the background to the foreground, and therefore has to generalize less when presented with an out of distribution image.

The main research question of this paper, whether MHAs perform are better able to generalize then CNNs can be tested using these various methods of combining the dataset. By training the model for different values of x, we have models that have their attention on different part of the image, either the irrelevant background or the digit on the foreground. By then presenting images with unseen backgrounds, we are able to measure their ability to shift attention to the digit. This ability corresponds to their generalizability, as they are able to focus on important parts of an image and ignore unseen ones. This papers hypothesis about the accuracy of both models under these circuimstances are as follows:

> ***Hypothesis 1:*** *MHAs have higher classification accuracies than CNNs for lower amounts of backgrounds per digit*
>
> ***Hypothesis 2:*** *CNNs have higher classification accuracies than MHAs for higher amounts of images per digit*

The first hypothesis comes from the hypothesis that MHAs are better able to generalize to unseen data because it can pay attention to multiple parts of the input. This generalizing ability would be most visible for lower x values, because in this scenario the CNN would focus on the background while the MHA would be able to shift its attention on an out-of-distribution image. In a scenario where the model has been trained with a lot of different backgrounds per digit, both the CNN and the MHA only pay attention to the foreground because no information about the foreground can be deduced from the background during training. Therefore, no 'shift' of attention to the foreground is needed by the CNN. In this setting, the CNN might outperform the MHA because the lower amount of parameters of CNNs makes them less susceptible to overfitting. This is why the second hypothesis predicts better performance for the CNN for higher values of x.

t

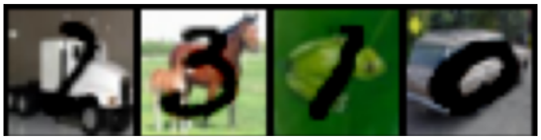

Figure 7: MNIST digit with CIFAR-10 background

3.2 CNN architecture

The CNN model is relatively simple consisting of the following layers: a common CNN combination of a convolutional layer followed by a rectified linear unit (ReLu) and a pool layer is repeated two times. The output is then flattened and used as input to three fully connected layers coupled with ReLus. The convolutional layers use a kernel size of 5. The pooling layer is a 2x2 kernel selecting the maximum of the values in the kernel.

The first convolutional layer has three input channels and outputs 6 channels. It takes a batch of 32x32 tensor with three channels, and outputs a 6 channel 28x28 tensor. This is halved by the pooling layer and then used as input to the second convolution layer. This convolution layer outputs a 10x10 tensor with 16 channels. This is pooled to a 5x5x16 tensor. To insert this in the fully connected layers, this is flattened to 400 values. This is reduced in width by the fully connected layers first to 120 and then to a final output of 10, corresponding to the 10 possible classifications of the MNIST dataset.

3.3 MHA architecture

The multihead attention model keeps the exact same structure as the CNN model but replaces the convolution layers with

MHA layers. This similarity between models is maintained to make sure that the performance between the convolution layer and MHA layer is compared, and not influenced by other model parameters. For the MHA layers, the standard MHA layers from the PyTorch library are used. To ensure that the output dimensions match those of the CNN, the dimensions of the keys, queries and values is configured as 6. Unlike [11], the patches are not generated so that they do not overlap. Instead, the original is unfolded in the same way CNNs are, with a stride of 1. This unfolding procedure generates 784 flattened patches of size 75, equal to the amount the content of a kernel times the amount of input channels. These patches are then multiplied by the key, query and value weights, as described in section 2.2. The output of the MHA layer has 6 channels and is reshaped to two-dimensional patches. Similar to the CNN a ReLu and max pool is then applied, halving the height and width from 28x28 to 14x14. This process is then repeated, unfolding the result into patches to feed into the next MHA layer. This layer has a dimension of 16, equal to the CNN. The result of this second MHA layer is then again pooled and flattened, after which it is used as input to three fully connected layers.

3.4 Data transformations

The performance of CNNs and MHAs of three types of transformations are tested. These transformations are:

1. Translation
2. Rotation
3. Scaling

These are the translations that often occur in real-world image recognition tasks. CNNs are, dependent on the architecture, invariant due to translation when combined with pooling layers. They are not naturally invariant to scaling or rotating, although this can be somewhat learned using data augmentation techniques. The MHA layer should be invariant to translation too, meaning that an architecture that is invariant to translation with a convolution layer should be invariant with a MHA layer too.

Which architecture might perform better under data transformations is debatable. A possibility is that MHAs are better able to do this, because the they are not, like CNNs, scarcely connected. This allows them to construct feature maps in which a feature is not restricted by the exact location of a input. If this is sufficiently thought to the model during training by natural variations, it may allow the MHA to generalize better. That is why the following hypothesis are constructed.

> ***Hypothesis 3:*** *MHAs are more invariant to translations then CNNs*
>
> ***Hypothesis 4:*** *MHAs are more invariant to rotations then CNNs*
>
> ***Hypothesis 5:*** *MHAs are more invariant to scaling then CNNs*

These hypothesis will be verified in the experiments. The next section details how these experiments are conducted

4 Experimental setup

The goal of the experiments is to verify the hypotheses set out in section 3.1 and 3.3. These hypotheses fall into two categories. Hypothesis 1 and 2 predict the relative performance of MHAs and CNNs for a varying number of backgrounds in the training set and therefore indicate the generalizability of both models under varying circumstances. The second set of hypotheses is about the generalizability of data transformations. This section discusses how the experiments are set up to answer these two sets of research questions. The first subsection details the hyperparameters both models use. The second subsection is about how the comparison is made between the generalizing capability of both models by comparing their performance on out-of-distribution data. The third subsection explains how this experiment is extended to data transformations.

4.1 Hyperparameters

To be able to compare the models they are replicated multiple times with the same parameters and data from equal distributions. The experiments are performed 10 times on both models to ensure that the results are statistically significant. Ideally, the experiments would be run more often, but due to restricted computing power, this was not possible. A batch size of 16 with a learning rate of 0.001 is used.

4.2 Comparison models

Per replication, the model is trained with a different dataset. Each dataset has a different amount of CIFAR-10 images per digit. To visualize their relative performance, the average accuracy over all replications is graphed per x, where x is the number of backgrounds per digit. The x data points are powers from 2 up to 64, which should be a sufficient amount to have both models pay their full attention to the foreground.

The accuracy of both the training set and the in-distribution and out-distribution test sets is visualized in a graph. The training set is the same set used for training the model. For the in-distribution set, the digits are never seen during training but the same mapping to background images is used. For the out-of-distribution set, both the digit and the background are not seen during training.

Visualizing the experiments in this manner allows us to answer hypothesis 1 and 2. The relative performance for high x values corresponds to the 'baseline' performance, where not much 'shifting' of attention is required because both models have been trained to look at the foreground. The relative performance for lower x values represents the performance where this is required because it has been trained to look at the background too. The hypothesis can be answered by comparing the difference in accuracy for low and high values of x.

4.3 Data transformations

This subsection is about the experimental setup for hypothesis 3, 4 and 5.

To measure how both models perform on different data transformations, new data sets are classified on the trained model. These test sets have a previously unseen background, to prevent the model from classifying on the background alone. This means that these experiments will measure the generalizability of both networks, as well as their invariance to data transformations.

The three transformations described in section 3.4 are applied to digits. Note that the transformations are only applied to the digit, and not the background. The translation is applied by moving the digit a certain amount of pixels. Every digit is translated in a random direction. The rotation is a rotation either clockwise or counterclockwise. In the results, only positive rotation values are used, which is the average of the clockwise and counterclockwise results. Finally, the scaling corresponds to a scaling factor. The digits are only scaled down.

Output

The output is visualized in two ways. In the first graph, a comparison of all data transformations is given for varying levels of backgrounds per digit. This graph illustrates the capacity of all models trained on different datasets, to generalize after a certain transformation. Comparing the difference between the accuracies for both models should give an indication of the impact these translations have on this capability and whether this differs for different types of transformations

Another set of graphs will illustrate how well both these models can generalize to transformed digits under different amounts of transformations. For example, the accuracy for both models for different rotations is visualized, from 0 to 50 degrees. For translation, the effect of translating further away up to 5 pixels is compared. Scaling compares how both models perform from a scale factor of 1 to a final scale of 0.5. These experiments are done with a set amount of backgrounds per digit. To still be able to compare two models in both a scenario where to model is trained to pay attention to the foreground with a model that also looks to the background and has to shift its attention more, the results include a line for both models trained for both for x = 8 and x = 64 backgrounds per digit.

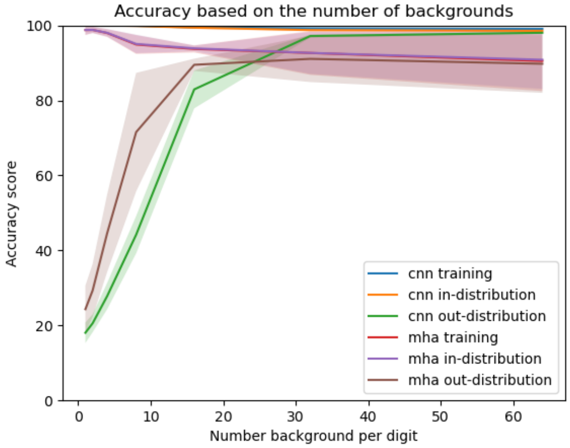

Figure 8: Accuracy on different test-sets for CNNs and MHAs

5 Results

This section is split in two, for the two research questions to be answered. The first section compares the the CNN with the MHA model for a varying amount of backgrounds per digit. The second section compares the effect data transformations have on the performance of both models.

5.1 Comparison tranformations

This section compares the performance of the CNN and MHA model on training, in-distribution, and out-distribution data. The result is visible in figure 8. The amount of backgrounds per digit is represented on the x-axis. The lines correspond with the average of the accuracies over all the replications. For both models, the accuracy of the model classifying the digit on an in-distribution and an out-distribution set is graphed.

Hypothesis 1 is that MHA models would be able to generalize could be shown by the model performing better on out-of-distribution data when the data is configured with a low amount of background images per digit. Figure 8 shows that this performance is higher for a lower value value of x. For 8 backgrounds per digit does the MHA classify the digits significantly better than the CNN. This verifies hypothesis 1. For higher x values, the CNN performs better than the MHA. This is explained by the attention mechanism providing no real benefit because due to the high amount of different backgrounds both the CNN and the MHA model have trained to pay attention to the foreground only. However, the chart shows that these results are very uncertain for higher values of x. The result is not statistically significant for this amount of replications. For this reason the second hypothesis is likely, but cannot be verified with the results from this experiment.

5.2 Effect data transformations

This section describes the impact of data transformations on the performance of both models. The first subsection compares different transformations to provide an indication of the performance of MHAs and CCNs for a certain amount of background. The second subsection investigates the invariance for both models for multiple degrees of transformations.

Comparison data transformations

In figure 9 the accuracy of both models for varying amounts of backgrounds per digit is given, for the three given transformations. The top two lines provide the baseline out-of-distribution test set. The three other lines represent a rotation of the digit by 20 degrees, a scaling of the digit with a scale factor of .75, and a translation in a random direction of 4 pixels.

The results in figure 9 show very similar accuracies for both models after the data transformation. The CNN outperforms the MHA significantly in generalizing to unseen data transformations. The relative performance for both models follows the same pattern for all data transformations: the MHA model performs relatively well for lower x values and the CNN performs better for higher x values. This is in line with this paper's hypothesis of MHA's being able to generalize better, as described in section 5.1. The different data transformations result in a more slightly diverging performance

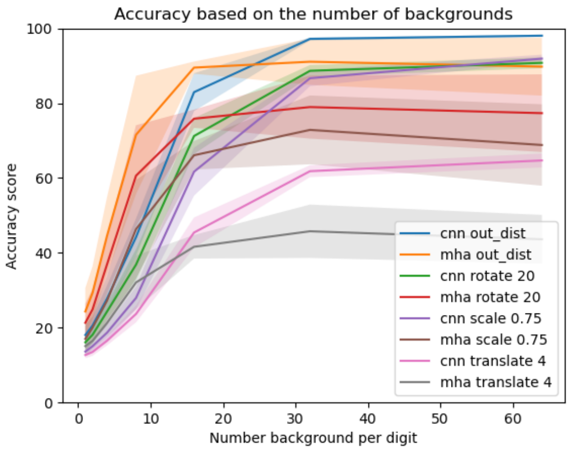

Figure 9: Accuracy CNNs MHAs after transformed data input

between the MHA and CNN for higher values of x. This is most visible for the scaling transformation. This is investigated further in the next section.

Effect severity transformation

This section compares the effect of specific types of transformations for a model trained on a dataset where the digits have 8 or 64 backgrounds. Figure 10 shows the relative performance for translations for x amount of translation in any random direction. The graph shows two sets of lines, one comparing a CNN and MHA for 8 backgrounds per digit and one comparing these for 64 backgrounds per digit. This graph should not be interpreted by looking at the absolute accuracy of both models. Instead, the performance between the MHA and CNN should be compared for both sets of lines. Even though for $x = 64$ the CNN performs better and for $x = 8$ the opposite holds, the graph shows that both MHA lines decline significantly more rapidly for larger translations. Comparing the blue line to the orange line and the green line to the red line shows that the MHA is more affected by the translation than the CNN.

Remarkable is the bad performance of the CNN under translation. CNNs can be constructed to be translation invariant. However, probably because this architecture flattens the output maps and is processed by linear layers, the CNN architecture loses this ability. While the MHA should also be invariant to translations, it seems unable to generalize well under this architecture. Because the CNN outperforms the MHA, hypothesis 3 proves to be incorrect.

Figure 11 contains the relative performances for different rotations of the digit. This is the average of clockwise and counterclockwise rotations. This result is very similar to the results for translation. Again, the CNN is more invariant to rotations than the MHA. This is in not line with hypothesis 5.

Figure 12 shows the accuracy for the scaled digits. Both models do not have any natural invariance to scaling. While the resulting accuracies of both models have a high variance, the CNN seems significantly more invariant to scaling than the MHA for both lower and higher amounts of backgrounds.

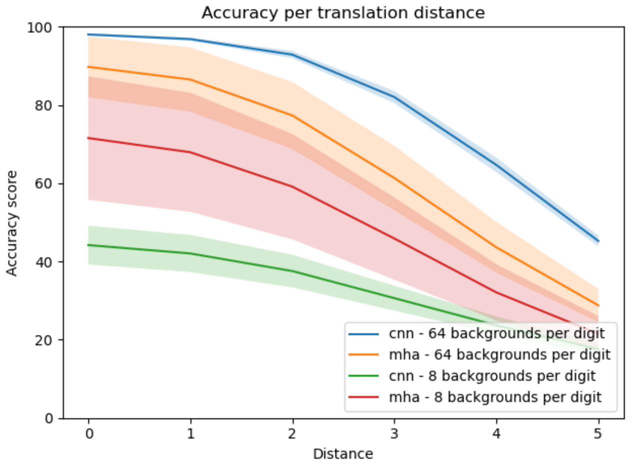

Figure 10: results accuracy for translation transformation

As mentioned in 5.2 The difference in performance between the models is the biggest for this transformation. Like the other transformation hypothesis, Hypothesis 5 is incorrect according to these results because the CNNs seem more resilient to rotation.

6 Discussion

The results comparing both models on an out-of-distribution test set show that the MHA performs slightly better in situations where the model is required to shift its attention more. This suggests that the attention mechanism is better able to generalize. The fact that the CNN performs better for higher amounts of backgrounds per digit suggests that they perform better in situations where this generalization is not needed. However, this assumption may be too big based on the results of this experiment, as many other factors have to be taken into account. For example, both architectures are not optimized for this task, to ensure that they are similar enough

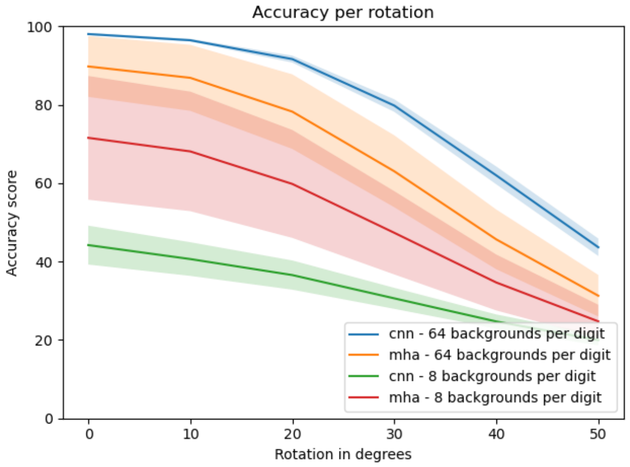

Figure 11: results accuracy for rotation transformation

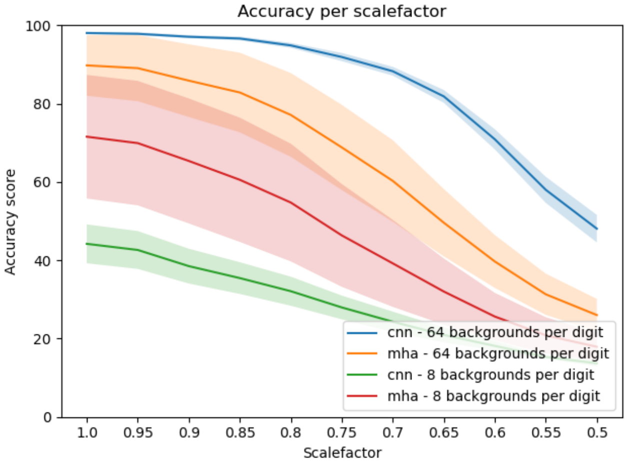

Figure 12: Accuracy CNNs and MHAs for different scale factors

to compare easily. It could be that a full visual transformer architecture is better able to make use of the attention mechanism and as a result have better generalizability or have a better baseline performance like the CNNs.

Furthermore, The results of the MHAs after the data transformations are surprisingly worse than those of CNNs. Contradictory to our hypothesis, the fact that they are not sparsely connected does not mean they perform better after slight data transformations. A possible explanation is that the MHAs did overfit to the data. No definitive explanation can be formulated as to what causes this performance difference. What is unfortunate is that this research was not properly able to research the translation invariance because the architecture was inherently not translation invariant due to the flattening and linear layers. Further research could investigate this invariance by designing an architecture that would be invariant. This could be done by implementing an aggregation function over the feature maps of the MHA.

Lastly, the results show a high standard deviation between the results of different replications of MHAs. This is in contrast to CNNs, which seem to have a consistent performance. It is unclear where this inconsistency in MHAs performance originates from. For further experiments on this matter, it is suggested that the model is first optimized so consistent results are achieved, before attempting any comparison to CNNs.

Further experiments are recommended for a definitive answer to this paper's research question. It could show for example, that an MHA is significantly better able to generalize if it has seen more variation during training. By training the data with slight data augmentations, this could be verified.

7 Responsible Research

This section reflects on the ethical aspects of this research. The first subsection discusses the reproducibility of the experiments and the usage of their data. The second subsection details the integrity of the author.

7.1 Experimental setup and results

This research and the structure of this paper have been structured in a way to be reproducible. For this reason, a standard implementation of an MHA is used, and big common open-source datasets have been used to test its accuracy. The CNN architecture is also easy to recreate. The code is also available on GitHub. This also includes the resulting data.

The experiments were replicated 10 times, to ensure that the results accurately mirror the average behavior of both models. Because the standard deviation in the results is still quite large for MHAs, further experiments could be conducted with an even larger amount of experiments. For this paper conclusions on relative accuracies of models are made not on statistical tests, but based on whether the standard deviations of results overlap.

Furthermore, giving as broad of a picture of the possible results of the experiments was a priority in this picture. There have not been any results with a contradictory conclusion to that that what has been presented in this paper. The data that has been presented in the paper has been selected because it shows a correct summary of all the data. This was a challenge, because the resulting data is for two models, for 7 different amounts of backgrounds per digit, for multiple types of translations, and for different severities of translations. An attempt was made to show the most relevant plots and explain why these plots are most relevant given the characteristics of both models and their apparent performance. For the sake of transparency, all data is made public online.

7.2 Scientific integrity

While this paper is structured according to a scientific paper, there is no intention of publication. The result of this is that, while its subject is not widely discussed in the literature, there is no pressure for the results to be significant. The author also does not have a special interest in either CNN's or MHA's, preventing any conflict of interest in comparing their performance.

8 Conclusion

The goal of this research was two-fold; First, to compare the generalizing capabilities of MHA and CCN models by classifying out-of-distribution data. Secondly, to investigate their invariance to certain data transformations applied to the input data. This has been researched by training both models on MNIST dataset with CIFAR-10 backgrounds, and testing them with an out-of distribution test set with unknown backgrounds. The experiments performed for this paper suggest that MHAs are better able to generalize in the context of this papers experiments. Futhermore, MHAs seem to perform slightly worse in scenario's where generalizing is not required. This result is not conclusive though, as the results from the MHA model are not consistent enough.

Finally, the results illustrated that MHAs do not perform well after affine data transformations have been applied to the model. This suggests they are not able to generalize to slightly modified data. However, this relation could be further investigated using models that are optimized for transformation invariance, which this model was not.

References

[1] Sanghyeon An, Minjun Lee, Sanglee Park, Heerin Yang, and Jungmin So. An ensemble of simple convolutional neural network models for mnist digit recognition, 08 2020.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.

[4] Siyuan Du and Hao Wang. Addressing syntax-based semantic complementation: Incorporating entity and soft dependency constraints into metonymy resolution. *Future Internet*, 14:85, 03 2022.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[6] Eric Kauderer-Abrams. Quantifying translation-invariance in convolutional neural networks. 12 2017.

[7] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[8] Damian Podareanu, Valeriu Codreanu, Sandra Aigner, Caspar Leeuwen, and Volker Weinberg. Best practice guide - deep learning, 02 2019.

[9] Sumit Saha. A comprehensive guide to convolutional neural networks-the eli5 way, Dec 2018.

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

[11] Yulin Wang, Rui Huang, Shiji Song, Zeyi Huang, and Gao Huang. Not all images are worth 16x16 words: Dynamic vision transformers with adaptive sequence length. 2021.