Seismic Data Compression using Convolutional Autoencoder

A report of Mini Project on Information Processing and Compression (EC761)
Submitted by

Rohan Jijju 181EC138
Roshan Rangarajan 181EC139

Under the guidance of
Dr. Aparna P.
in partial fulfilment of the requirements for the award of the degree of
BACHELOR OF TECHNOLOGY

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA
SURATHKAL, MANGALORE - 575025

Contents

1 Abstract

Seismic data is important for a variety of reasons. To address the exponential increase in seismic data, a variety of methods for seismic data compression have been created. In this work, we explore some of the different methods of seismic data compression. The models implemented in this paper include convolutional autoencoder models as well as discrete cosine transform(DCT) and discrete wavelet transform(DWT) models. Further, quantization techniques are used to create a model that gives much higher compression ratios as compared to the rest. All the models are compared on the Utah FORGE dataset and are quantitatively analyzed using the NMSE, NRMSE and SNR metrics.

2 Introduction

Seismic data refers to recordings of tremors from the earth's surface. This data can consist of either 1D time series integers or 2D/3D cross-sectional images. They have a variety of uses, but are mainly used for analyzing the patterns and occurrence of tsunamis and earthquakes. Seismic data is also used for things like checking the viability of oil and gas exploration.

Seismic data has been growing exponentially for the past few decades. Daily seismic surveys acquire petabytes of data. To handle this much information, there needs to be a method to store the data more effectively. As a result, data compression takes place on Seismic data prior to storage or transmission.

The goal of data compression is generally to achieve the greatest levels of performance as possible in terms of compression ratios. The higher the compression ratio, the less space the data will take up. However, it is also necessary for the reconstructed data from the compression to be of a high enough quality.

Seismic data compression are of two types- lossless and lossy. Lossless compression provides perfect reconstruction but gives low compression ratios. Traditional methods of lossy compression for seismic data include using DWT(Discrete Wavelet Transform) and other integer wavelet transforms. Models such as Adaptive linear filters can give lossless compression with compression ratios in the range of 8:1. The traditional method for lossy compression is to use some transformation on the input signal and to then quantize for compression.

For this paper, we have compared different lossy compression methods. Autoencoder models as well as discrete cosine transform(DCT) and discrete wavelet transform(DWT) models are implemented. Further, quantization techniques are used to create a model that gives much higher compression ratios as compared to the rest.

All the models are analyzed and compared using different metrics. The original signal and the reconstructed signal are compared based on the NMSE, NRMSE and SNR metrics.

3 Literature Survey

A number of transform methods exist for seismic data compression. Spanias et al. explore this in their paper[4]. They compared the DFT(Discrete Forier Transform), KLT(Karhunen-Loeve Transform), the DCT(Discrete Cosine Transform) and the WHT(Walsh-Hadamard Transform). The KLT was found to give best performance, but was found to be non-robust. They thus concluded that the DCT was best for this application.

Villasenor et al. [5] have proposed a method where they have used a wavelet-based algorithm that operates directly in the highest dimension of the data available. They have been able to successfully compress data with no observable loss of information at compression ratios substantially greater than 100:1.

A wavelet based approach is used by Zhenbing et al. in [6]. It is able to produce decent lossy compression at low CRs(compression ratios), but performs relatively poorly at high CRs.

Emad et al. have created a CAE(convolutional autoencoder) based model for seismic data compression[1]. They have provided a series of models that can be used for 1d signal encoding that provide decent compressions, that we will use for our model.

Further, adaptive linear prediction has been used by Wes McCoy et al in their paper [2]. They have implemented a lossless seismic data compression through three different algorithms :- normalize least-mean square (NLMS) algorithm, the gradient adaptive lattice (GAL) algorithm, and the recursive least squares lattice (RLSL) algorithm.

4 Background (Description of Dataset)

The dataset used for quantitative comparison and for training the machine learning model was the Utah FORGE dataset [3]. The dataset is made up of 2D and 3D seismic data from the Utah FORGE study area near Roosevelt Hot Springs. It consists of a set of 27,722 seismic traces. Each seismic trace consists of a 4000 samples 1d signal at a variety of amplitudes. For the purposes of this comparison, we have normalized it to the range of [0,1]. We have used 900 samples for training the machine learning model and have used 100 samples for quantitative analysis.

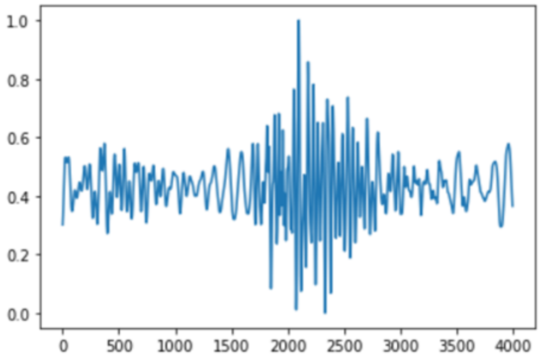

Figure 1: Dataset example

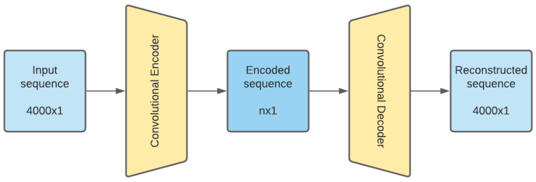

Figure 2: Autoencoder model

An example of the normalized dataset is shown in Fig. 1.

5 Methodology

For the purpose of seismic data compression, many new machine learning models have been proposed. We begin by implementing an autoencoder model for seismic compression. Then we implement a dct and dwt based encoder. Finally we make a refinement to the basic autoencoder model.

5.1 Autoencoder model

The autoencoder model we have used is based on the work by Emad et al. in [1]. We have implemented the proposed model - 1 from their paper for our comparison. Proposed model - 1 consists of a basic encoder decoder network as shown in Fig. 2.

It consists of an input signal of size 4000x1 that we pass through a 1d convolutional encoder to get an encoded signal of size nx1. By changing n, the size of the middle layer, we can get different CRs(compression ratios) for the model. This is then passed through mirrored 1d decoder to get the reconstructed signal. An example of this type of model for CR = 5:1 is shown in Fig. 3.

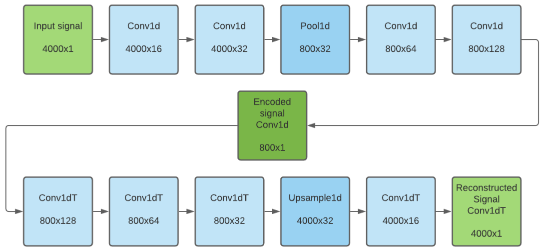

Figure 3: Autoencoder model for CR 5:1

The input signal is passed through a series of 1d convolutional blocks and 1d max-pooling blocks to get the encoded representations. The reconstructed signal is then obtained by using mirrored 1d transpose convolutional blocks and upsample 1d blocks. The exponential linear unit(ELU) is used as activation for all convolutional layers except the last one. The last layer, used for reconstruction, does not use any activation. The ELU for $\alpha > 0$ is:

$$elu(z) = \begin{cases} z, & \text{for } z > 0 \\ \alpha * (e^z - 1), & \text{for } z \leq 0 \end{cases}$$

where α controls the scale of the negative input.

A number of such models are provided for different CRs by [1]. We have thus implemented these models for comparison.

5.2 DCT model

A standard for 2d image compression is DCT(Discrete cosine transform) compression. So we have adapted the DCT compression algorithm for 1d compression. The adapted DCT compression algorithm is shown in Fig. 4.

5.2.1 Discrete cosine transform

First, the DCT(Discrete Cosine Transform) is applied to the signal. The DCT is able to compress the energy of the signal into a smaller space. This will introduce statistical redundancies that can later be exploited. In 2d data compression, the image is split into blocks of size 8x8 before compression. This is because splitting into smaller blocks highlights the spatial redundancy in each block and because dct computation is faster for smaller block sizes. To adapt this into a 1d signal, we split the whole signals into non-overlapping samples of size 32 before applying the dct transform to each individual sample.

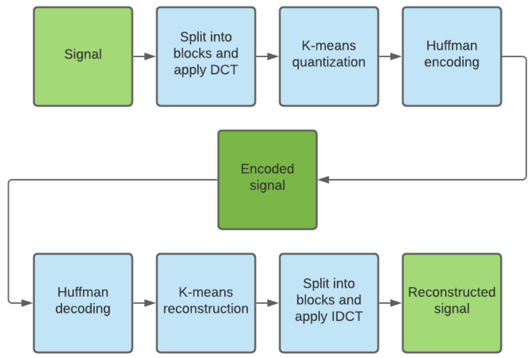

Figure 4: DCT compression algorithm

5.2.2 Quantization

For quantization, since we do not have a uniform range for the dct transformed signal, we use a k-means quantization algorithm. K-means is a clustering algorithm which partitions a sequence of vectors into k clusters which allows us to represent every sample in our 1d signal by one of k symbols. Thus the signal is quantized. By changing the value of k, we can change the CR as well as the SNR(signal to noise ratio). A higher value of k will quantize the signal into more levels, but give us a smaller compression ratio and vice versa.

5.2.3 Huffman encoding

Many statistical redundancies were introduced when the dct transform took place. This can be exploited by applying huffman encoding to the quantized signal. This will give us our encoded sequence. The encoded sequence is used to find the compression ratio of the algorithm, by comparing the 4000 sampled, 64 bit float representation of the original signal.

5.2.4 Decoding

For decoding, we first take the huffman encoded signal and decode it using the huffman tree. Then we reconstruct the quantized signal by using the k-mean dictionary to convert the k symbols back into the reconstucted dct signal. Then the dct signal is split into 32 sized samples and idct is applied to each 32 sized sample. Thus we get the reconstructed signal, which can be used for evaluation of performance of this algorithm.

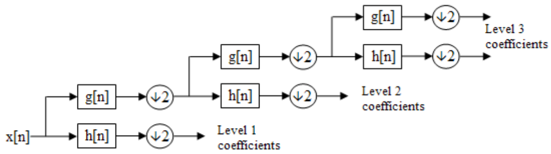

Figure 5: 3 level DWT

5.3 DWT model

Another compression model is the DWT(Discrete Wavelet Transform) model. A DWT can be applied to a signal to split it up into different signals that encode different information about the signal. It has traditionally been used for compression of images and compression of 1d signals such as ECG(Electrocardiogram).

5.3.1 Discrete Wavelet Transform

Applying the DWT to a 1d signal will split the 1d signal into two signals of half the length. The two signals it is split into is the course signal and the detail signal. The course signal is a rough estimation of the original signal and the detail signal is a signal that contains some of the details that were removed from the original signal to create the rough signal. Consecutively applying the DWT on the course signal obtained by the DWT can give us many levels of details for the signal. For this compression algorithm, a 3-level DWT was performed. The 3-level DWT model is shown in Fig. 5. It decomposes the 4000 sample signal into D1(2000 samples), D2(1000 samples), D3(500 samples) and C3(500 samples), i.e. 3 detail signals and 1 course signal. For the purposes of this paper, we have tested a few different wavelets on the training sample and decided to use the 'db2' wavelet as it was giving superior performance.

5.3.2 Quantization

For quantization, we use the same algorithm described in 5.2.2. However unlike the DCT signal where every part could be given equal importance, in the DWT decomposition, higher levels are more important than lower levels. Thus we should give priority to these levels while quantizing the signal. Thus for levels 1, 2 and 3, we apply the quantization algorithm seperately as described in 5.2.2 with different levels k1, k2, k3 where $k3 >= k2 >= k1$. Tweaking the values of k1, k2 and k3 will give us different values of CR and corresponding quality of reconstruction.

5.3.3 Huffman encoding

Huffman encoding is then applied to each individual signal as described in 5.2.3. We use the huffman encoded signal to get the CR of the algorithm.

5.3.4 Decoding

For decoding, we decode the huffman encoded signal using huffman tree and reconstruct the signal using k-mean dictionary. Then the signals are given back to a 3 level Inverse Discrete Wavelet Transform filter for reconstruction. We use the reconstructed signal to get the performance metrics.

5.3.5 DWT compression ignoring D1

D1 being the first level of detail obtained using the DWT filter corresponds to very fine detials. This can be ignored to reduce the size of the signal by half and increase compression. This will have some impact on quality of reconstruction, but the decrease in quality is mostly negligible.

5.4 Autoencoder with quantization

The autoencoder described in 5.1 has the drawback of not using quantization as compared to the other models. To address this drawback, we have attempted to add quantization to the autoencoder model. The encoded layer may not have the same amount of redundancy as some of the other methods, so applying k-means quantization and huffman encoding wouldn't be as helpful here. Instead we choose to apply uniform quantization. To this end, we replace the ELU activation in the encoded layer with a sigmoid activation. The sigmoid activation is given by-

$$sigmoid(z) = \frac{1}{1 + e^{-z}}$$

The sigmoid function takes the input and maps it to the range (0,1). Since we know the range of the encoded signal now, we can apply uniform quantization. For this we choose a quantization constant Q to divide the whole signal by and then take the floor of it to convert it to an integer. Thus-

$$quantized(x) = floor(\frac{x}{Q})$$

The number of bits required to store a quantized sample in the range of (0,1) for a quantization constant Q can then be given by-

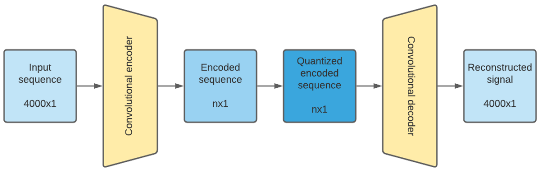

Figure 6: Quantized autoencoder model

$$no\_of\_bits(Q) = ceil(\log_2 \frac{1}{Q})$$

Using this formula we can obtain the CR for different values of Q. The quantized signal is then fed back to the encoder of the autoencoder model and the reconstructed signal is obtained. The reconstructed signal is then used for getting the quantitative performance metrics. The whole model is shown in Fig. 6.

6 Results and Discussions

As part of the evaluation, we have analyzed the different models based on 3 different metrics which are Normalised Mean Square Error, Normalised Root mean Square error and Signal to Noise Ratio.

1. Normalised Mean Square Error :- It is calculated by taking the square of the difference in values between the original signal and the reconstructed signal at each point and is given by the formula :-

$$NMSE = \frac{\sum_{n=1}^{K}(x - x_1)^2}{\sum_{n=1}^{K}(x)^2}$$

   In the above formula, K is the seismic signal index, x is the instantaneous values of the original signal and x_i is the reconstructed signal.

2. Normalised Root Mean Square Error :- It is normally used to find the distortion level when lossy compression is used. It is given by the formula :-

$$NRMSE = \frac{\sqrt{mean[(x - x_1)^2]}}{max(x) - min(x)}$$

   In the above formula, x is the original signal and x_i is the reconstructed signal.

3. Signal to Noise Ratio :- It is a very popular evaluation metric used to compare the quality of the signal. It is calculated using the formula :-

$$SNR(dB) = \log_{10}(NMSE)$$

Here, NMSE is the Normalised Mean Square Error.

6.1 Autoencoder Model Performance

The autoencoder model mentioned in section 5.1 was tested for different Compression Ratios based on the different evaluation metrics. The results are shown in Table 1.

Table 1: Performance of Autoencoder Model

CR	NMSE(10^{-3})	NRMSE(10^{-3})	SNR(dB)
100	21.53	70.72	17.62
50	16.53	62.98	18.50
30	13.35	56.65	19.41
15	3.11	27.82	25.53
5	0.088	4.67	40.86
2	0.023	2.40	46.67

Analyzing the results shown in table 1 we can see that as the compression ratio increases we achieve higher NMSE and NRMSE values. Further the reconstructed signal has a smaller signal to noise ratio for higher compression ratios. The results indicate that to have the smallest error rates and the most accurate reconstruction, one should use lower compression ratios and compromise on performance.

6.2 DCT Model Performance

The Discrete cosine transform model mentioned in section 5.2 was tested for different values of K(value used for k-means quantizer) based on the different evaluation metrics. The results are shown in Table 2.

The results shown in Table 2 show that for a higher value of K, we get lower compression ratios. Moreover, a higher compresion ratio results in higher errors(NMSE,

Table 2: Performance of DCT Model

K	CR	NMSE(10^{-3})	NRMSE(10^{-3})	SNR(dB)
2	64.01	24.68	74.96	17.22
4	60.81	12.69	53.53	20.21
8	56.31	3.88	29.63	25.37
16	50.19	0.93	14.46	31.73
32	41.03	0.19	6.43	38.95
64	30.17	0.03	2.56	47.18
128	20.38	0.0037	0.87	56.79

NRMSE) and a lower SNR. Overall, if we have higher k(more symbols), we will have better reconstruction but worse compression ratio. For a compression ratio of approximately 30, we see that the DCT model performs better than the autoencoder model by giving lower errors and higher SNR.

6.3 DWT Model Performance

The Discrete cosine transform model mentioned in section 5.3 was tested for different values of K1,K2,K3(values used for k-means quantizer) based on the different evaluation metrics. The metrics are evaluated for both the normal DWT model as well as the model obtained by ignoring the D1 coefficients as mentioned in section 5.3.5. The results of DWT with D1 coefficients are shown in Table 3.

The figures in Table 3 show that an increase in any of the parameters K1, K2 and K3 ends up giving a lower compression ratio. Lower values of K(K1,K2,K3) give better performance but also result in higher error rates and lower SNR for the reconstructed signal. For an approximate compression ratio of 30, the DWT model gives almost equivalent outputs as compared to the the DCT model. Further, for CR=30, the DCT model performs better than the autoencoder model by giving lower errors and higher SNR.

The results of the DWT model when ignoring the D1 coefficients is shown in Table 4. It is seen that the compression ratios obtained here are almost 1.5-2 times the compression ratio when D1 coefficients are included in the model. Further, the error

Table 3: Performance of DWT Model

K1	K2	K3	CR	NMSE(10^{-3})	NRMSE(10^{-3})	SNR(dB)
2	2	2	63.88	14.09	56.94	19.56
2	2	4	58.46	4.73	33.47	24.08
2	4	8	48.93	1.18	16.56	30.31
2	4	16	41.39	0.27	7.81	36.88
2	8	16	38.22	0.25	7.62	37.19
4	8	32	31.14	0.05	3.42	44.16
4	8	64	27.15	0.01	1.70	49.88
8	16	64	22.63	0.0086	1.37	52.22

rates and SNR obtained in this case is almost exactly equivalent with and without the D1 coefficients.

6.4 Autoencoder Model with Quantization Performance

The Autoencoder model with quantization as mentioned in section 5.4 was tested for different values of the quantization constant(Q) based on the different evaluation metrics. The metrics are evaluated for both the 30:1 CR and 5:1 CR models as mentioned in the paper [1]. The results of them are shown in Tables 5 and 6 respectively.

Table 5 shows the performance of the 30:1 CR autoencoder Model with quantization. We notice that, a higher value of Q gives better performance in terms of compression ratios. However, a higher Q also results in a lower SNR and higher rates of error for the reconstructed signal. However, we notice also that unlike the other models, decreasing Q does not continue to increase SNR beyond a point. This is because the quantized model cannot achieve a better reconstruction than the unquantized model, and thus will saturate at the SNR for unquantized model.

The compression rates achieved for this model is however much higher than without quantization as seen in Table 1. A SNR of 17 is obtained with a CR of 385 with quantization, while a CR of 100 itself resulted in SNR of 17 without quantization. Hence we can see that quantization greatly improves the performance of the

Table 4: Performance of DWT Model without D1 coefficient

K2	K3	CR	NMSE(10^{-3})	NRMSE(10^{-3})	SNR(dB)
2	2	127.65	14.09	56.93	19.56
2	4	107.62	4.73	33.47	24.08
4	8	79.28	1.18	16.56	30.29
4	16	61.13	0.27	7.85	36.82
8	32	44.14	0.05	3.58	43.60
16	64	32.63	0.01	1.77	49.51

autoencoder model.

Table 6 shows the performance of the 5:1 CR autoencoder Model with quantization. Similar to the 30:1 CR model, a higher value of Q gives better performance in terms of compression ratios while resulting in a lower SNR and higher rates of error for the reconstructed signal. For a given value of Q, the 30:1 CR model gives much higher performance and compression ratios as compared to the 5:1 CR model. Remarkably, for a value of Q=0.1, the 30:1 model gives a compression ratio more than 5 times that of the 5:1 CR model without a much poorer value of SNR.

6.5 CR vs SNR Performance for all Models

The graph in Figure 7 shows the Compression ratio vs Signal to noise ratio comparison for the different models implemented in this paper. It can be seen that generally with an increase in the compression ratio, the signal to noise ratio reduces. The best compression ratios are obtained with the 30:1 Autoencoder model with quantization, however the SNR is saturated near a level of approximately 20.

The DWT model without D1 coefficients appears to be the next best model giving high compression ratios with a similar level of SNR as compared to the remaining models. Further, the standard convolutional autoencoder model without quantization seems to be give the least performance as compared to all models, with slightly less compression ratios as compared to the rest. The remaining models all give similar levels of SNR vs CR.

Table 5: Performance of the 30:1 CR Autoencoder Model with Quantization

Q	CR	NMSE(10^{-3})	NRMSE(10^{-3})	SNR(dB)
0.1	481.32	59.25	118.22	12.90
0.05	385.05	24.04	77.05	16.56
0.01	385.05	24.04	77.05	16.56
0.005	240.66	13.21	56.39	19.45
10^{-3}	192.52	13.09	56.13	19.49
10^{-5}	113.25	13.08	56.11	19.50

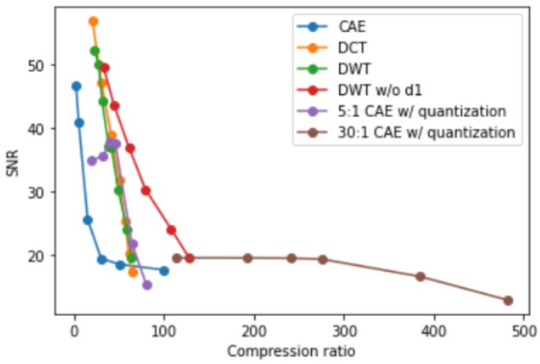

Figure 7: Compression Ratio vs SNR comparison for the different models

7 Conclusion

DWT, DCT and convolutional autoencoder model was implemented for seismic data compression. The different models were compared quantitatively. A model was developed that provides very high CRs at reasonable SNR as compared to all the other models.

It was observed that the DWT model without D1 provided the best SNR for mid-high level of compression. However, due to the presence of the k-mean and huffman step, the algorithm for DWT and DCT compression takes place linearly with respect to number of signals to be compressed, while the autoencoder models can take advantage of vectorization and can compress and reconstruct multiple signals much quicker.

Table 6: Performance of the 5:1 CR Autoencoder Model with Quantization

Q	CR	NMSE(10^{-3})	NRMSE(10^{-3})	SNR(dB)
0.1	80.02	33.84	89.24	15.39
0.05	64.02	7.26	41.95	21.82
0.01	45.73	0.19	6.85	37.56
0.005	40.01	0.17	6.66	37.86
10^{-3}	32.01	0.28	8.69	35.53
10^{-5}	18.82	0.33	9.38	34.84

8 Future works

The future works include the following:

- Look at more complicated neural net models for seismic data compression.
- Look to combine the output of the dwt model and neural networks to provide superior compession, as these were the best models found in our comparison.

References

[1] Emad B. Helal et al. "Seismic Data Compression Using Deep Learning". In: *IEEE Access* 9 (2021), pp. 58161–58169. DOI: 10.1109/ACCESS.2021.3073090.

[2] Giridhar Mandyam, Neeraj Magotra, and W. McCoy. "Lossless seismic data compression using adaptive linear prediction". In: *International Geoscience and Remote Sensing Symposium (IGARSS)* 2 (June 1996), 1029–1031 vol.2. DOI: 10.1109/IGARSS.1996.516556.

[3] John Miller. "Utah FORGE: 2D and 3D Seismic Data". In: (Mar. 2018). DOI: 10.15121/1452746.

[4] A.S. Spanias, S.B. Jonsson, and S.D. Stearns. "Transform methods for seismic data compression". In: *IEEE Transactions on Geoscience and Remote Sensing* 29.3 (1991), pp. 407–416. DOI: 10.1109/36.79431.

[5] J.D. Villasenor, R.A. Ergas, and P.L. Donoho. "Seismic data compression using high-dimensional wavelet transforms". In: (1996), pp. 396–405. DOI: 10.1109/DCC.1996.488345.

[6] Zhengbing Zhang, Wei Zhang, and Zhixian Gui. “Seismic data compression based on wavelet transform”. In: *Proceedings of SPIE - The International Society for Optical Engineering* 9273 (Oct. 2014). DOI: 10.1117/12.2071594.