Historical Style Generator (*HStyle*)

Project Book

Academic Advisor - Dr. Irina Rabaev

Students - Leor Ariel Rose, Yahav Bar David

Project number:	BS-SE-21-89
Written by:	Leor Ariel Rose and Yahav Bar David
Last updated:	04/11/20
Verified by:	Dr. Irina Rabaev
Verified date:	04/11/20

Table of Contents

Introduction **10**

System Purpose 10

Document Conventions 10

Intended Audience and Reading Suggestions 10

Terminology 10

Literature Review **13**

Abstract 13

First Approach - Using Deep Learning 13

Second Approach - Using Computer Vision Methods 21

Neural Style Transfer Model - First Approach **23**

Concepts and Principles 23

Content Loss 23

Style Loss 23

Gram Matrix 23

Total variation loss 24

Total loss 24

Model Architecture 24

Elements to experiment 24

Computer Vision Methods - Second Approach **25**

Using Kanungo noise model 25

Binarization 25

Distance Transform 25

Probability calculation 25

Seed Threshold 26

The Model We Will Use 26

Kanungo noise model 26

Noise region 26

Noise generation 26

Software Project Management Plan (SPMP) **27**
Project Organization 27
Process Model 27
Organizational Structure 27
Managerial Process 27
Management Objectives and Priorities 27
Assumptions, Dependencies, and Constraints 27
Risk Management 28
Technical Process 30
Methods, Tools and Techniques 30
Work Packages, Schedule 32
Work Packages 32
Dependencies 32
Resource Requirement 33
Schedule 33
Software Requirements Specification (SRS) **34**
Overall Description 34
Product Perspective 34
Product Functions 34
User Classes and Characteristics 35
Operating Environment 35
Design and Implementation Constraints 35
User Documentation 35
Assumptions and Dependencies 35
Assumptions 35
Dependencies 36
External Interface Requirements 36
User Interfaces 36
Hardware Interfaces 36
Software Interfaces 36

Communications Interfaces 36
System Features 36
Historical document synthesis 36
Description and Priority 36
Stimulus/Response Sequences 37
Functional Requirements 37
About 38
Description and Priority 38
Stimulus/Response Sequences 38
Functional Requirements 38
Other Non-functional Requirements 39
Performance Requirements 39
Safety Requirements 39
Security Requirements 39
Software Quality Attributes 39
Results and conclusion **40**
Original content image 40
Dilate content image 40
Binary content image 41
Binary-Dilate content image 41
References **42**
Appendix A - Experiments Analysis **44**
Fast arbitrary image style transfer 44
Experiment 1 44
Content Input 44
Style Input 44
Results 44
Discussion 45
Experiment 2 45
Content Input 45

Style Input 45
Results 45
Discussion 45
Neural Style Transfer 46
Content layers 46
Style layers 50
Weights 50
Experiment 1 51
Content Input 51
Style Input 51
Results 51
Weight 0 52
Weight 1 53
Weight 2 54
Weight 3 55
Weight 4 56
Weight 5 57
Weight 6 58
Weight 7 59
Weight 8 60
Discussion 60
Experiment 2 61
Content Input 61
Style Input 61
Results 61
Weight 0 62
Weight 1 63
Weight 2 64
Weight 3 65
Weight 4 66
Weight 5 67

Weight 6 68
Weight 7 69
Weight 8 70
Discussion 70
Experiment 3 71
Content Input 71
Style Input 71
Results 71
Weight 0 72
Weight 1 73
Weight 2 74
Weight 3 75
Weight 4 76
Weight 5 77
Weight 6 78
Weight 7 79
Weight 8 80
Discussion 80
Experiment 4 81
Content Input 81
Style Input 81
Results 81
Weight 0 82
Weight 1 83
Weight 2 84
Weight 3 85
Weight 4 86
Weight 5 87
Weight 6 88
Weight 7 89
Weight 8 90

Discussion 90

Experiment 5 91

Content Input 91

Style Input 91

Results 91

Weight 0 92

Weight 1 93

Weight 2 94

Weight 3 95

Weight 4 96

Weight 5 97

Weight 6 98

Weight 7 99

Weight 8 100

Discussion 100

Experiment 6 101

Content Input 101

Style Input 101

Results 101

Weight 0 102

Weight 1 103

Weight 2 104

Weight 3 105

Weight 4 106

Weight 5 107

Weight 6 108

Weight 7 109

Weight 8 109

Discussion 110

Experiment 7 110

Content Input 111

Style Input 111
Results 111
Weight 0 111
Weight 1 113
Weight 2 114
Weight 3 115
Weight 4 116
Weight 5 117
Weight 6 118
Weight 7 119
Weight 7 120
Discussion 120
Experiment 8 120
Content Input 121
Style Input 121
Results 121
Weight 0 122
Weight 1 123
Weight 2 123
Weight 3 124
Weight 4 125
Weight 5 126
Weight 6 127
Weight 7 128
Weight 8 129
Discussion 130
Appendix B - Project Data **130**
English Data 130
Modern English 131
Cropping 131
Hebrew Data 131

Modern Hebrew 131

Changing Type 131

Rotating 131

Cropping 131

Trehoulding 131

Cairo Geniza Historical Hebrew 131

Middle Ages Historical Hebrew 131

Arabic Data 131

Modern Arab 131

Changing Type 131

Historical Arab 131

Latin Data 131

Historical Latin 132

Introduction

1. System Purpose

Historical documents can reveal a lot of information, such as form of writing, wording, content that did not exist and more. Since those documents are usually written in diaries, pages or letters that have a final lifespan, there are certain periods that lack historical documents. The lack of data restricts us from using the power of deep learning and machine learning models, because most of these models require a large amount of data (big data).

HStyle lets users synthesize a historical document. What does this mean? Given two documents, one is a historical document (the style), and the other is a modern document (the content), *HStyle* will transfer the style from the historical document to the modern document. This system is based on a deep learning model and computer vision techniques and will be a web based application.

2. Document Conventions

This document uses the following conventions:

- Important terms and headings will be marked in **bold**.
- Heading one, Heading two, Heading Three, Heading Four, Normal text will correspond to these font sizes: 16, 14, 13, 12, 12.
- System name will be marked in *italic*.

3. Intended Audience and Reading Suggestions

HStyle is a web based application for the historical style document synthesis. *HStyle* is implemented under the guidance of college advisors. This system is useful for any user who wishes to synthesize new documents with historical style.
Anyone with some programming experience and familiarity in Python, can understand this document. The document is intended for developers, software architects, testers, project managers and documentation writers.

4. Terminology

User	Any human being who is interacting with the software is a user
System	The package of all the components which takes input and gives output to demonstrate the features of the software is called a system
Machine Learning	Machine Learning(ML), is the field of study that gives computers the capability to learn without being explicitly programmed
Deep Learning	Deep Learning(DL), is a subfield of ML concerned with algorithms inspired by the structure and function of the brain called artificial neural networks

Computer Vision	Computer Vision (CV), Computer vision is an interdisciplinary scientific field that deals with how computers can gain high-level understanding from digital images or videos
Big Data	Big Data is a collection of data that is huge in volume, yet growing exponentially with time
Python	Python is an interpreted, high-level and general-purpose programming language
convolutional neural network	convolutional neural network (CNN) is a class of deep neural networks, most commonly applied to analyzing visual imagery
VGG16	VGG-16 is a convolutional neural network that is 16 layers deep
VGG19	VGG-19 is a convolutional neural network that is 19 layers deep
Generative adversarial network	Generative adversarial network(GAN) is a machine learning model in which two neural networks compete with each other to become more accurate in their predictions
SC-GAN	Type of GA called self-attention conditional GAN.
AdaIN	Adaptive Instance Normalization (AdaIN) is a normalization method that aligns the mean and variance of the content features with those of the style features
Gram Matrix	In linear algebra, the Gram matrix of a set of vectors in an inner product space is the Hermitian matrix of inner products, whose entries are given by
Activation layer	The activation function is a mathematical "gate" in between the input feeding the current neuron and its output going to the next layer
LSTM	Long short-term memory (LSTM) is an artificial recurrent neural network architecture used in the field of deep learning
StyleGAN	StyleGAN is a type of generative adversarial network
Neural Style Transfer	Neural Style Transfer (NST) refers to a class of software algorithms that manipulate digital images, or videos, in order to adopt the appearance or visual style of another image
CycleGan	The CycleGAN is a technique that involves the automatic training of image-to-image translation models without paired examples
Content image	Image representing content
Style image	Image representing style
Binary image	A binary image is one that consists of pixels that can have one of exactly two colors, usually black and white
Degradation	The condition or process of degrading or being degraded
Euclidean distance	In mathematics, the Euclidean distance between two points
Gradient descent	Gradient descent is a first-order iterative optimization algorithm for finding a local minimum of a differentiable function

Epoch	An epoch is a term used in machine learning and indicates the number of passes of the entire training dataset the machine learning algorithm has completed
Angular	Angular is a platform for building mobile and desktop web applications
Hypertext Markup Language	Hypertext Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser
Cascading Style Sheets	Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in a markup language such as HTML
JavaScript	JavaScript, often abbreviated as JS, is a programming language that conforms to the ECMAScript specification
TypeScript	TypeScript is a programming language developed and maintained by Microsoft
FastAPI	FastAPI is a modern, fast (high-performance), web framework for building APIs with Python 3.6+ based on standard Python type hints
RESTful API	A RESTful API is an architectural style for an application program interface (API) that uses HTTP requests to access and use data
Uvicorn	Uvicorn is a lightning-fast ASGI server implementation, using uvloop and httptools
Tensorflow	An end-to-end open source machine learning platform for everyone
OpenCV	OpenCV is a library of programming functions mainly aimed at real-time computer vision

Literature Review

1. Abstract

Historical documents can reveal a lot of information, such as form of writing, wording, content that did not exist and more. Since those documents are usually written in diaries, pages or letters that have a final lifespan, there are certain periods that lack historical documents. The lack of data restricts us from using the power of deep learning and machine learning models, because most of these models require a large amount of data (big data).
In this study we want to test whether it is possible to synthesize historical documents using two sources: a historical document (as a style) and an image of modern handwritten text (as a content).

2. First Approach - Using Deep Learning

Transferring the style from one image into another can be considered a problem of texture transfer. In texture transfer the goal is to synthesize a texture from a source image while constraining the texture synthesis to preserve the semantic content of a target image.

In the papers of Gatys et.al. [1] and [2], the researchers discovered a new technique for texture synthesis, an artificial system based on a deep neural network that creates artistic images of high perceptual quality. In those studies, the researchers used a CNN (Convolutional Neural Network) called VGG19 from the Caffe deep learning framework explained in [3] which is known as a deep network to classify images. The researchers used the output of the middle layers in the CNN network to extract the representation of the images. In each layer a representation of the image is created, and it is possible by rebuilding this representation to see the content obtained from this layer and take the output from the layer in which the representation of content and style is ideal.
As you can see in Figure 1 the researchers found that in the first layers the output of the layers is almost identical to the original image, while the content is in the higher layers (deeper in the neural network).
In addition, the researchers found that the style of an image can be described by the means and correlations across the different feature maps. Therefore, they calculate a Gram matrix that includes this information by taking the outer product of the feature vector with itself at each location and averaging that outer product over all locations.
To connect the content representation and the style representation, the researchers create a new image that matches the two representations while reducing the distance and loss of information until reaching an acceptable threshold.
In this study, the researchers were able to come to the understanding that there is a clear separation between content and style and that images can be combined to produce a synthesis of a particular style on a source image, as can be seen in Figure 1.
In addition, the researchers came to the understanding that there is a trade-off between content and style, which means that coefficients can and should give more importance to the content of the image or the style of the image in order to achieve desired results.

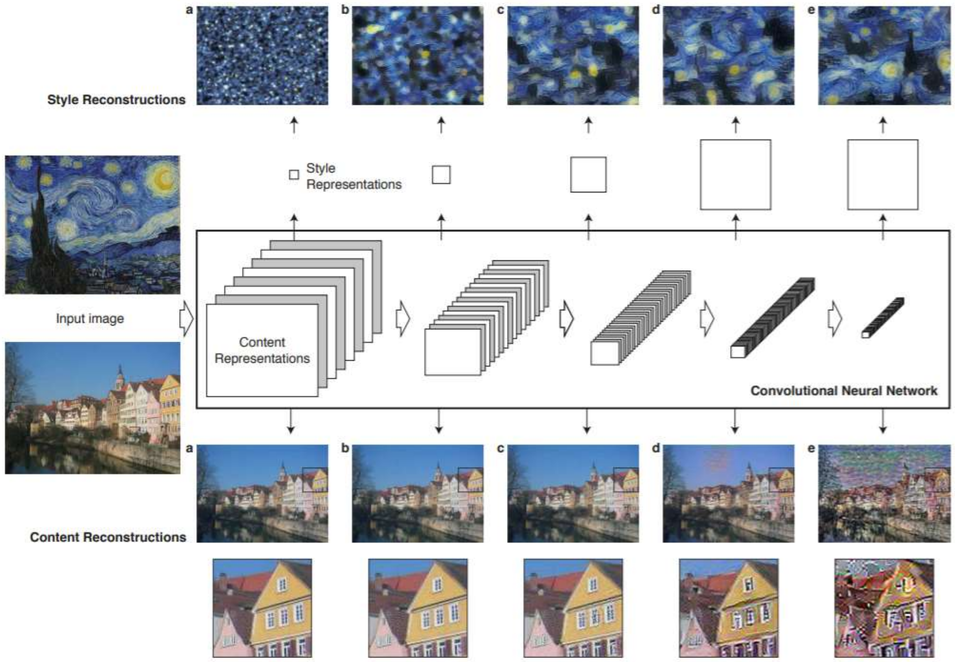


Figure 1 – Taken from "Image Style Transfer Using Convolutional Neural Networks". The Figure shows the content and style in the VGG network.

Based on Gatys et.al papers [1] [2], a few papers were released about image style transfer, some of these papers use the original VGG19 architecture and some use additional techniques or models.

In the paper of Ter-Sarkisov [4] the researcher tries to transfer style from the logos of heavy metal bands onto corporate logos using a VGG16 network.
The researcher used the VGG16 CNN architecture to extract style and content. Like the original paper [2], in this research he used the same technique to extract the style from heavy metal bands logos and apply them onto corporates logos technique (they performed experiments trying to find which layers are best for extracting content and style).
It was found that there is a significant trade-off between content and style in order to maintain the logo readable. Since the researcher only wanted to transfer the font style, he suggests that this model is not suitable for this mission and that maybe it needs to be approached in a different way. Some of his results are visible in Figure 2.

Figure 2 – Taken from "Network of Steel: Neural Font Style Transfer from Heavy Metal to Corporate Logos". The Figure shows the company logos with metal bands style.

Although Ter-Sarkisov [4] says this technique is not the best for the text, in the paper of Atarsaikhan et.al. [9] the researchers generate fonts by using neural style transfer. The VGG16 model was used to extract and determine the content representation of a source image and feature maps of a style image to synthesize them together.
It was found that α (the coefficient of content representation) should be bigger than β (the coefficient of style representation) in order to give the content more importance.
The researchers showed that it is possible to generate readable fonts using neural network approach. They observed that font style can only be transferable if style images contain multiple characters. Also, the researchers managed to transfer a non-font style to sources like icons and graphs. In addition, they found that if the difference between source and style is large the result characters will be illegible. Some of their results are illustrated in Figure 3.

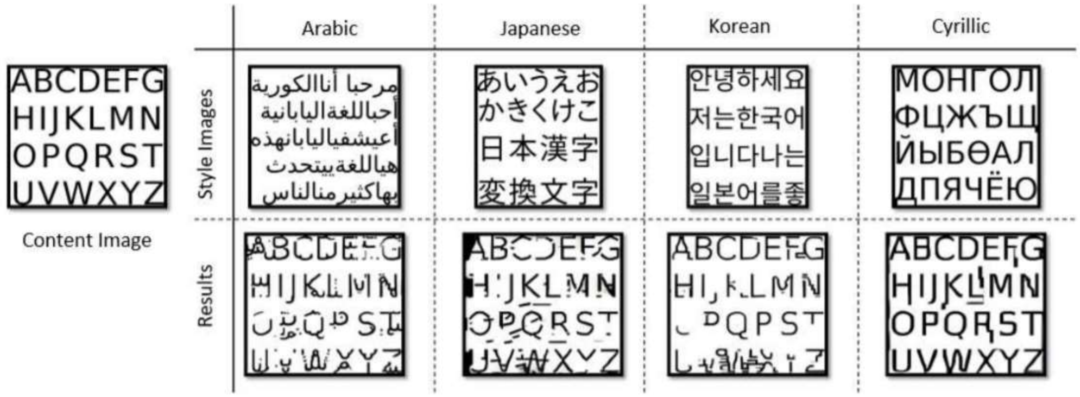


Figure 3 – Taken from "Neural Font Style Transfer". The Figure shows the results of transferring style to content image from English content and foreign languages.

The works of Ter-Sarkisov [4] and Atarsaikhan et.al. [5] study the style transfer on isolated words and letters and not full sentences or documents. In addition, they focus on font transformation and not the background and paper style that is very important in the field of historical documents. In the paper of Gomez et.al. [6], the researchers suggest a new architecture that learns the features that encode a certain text style and can transfer them to other text instances while preserving their content. In addition, they built a model that transfers style to only desired image pixels. The researchers used a VGG16 model for style transfer and two different models for selective style transfer (transfer style to only desired image pixels).
They found out that the style transfer model is able to learn text styles as the characters shapes, line style, and colors, and to transfer it to an input text preserving the original characters. Some of their results shown in Figure 4 and Figure 5.

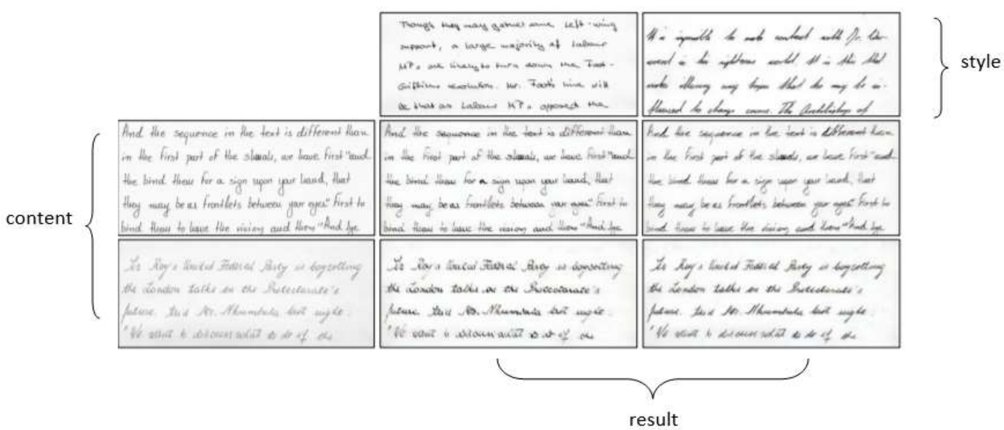


Figure 4 – Taken from "Selective Style Transfer for Text". The Figure shows the result of their model to transfer content with given style.

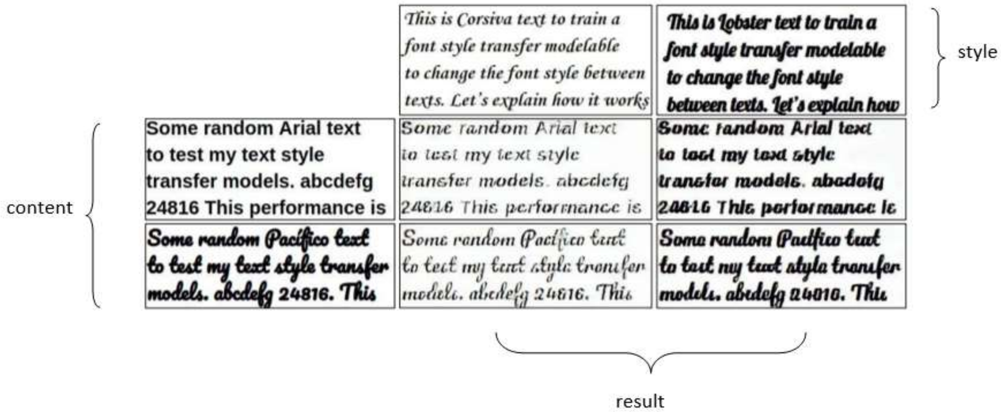


Figure 5 – Taken from "Selective Style Transfer for Text". The Figure shows the result of their model to transfer content with given style.

Some researchers combined the original model architecture with VGG19 with other techniques, like in the paper of Guan et.al. [7]. The work proposes a style conditioned generative adversarial network (SC-GAN). This network is used to transfer the styles of real handwriting images to skeleton images extracted from handwriting samples to generate photo-realistic text line images. In their paper the researchers used a VGG19 model to extract content and style from given images. First, given a content image they create a skeleton image. Then the skeleton and style image are given to a VGG19 model and combined after that by a generator that uses AdaIN and a discriminator GAN model. In this paper the researchers showed that they can create synthetic images, whose styles look quite similar with the corresponding style images. Some of their results are presented in Figure 6.

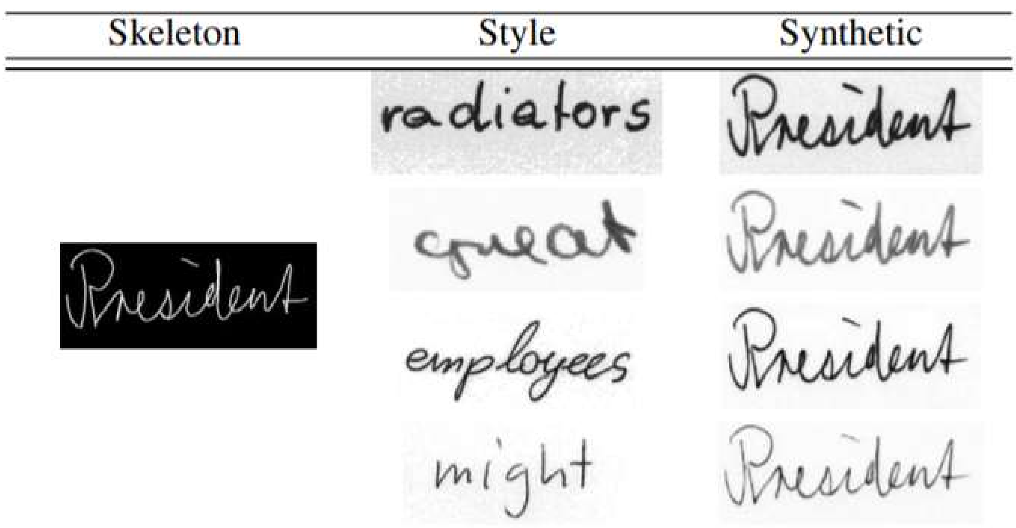


Figure 6 – Taken from "Improving Handwritten OCR with Augmented Text Line Images Synthesized from Online Handwriting Samples by Style-Conditioned GAN". The Figure shows the results of transferring style to skeleton image.

Another example is the paper of Kang et.al. [8]. The researchers suggest a generator aimed at transferring writing style features from one sample to another in an

image-to-image translation approach.
Given a pair of handwritten word images (x_i , x_j), the proposed module extracts content and style using the VGG19 model, and then combined by another model to generate a new handwritten word image. To combine the content and style, instead of using Gram matrix, they used a generator model that consists of two residual blocks with AdaIN normalization layers followed by four convolutional modules with nearest neighbor up-sampling, and a final tanh activation layer. Some of their results are illustrated in Figure 7, and as we can see from it, the researchers managed to combine style and content to a new image.

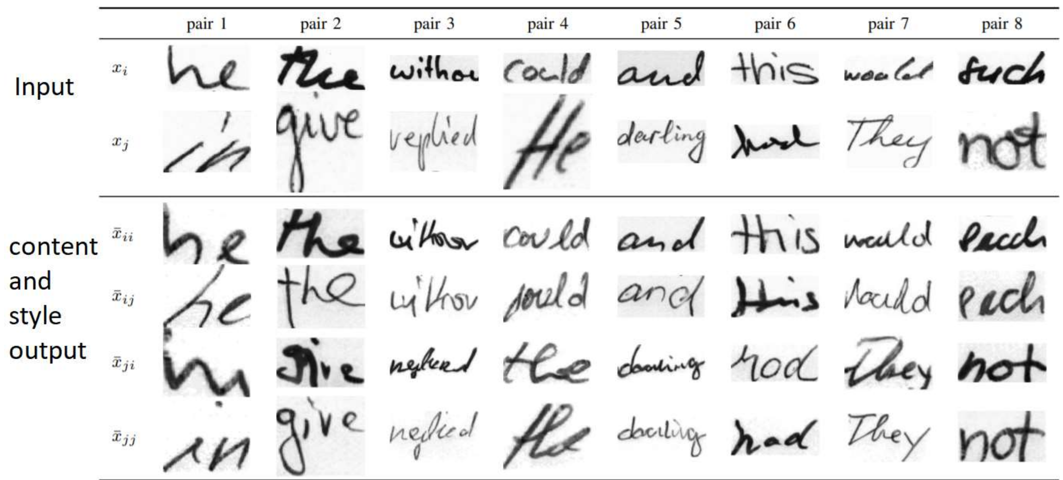


Figure 7 – Taken from "Distilling Content from Style for Handwritten Word Recognition". The Figure shows the results of transferring style to content given style and content image.

Similar to Guan et.al. [7] and Kang et.al. [8], in the paper of Liu et.al. [9] the researchers propose a word image generating method called Synth-Text Transfer Network (STN). They use an Encoder-AdaIN-Decoder architecture, where the encoder is a pre-trained VGG-19 with first few layers fixed in order to encode both content and style images. The decoder is learned to invert the AdaIN output to the image spaces. The AdaIN layer is used to conduct style transfer in the feature space. Then, they compute the content loss and the style loss by using the same VGG encoder. The researchers found that AdaIN is the most proper module for arbitrary style transfer. However, they mentioned that their method has limitations in the case where the style image has complicated background texture or uncommon font. Some of their results are shown in Figure 8.

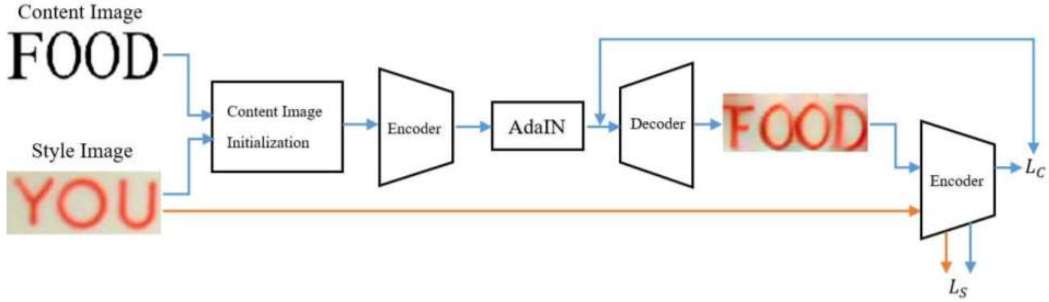


Figure 8 – Taken from “Synthesizing Scene Text Images for Recognition”. The Figure shows the Encoder-AdaIN-Decoder architecture of the proposed method.

In addition, there are different attempts to transfer image style in writing with different models and techniques.

For example, in the paper of Mayr et.al. [10] the researchers present a method for online handwriting synthesis, given a handwritten sample they produce a new style-adapted realistic-looking text.
In this paper the researchers tested their model on words and sentences. In order to extract the content they use Graves’ algorithm (LSTM), and in order to extract the style and combine it with the content they used the pix2pix generator network. To conclude, the researchers managed to create a fully automatic method to imitate handwriting using spatial-temporal style transfer. Some of their results are shown in Figure 9.

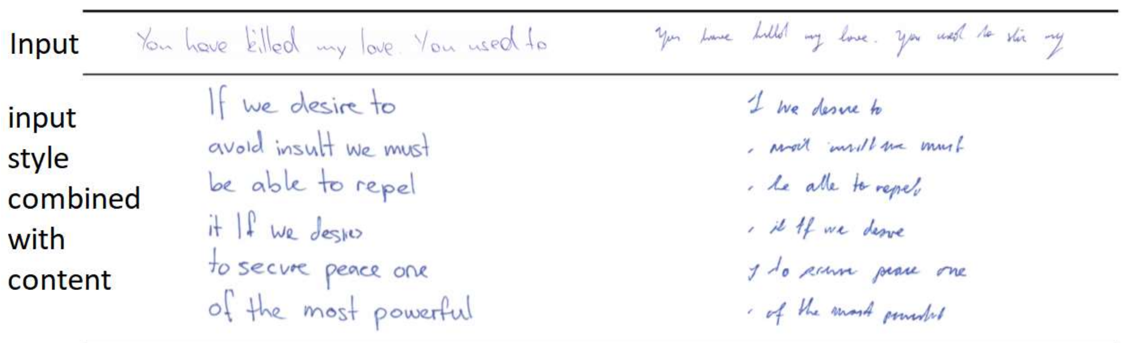


Figure 9 – Taken from “Spatio-Temporal Handwriting Imitation”. The Figure shows the results of transferring input handwriting style to different texts.

Another example is the paper of Davis et.al. [11]. The researchers present a GAN for generating images of handwritten lines conditioned on arbitrary text and style vectors. Given three inputs, content, style and noise, the model generates handwriting.
The researchers used a big and complex model made of six models in order to get the final result: (1) A generator network G to produce images from spaced text, a style vector and noise-based on StyleGAN; (2) A style extractor network S, that produces a style vector from an image and the recognition predictions; (3) A spacing network C, which predicts the horizontal text spacing based on the style vector; (4) A patch-based convolutional discriminator D; (5) A pretrained handwriting recognition network R to encourage image legibility and correct content; (6) A pre-trained encoder E, to compute a perceptual loss.
This method has presented a system capable of directly generating the pixels of a handwriting image of arbitrary length. Their model can extract a style from example images and their method does well at capturing the variations of global style in handwriting, such as slant and size. Some of their results are illustrated in Figure 10.

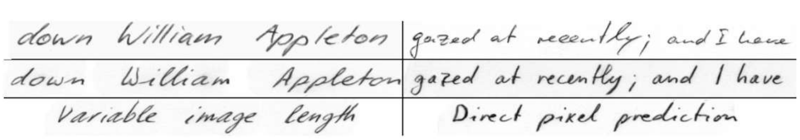


Figure 10 – Taken from "Text and Style Conditioned GAN for Generation of Offline Handwriting Lines". The Figure shows the results of mimicking two authors style, top sentence is the original author writing bottom is generated sentence.

All the mentioned papers (NST and GAN) focus more on the font transfer of the writing and less on the style of the document and the other aspects of the document. In addition, none of the papers try to implement the transfer on a full document image.

In the paper of Pondenkandath et.al. [12] the researchers apply document style transfer on historical documents. They use two models and compare results between them. The first model is GAN (cycleGAN) and the second model is VGG-19 CNN. The main motivation of this research is to generate massive synthetic datasets of "historical" documents which can be used for the training of document analysis systems. The researchers found that the synthetic images generated by the cycleGAN appear significantly better than those generated with NST. The researchers notice many similarities between the target domain samples and the synthetic samples. The overall style content of the target domain is well expressed. However, in a structural content point of view the initials are not well detected and expressed. The two column-mode is not at all expressed. When considering the synthetic documents produced with the NST, the structural content is better preserved. However, the style is mixed and standardized over the entire synthetic document, leading to the presence of a lot of colored artifacts. Also, the font does not change as compared to the synthetic images generated by the GAN. Some of their results are shown in Figure 11.

Figure 11 – Taken from "Historical Document Synthesis with Generative". The Figure shows the source transform given target with cycleGAN and NST.

As we can see there are various researches dealing with synthesizing writing/fonts from style image into content image. These studies are a great base for our research, but are missing a few aspects of what we want to investigate. Most of the discussed above work/deal with words/sentences/short paragraphs but not with *full* document images. In addition, all these papers do not consider the background/paper style to be part of the wanted style to transfer and they work with grey/binary images. The only paper that tries to create a full document is the paper of Pondenkandath et.al. [12], but this work is tested only on Latin languages and focuses more about the font and less about the style of the document. In our project, the goal is to transfer historical style (color, degradation of document and more features). In addition, we want to explore this transformation with the original simple model using VGG19 and not using GAN, transformers, and complex models.

To conclude, from the researches available today it seems our research is possible. We will want to answer a few questions in our research. Our main research question is (1) “Is it possible to transfer style from a full historical document image to a modern document image by using the simple image transfer style model?”. Additional questions we would like to explore (2) “What is the ratio/trade-off between content and style?”, (3) “Can we perform style transfer on documents that are not written in the same language?”.

3. Second Approach - Using Computer Vision Methods

In the case that the deep learning approach will not result in good historical synthesis there are computer vision methods that can enable us to create text degredization. These degredization texts can be pasted onto historical style background images, which will create a historical synthesis document.

In the papers of Kieu et.al. [13] the researchers suggest a technique for character degradation. This technique enables the synthesis of degradation on a character. The technique includes image binarization, probabilities and computer vision methods. This technique only works on grayscale images. As you can see in Figure 12 the results of degradation of a character from Kieu et.al. [13] paper.

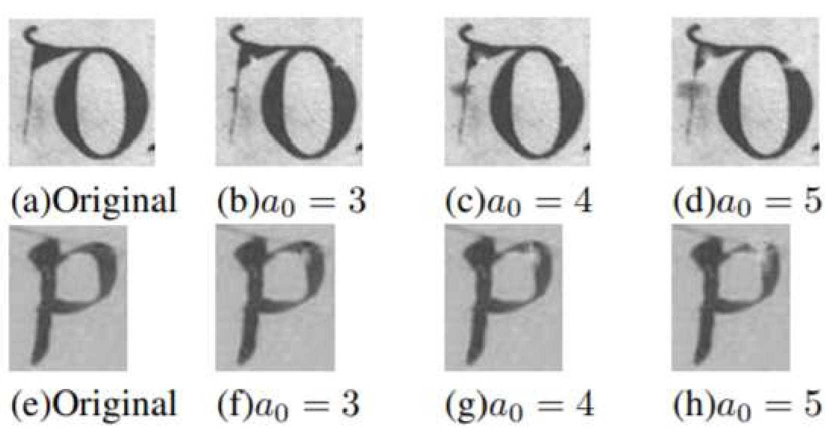


Figure 12 – Taken from “A character degradation model for grayscale ancient document images”. The Figure shows the degraded images with the seed-points fixed.

If images are color images we can use a similar technique suggested in Luyen et.al. [14] paper that works on color images. As you can see in Figure 13 the results of degradation of a color image paragraph from Luyen et.al. [14] paper.

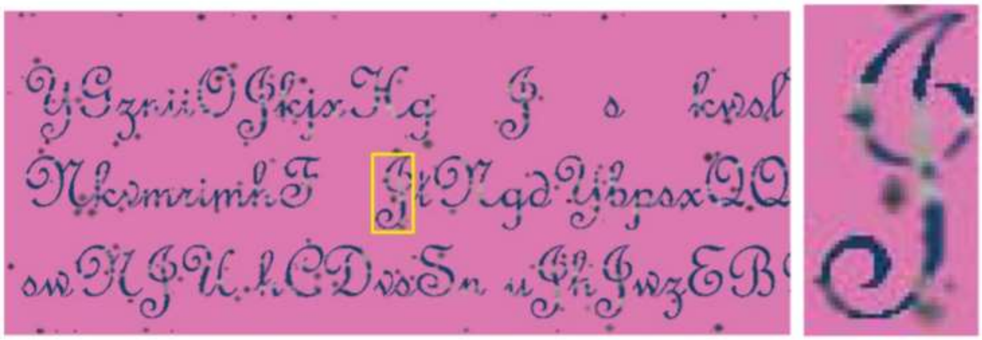


Figure 13 – Taken from "A Character Degradation Model for Color Document Images". The Figure provides the degraded image of the original one with 1616 seed-points. The right image represents the zoomed parts of the left image.

Neural Style Transfer Model - First Approach

Our system is based on the NST, the concepts and use is explained in this section.

1. Concepts and Principles

1.1. Content Loss

This function helps to check how similar the generated image is to the content image. It gives the measure of how far (different) are the features of the content image and target image. We will pass the network both the desired content image and our generated image. This will return the intermediate layer outputs from our model. Then we simply take the Euclidean distance between the two intermediate representations of those images. The loss equation:

$$L^{l}_{Content} = \sum_{i,j} (C_{i,j} - P_{i,j})^2$$

Content loss   ontent image   Generated image

1.2. Style Loss

Style Loss measures how different the generated image, in terms of style features, is from your style image. But it's not as straightforward as content loss. The style representation of an image is given by Gram Matrix. The loss equation:

$$L^{l}_{style} = \frac{1}{4 * N^2 * M^2} \sum_{i,j} (G^{s}_{i,j} - G^{p}_{i,j})^2$$

Style loss   Generated Image Dimensions   Generated image Size (H*W)   Gram matrix for style image   Gram matrix for generated image

1.3. Gram Matrix

The Gram Matrix G is the set of vectors in a matrix of dot products. To get the results, the matrix is multiplied by its transpose matrix, its equation is as follows:

$$\mathrm{G}^{\mathrm{l}}_{\mathrm{i,j}} = \sum_{k} F^{l}_{i,k} F^{l}_{j,k}$$

Gram Matrix   Original Matrix   Transpos e Matrix

1.4. Total variation loss

It was observed that optimization to reduce only the style and content losses led to highly pixelated and noisy outputs. To cover the same, total variation loss was introduced. The total variation loss is analogous to regularization loss. This is introduced for ensuring spatial continuity and smoothness in the generated image to avoid noisy and overly pixelated results.

1.5. Total loss

Loss function for considering all elements (content loss, style loss, variation loss). The loss equation:

Style Weight

$$\mathrm{L}_{\mathrm{total}}(i,j,k) = \alpha L_{content}(i,k) + \frac{\beta}{N} L_{style}(j,k) + \gamma L_{variation}(k)$$

Total loss

Content image

Style image

Generated image

Content Weight

Number of style layers we used

Variation Weight

2. Model Architecture

The NST technique is based on pre pre-trained VGG19 DL model.
We will extract style from the next layers:

block1_conv1, block2_conv1, block3_conv1, block4_conv1, block5_conv1

And content we will test different content layers every time from these layers:

block5_conv2, block4_conv2, block3_conv4, block3_conv3, block3_conv2

Each iteration we will pass the content Image, style Image and generated image through the model and by our loss function we will apply gradient descent to our generated image to reduce loss.

The total of iteration (epochs and steps per epoch) our model will perform is 1000 total.

3. Elements to experiment

The next elements can be change and we will want to test their effect on our model result:

- α – content weight.
- β – style weight.
- γ – variation weight.

Computer Vision Methods - Second Approach

Using Kanungo noise model

A document degradation model for local distortions that are introduced during the printing, photocopying, and scanning processes.
This model is the base for our model, it helps us select the centers of the degraded region in the neighborhood of the characters.

Kanungo noise model has the following input parameters:
$\Theta = (\eta, \alpha_0, \alpha, \beta_0, \beta)$
η - Is the constant probability of flipping for all pixels.
α_0 - Is the initial value for the exponential in the background probability equation of flipping pixel value.
α - Is the decay speed of the exponential in the background probability equation of flipping pixel value.
β_0 - Is the initial value for the exponential in the foreground probability equation of flipping pixel value.
β - Is the decay speed of the exponential in the foreground probability equation of flipping pixel value.
The steps in the Kanungo noise model:

1. **Binarization**

   The first step of this model is to transform image to grayscale image and then create a binary image with a binarization method. We will use the OTSUmethod (same as the original paper suggests).

2. **Distance Transform**

   The next step is to apply a distance transform method on the binary image. In order to get the distance to the closest boundary (character pixel) from each point.

3. **Probability calculation**

   Calculate the probability for flipping each pixel in the grayscale image this means the probability the background pixel will become foreground pixel and foreground pixel will become background.The probability equation is as follows:

$$p = \begin{cases} \alpha_0 \times e^{-\alpha d^2} + \eta_1 & \text{if P is foreground} \\ \beta_0 \times e^{-\beta d^2} + \eta_0 & \text{if P is background} \end{cases}$$

4. **Seed Threshold**

   We select a threshold for seed selection. Each pixel with a probability bigger or equal to threshold will be a seed for degradation.

The Model We Will Use

A local noise model for grayscale images. Its main principle is to locally degrade the image in the neighbourhoods of “seed-points” selected close to the character boundary. These points define the center of “noise regions”. The pixel values inside the noise region are modified by a Gaussian random distribution to make the final result more realistic. While Kanungo noise models scanning artifacts, our model simulates degradations due to the age of the document itself and printing/writing process such as ink splotches, white specks or streaks. The steps of this model:

1. **Kanungo noise model**

   In order to get seed points to apply local degradation we apply the Kanungo noise model, the pixels that are flipped will be seeds (we don't need to take all of them but we will choose from them).

2. **Noise region**

   For each seed we use to generate noise we will define a region. In this region we will apply our noise. In the original paper the used an ellipsis defined in the following way:

   semi-major axis - defined as a and is equal to $a = a_0 * (1 + \frac{v}{V})$
   semi-minor axis - defined as b and is equal to $b = a * (1 - g)$
   v - is the pixel gradient value.
   V - is the max pixel gradient value.
   a_0 - is an input parameter controlling the size of the noise regions.
   g - g is the flattening factor.

   We will use a rectangle defined the following:

   Half-diagonal - defined as a and is equal to $a = a_0 * (1 + \frac{v}{V})$

3. **Noise generation**

   For each seed we will generate gaussian noise in the defined region.
   In the originale paper for each seed C_i if its a background seed they calculated the average grayscale value of all foreground pixels and if its a foreground seed they calculated the average of all background pixels, the average is marked as $\neg c_i$. Next for each pixel B_j on the edge of the ellipsis
   they calculated the average of its 8-neighbours, marked as $\neg b_j$. for each pixel on the line $C_i B_j$ let's call it P_k we will give it a normal distribution value when σ is a input parameter and u $u = \neg c_i + (\neg b_j - \neg c_i) * \frac{d_{ik}}{d_{ij}}$.
   d_{ik} is the distance from C_i to P_k and d_{ij} is the distance from C_i to B_j.

Software Project Management Plan (SPMP)

1. Project Organization

1.1. Process Model

The project will begin with researching for a proper technique to synthesis style. This includes preprocess, model creation and testing. After the research is done the team will create a basic web based application to show research results in an agile medology.

1.2. Organizational Structure

Name	Role and Responsibilities
Dr. Irina Rabaev	Academic advisor, responsible for leading, assigning duties and ensuring assignments are finished on time.
Yahav Bar David	Student, responsible for development, researching and fulfilling assignments.
Leor Ariel Rose	Student, responsible for development, researching and fulfilling assignments.

2. Managerial Process

2.1. Management Objectives and Priorities

Team will have regular meetings and development processes will be consulted and accepted during these meetings.

Team Meetings - The team will meet twice weekly, once on Sunday and once on Wednesday. Each meeting will focus on planning, strategy and working collaboratively on the project.

Academic Advisor Meetings - Every two weeks on Sunday, the team will meet with their Academic advisor. The team will give a brief oral report of their progress over the last week. The team will submit any documents requiring the academic advisor's approval at these meetings. Academic advisors will consult with the process.

2.2. Assumptions, Dependencies, and Constraints

This section describes the assumptions, dependencies, and constraints that this project is based upon.

- This project will be web-based.
- Technical constraints:
    - No support for legacy browsers.

2.3. Risk Management

The goal of risk management is to identify and mitigate potential sources of expense or delay. Some risks are common to every project phase, and some risks are closely associated with a particular project phase. Risks for this project have been classified accordingly:

Common Risks:

- **Name** - Team Member Unavailability.
  Description - During this project, it is almost certain that some members of the team will be unavailable for certain periods due to illness or emergency.
  Probability - High
  Impact - Low
  Prevention - Team members will alert at the first opportunity regarding potential absences, and coordinate with other members of the team to cover their responsibilities for the duration of the absence.
- **Name** - Miscommunication.
  Description - The volume of communication regarding this project almost guarantees that miscommunications will occur.
  Probability - High.
  Impact - Medium.
  Prevention - The primary method of avoiding miscommunication is to document and verify verbal communications. For this project, the documentation and verification process will consist of meeting agendas, minutes and reports that will be shared.
- **Name** - Changes to Project Scope.
  Description - Changes to project scope are a common request, but can derail project timelines.
  Probability - High
  Impact - High
  Prevention - Every step will be defined and rethinked in order to minimize changes to project scope.

Research Risks:

- **Name** - Misunderstanding of Problem Domain.
  Description - There are many questions to be answered about how to apply document style transfer.
  Probability - High.
  Impact - High.
  Prevention - Reading and expanding our knowledge and every step will be considered and advised by our academic advisor.

- **Name** - Applying Concepts and Techniques Wrongly.
  Description - Document style transfer is a very complex idea and the techniques and mathematics are very confusing and hard to understand. This can cause applying these ideas wrongly.
  Probability - High.
  Impact - High.
  Prevention - Checking and testing every step by all team members.
- **Name** - Uncompatible Resources for Research.
  Description - Resources like GPU and computation power can affect the speed and the progress of the research, causing slower and inefficient research.
  Probability - Medium.
  Impact - Medium.
  Prevention - No prevention can be applied in these circumstances.
- **Name** - Deep Learning Method Failure.
  Description - Our first approach to solve this problem is with a deep learning model. This model can result in bad results.
  Probability - High.
  Impact - Low.
  Prevention - No prevention can be applied but in case this does happen we have a plan b (our second approach).

Software Design Risks:

- **Name** - Incorrect Design.
  Description - Design is the foundation of implementation, and errors in design can create an imperfect project.
  Probability - High.
  Impact - Medium.
  Prevention - Design documents will face internal review within the team and must be approved by all team members.

Software Implementation Risks:

- **Name** - Lack of Experience with Relevant Technologies.
  Description - Team members may not have experience with the technologies used in this project.
  Probability - High.
  Impact - Low.
  Prevention - Work will be assigned according to experience with the relevant technology. Team members are encouraged to ask questions when faced with something they don’t understand. Team members with experience are expected to assist less experienced members. Team members are expected to independently research technologies during the summer break.

Software Testing Risks:

- **Name** - Evaluation and Testing of model.
  Description - One of the most difficult aspects in this project is how to test and evaluate the project.
  Probability - High.
  Impact - Low.
  Prevention - Research will be done to understand various testing and evaluation techniques.

Software Delivery Risks

- **Name** - Incompatible Server Architecture.
  Description - The finished web application will be dependent on a server environment chosen. Such environments vary in their support for web application programming languages and language versions. also this application will need hardware requirements relevant for this project.
  Probability - High.
  Impact - High.
  Prevention - Document necessary requirements and validating the server contains them before deployment.

3. Technical Process

3.1. Methods, Tools and Techniques

The Methods, Tools and Techniques section aims to outline specific plans, methods, or tools to be used during the project.

Diagrams:

Team will use standard UML diagrams to represent data, relationships and requirements.

Programming Languages & Tools:

Team will utilize various programming languages and tools.

For task management:

- teamwork.

For client side development:

- Angular.
- HTML.
- CSS.
- TypeScript.
- Bootstrap
- Nginx

For server side development:

- Python.
- FastAPI.
- Uvicorn.
- Tensorflow.
- Keras.
- OpenCV.

For research:

- Google Colab

For version control:

- Github
- Google Drive

For diagrams:

- draw.io

Techniques:

- Neural Style Transfer.
- Generative Adversarial Network.
- Mathematical, Computer Vision and Deep Learning Techniques.

3.2. Software Documentation

Code will be documented inside the project by team members.

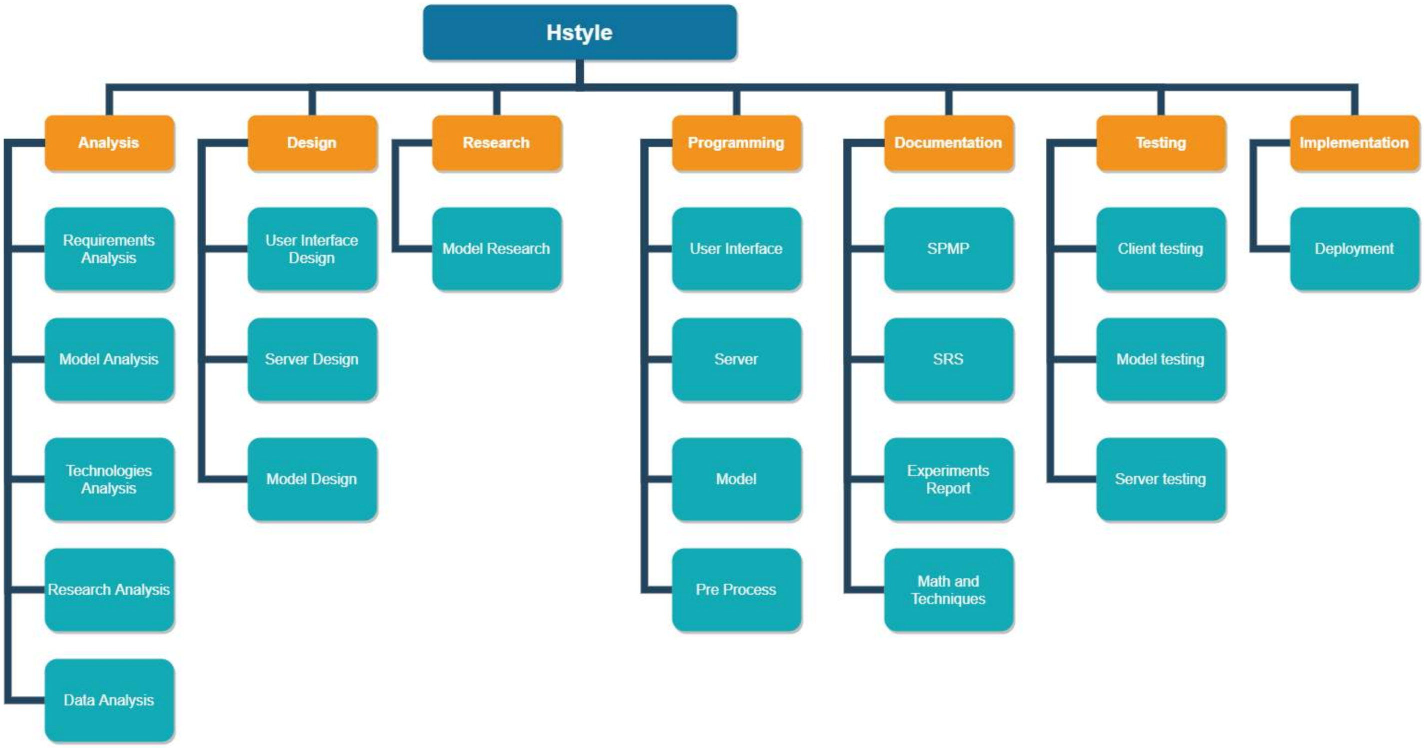
Project documentation can be found in the project's github repository (link included in document references) where every change is logged for version control. Any additional documentation will be in the project's google drive where every change is logged.

Documentation that will be included:

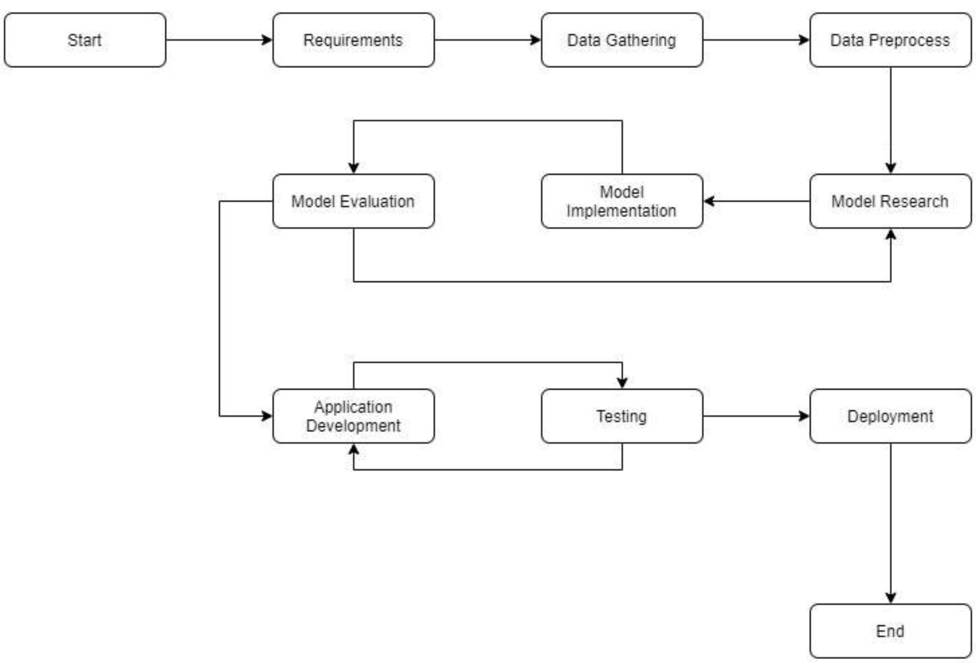
- Software Project Management Plan - Google drive.
- Software Requirements Specification - Google drive.
- Research Techniques - Google Drive.
- Deployment of project - Github.
- Project description - Github.

4. Work Packages, Schedule

4.1. Work Packages



4.2. Dependencies



4.3. Resource Requirement

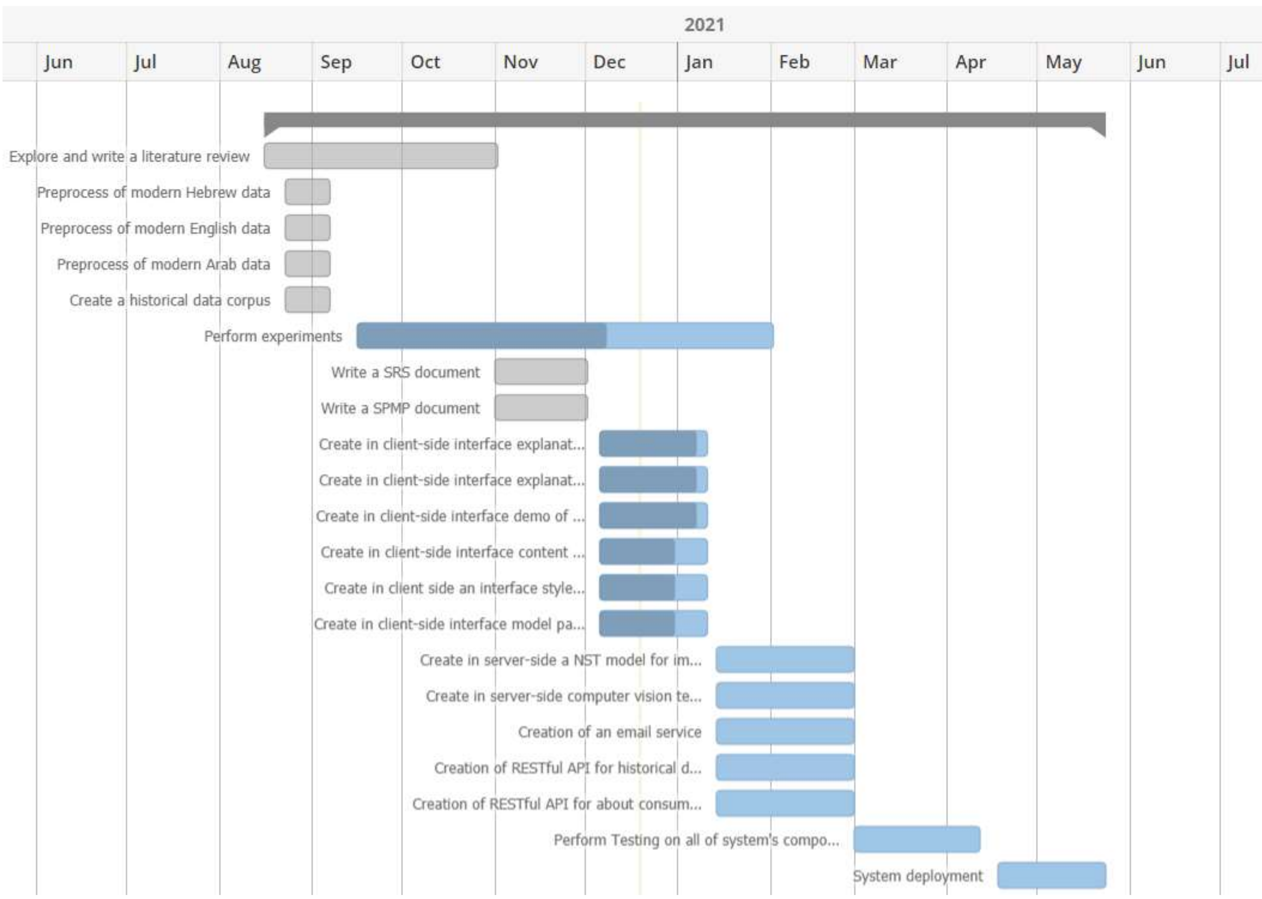
Hardware:

- Two laptops - one for each of the team members.

Software:

- Zoom for meetings.
- Google Colab for experiments.
- Google Drive for documentation and file storage.
- Server for deployment.
- teamwork application.

4.4. Schedule



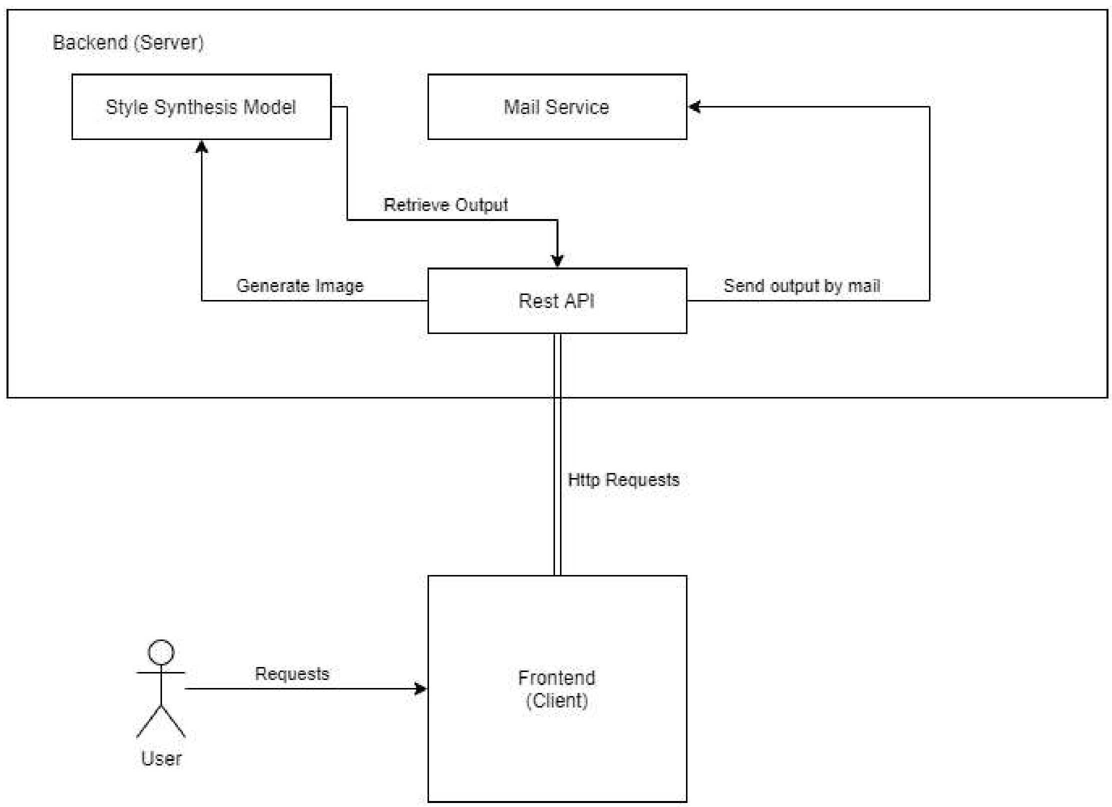
Software Requirements Specification (SRS)

1. Overall Description

1.1. Product Perspective

HStyle is a new, innovative and self-contained full stack application (client-server), with the purpose to let anyone synthesize a historical document. *HStyle* consists of following components:

High level view system diagram



1.2. Product Functions

HStyle enables the end-user to generate a historical style document from one of the appropriate inputs:

- User modern document and a historical document.
- User modern document and one of our historical documents presented in the client side.
- One of our modern documents and one of our historical documents presented on the client side.

Output will be displayed to or sent to user email (by user selection) to spare the user the waiting of the image creation.

1.3. User Classes and Characteristics

HStyle has only one type of user and it is any user that wishes to use the client side of this application to synthesize a historical document.

1.4. Operating Environment

For the user, *HStyle* will operate on the latest versions of Google Chrome, Mozilla Firefox, Microsoft Edge and Opera. Users will be able to use the application using computers and mobile devices.

For deployment, *HStyle* will need a cloud server that able to host a TensorFlow application with these requirements:

- Operating system:
  - Any OS capable of running TensorFlow models. We use a Windows 10 64 bit.
- Central processing unit:
  - Any CPU capable of running a server and TensorFlow models. We use for production an Intel I78750H (9M Cache, up to 4.10 GHz) CPU.
- Graphics processing unit:
  - Any GPU capable of running TensorFlow models. We use for production a Nvidia GeForce GTX 1050 Ti GPU.
- RAM:
  - Enough RAM capable of running a server and TensorFlow models. We use 12GB RAM for production.
- Disk Memory:
  - Enough disk space to contain the project and its runtime. We use 60GB for production.

1.5. Design and Implementation Constraints

- Heavy processing power for running deep neural networks for image style transfer.

1.6. User Documentation

No need for user documentation because *HStyle* is a web application, where all necessary actions will be explained in the application.

1.7. Assumptions and Dependencies

1.7.1. Assumptions

The basic assumption is that the user input images are in the same format as the given images in the client application. That means that the content is a textual image and style is a historical image.

1.7.2. Dependencies

For user usage there are no dependencies. For deployment, the dependencies are included in the GitHub repository (link in references).

2. External Interface Requirements

2.1. User Interfaces

The user interfaces will be a web application. This web application will include the following pages:

- A page to get information on this project, including research ,demo, researchers and more, This page is called an about page.
- A page to apply a historical style synthesis. This page will include the option to upload content and style images and to get other input fields for model parameters. Users will be given the option to select from preexisting images to test the application.

Inputs will be entered via standard web controls such as combo box, check box, text box, calendar, etc. Navigation and acceptance will be handled with buttons.

2.2. Hardware Interfaces

HStyle is a cross platform application and doesn't need any hardware from its users except regular hardware for browsing the web.

2.3. Software Interfaces

HStyle is a cross platform application and doesn't need any software from its users except regular software for browsing the web.

2.4. Communications Interfaces

- The client side will communicate with the server using an HTTP protocol.
- The server side will send output from the DL model by SMTP protocol.

3. System Features

3.1. Historical document synthesis

3.1.1. Description and Priority

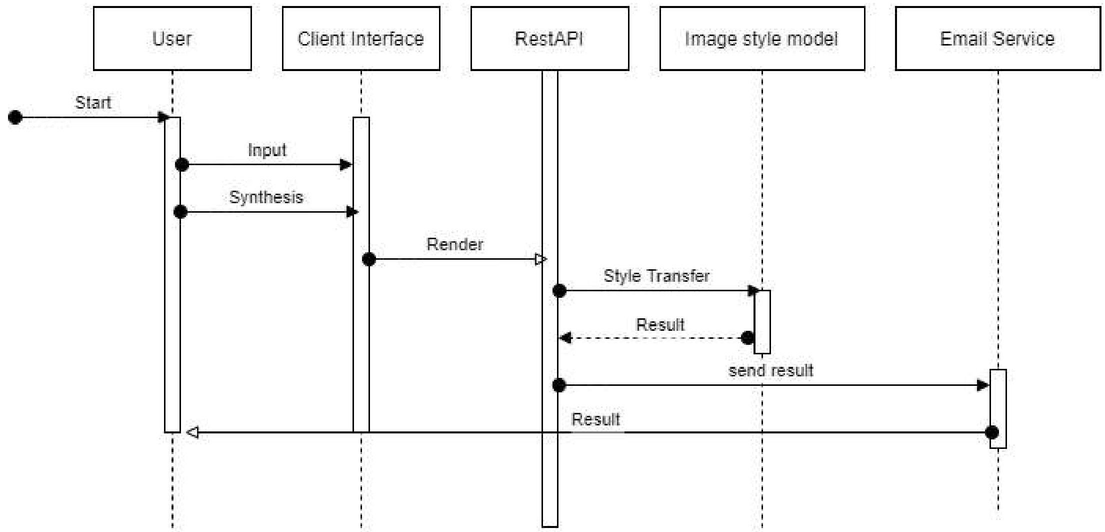
End-user can generate a historical style document from one of the appropriate inputs:

- User’s modern document and a historical document.
- User’s modern document and one of our historical documents presented in the client side.
- One of our modern documents and one of our historical documents presented on the client side.

Output will be displayed to or sent to user email (by user selection) to spare the user the waiting of the image creation.

Priority - high.

3.1.2. *Stimulus/Response Sequences*



3.1.3. *Functional Requirements*

REQ-1 – Preprocess of modern Hebrew data:

- In order to begin the research of creating a model to synthesis style we need to first create the correct data for the process.

REQ-2 - Preprocess of modern English data:

- In order to begin the research of creating a model to synthesis style we need to first create the correct data for the process.

REQ-3 - Preprocess of modern Arab data:

- In order to begin the research of creating a model to synthesis style we need to first create the correct data for the process.

REQ-4 - Create a historical data corpus:

- In order to begin the research of creating a model to synthesis style we need to first create the correct data for the process.

REQ-5 - Create in server-side a NST model for image style transfer:

- In order to synthesize style we first need to create a model that transfers the style appropriately.

REQ-6 - Create in server-side computer vision techniques to improve or change system's results:

- In order to improve or change results we want to enable users to apply computer vision techniques on input images.

REQ-7 – Creation of an email service:

- In order to send the user the model's output and to not make the user wait for the process this system needs to send the output by an email service if the user selects this option.

REQ-8 - Creation of RESTful API for historical document synthesis consumption:

- In order to communicate between users and our model we need a RESTful API.

REQ-9 - Create in client-side interface content image selection and image upload section:

- In order to get a content image from a user we need to create an option to upload an image or to select from given images.

REQ-10 - Create in client-side interface style image selection and image upload section:

- In order to get a style image from a user we need to create an option to upload an image or to select from given images.

REQ-11 - Create in client-side interface model parameters selection section:

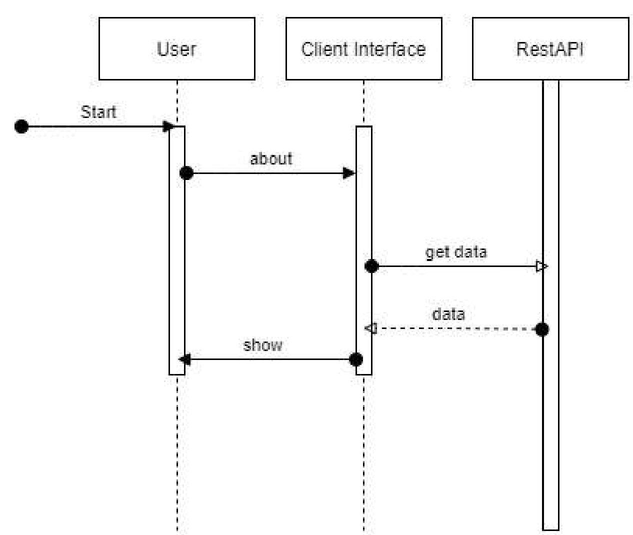
- In order to get a user's parameters to apply on model we need to create an option to select options.

3.2. About

3.2.1. Description and Priority

End-users can get information about the system including what the system is, details about the research, details about the researchers and more.

3.2.2. Stimulus/Response Sequences



3.2.3. Functional Requirements

REQ-12 - Create in client-side interface explanation section on *HStyle*:

- In order to give user information about the system we will create a section with system explanation.

REQ-13 - Create in client-side interface demo of *HStyle*:

- In order to show the user system's result we will create a section with system demo.

REQ-14 - Create in client-side interface explanation section of *HStyle*'s researchers:

- In order to give user information about the system researchers we will create a section with system researchers information.

REQ-15 - Creation of RESTful API for about consumption:

- This system includes a RESTful API in order to send the needed data to the client side.

4. Other Non-functional Requirements

4.1. Performance Requirements

The system must be interactive and the delays involved must be less. So, in every action-response of the system, there are no immediate delays.
Because the act of image style transfer is style consuming, the model should be optimized and output will be displayed to or sent to user email (by user selection) to spare the user the waiting of the image creation.

4.2. Safety Requirements

Software shall provide error handling to support functionality, software shall provide fault containment mechanisms to prevent error propagation. Software termination shall result in a safe system state.

4.3. Security Requirements

No security requirements.

4.4. Software Quality Attributes

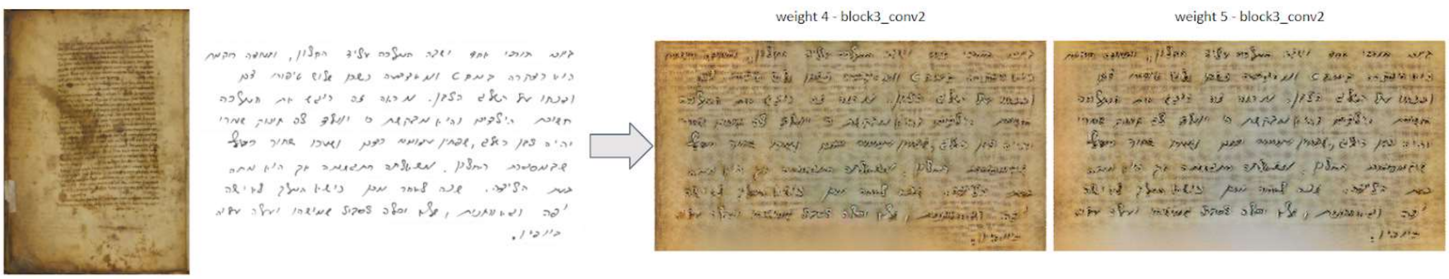
Software quality requirements for this platform are:

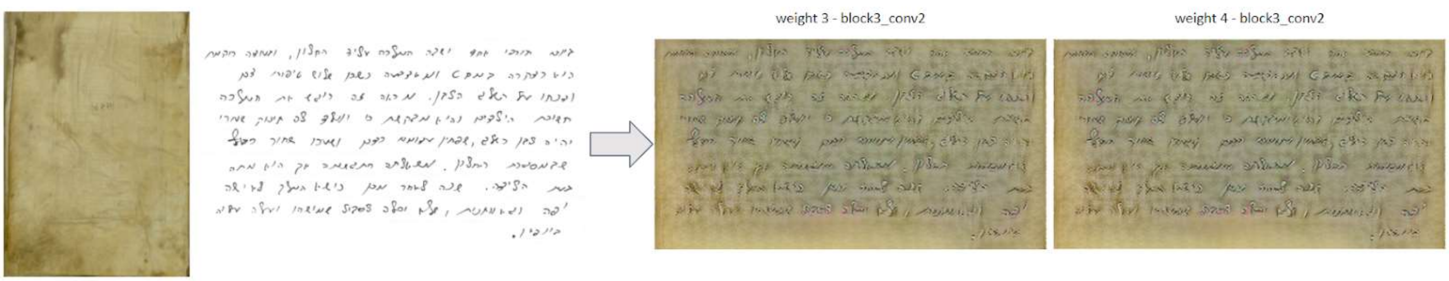
- Availability - The system will be available to access.
- Usability - The system is easy to handle and navigates in the most expected way with no delays. In that case the system program reacts accordingly and transverses quickly between its states.
- Performance – Performance of functions will be tested.
- Compatibility - Any additional feature that will be added should peacefully coexist with existing features.

Results and conclusion

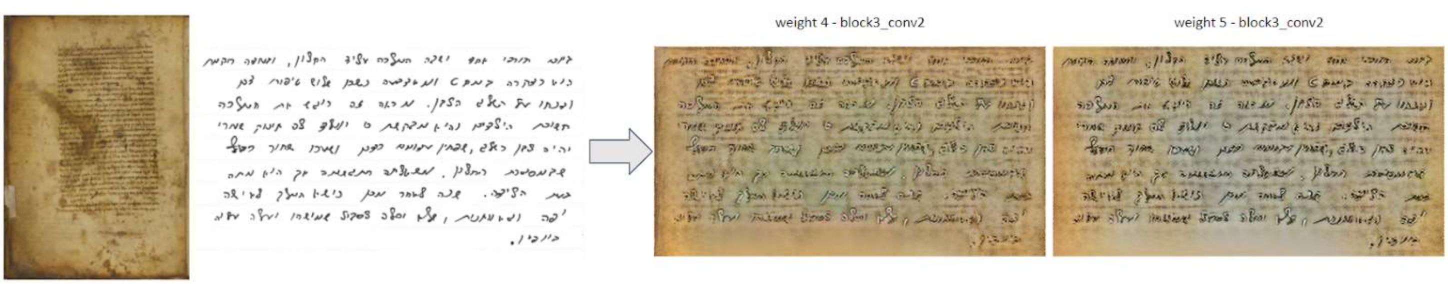
In our research, we perform multiple experiments in order to get the best results. The examples below show a sample of our experiments:

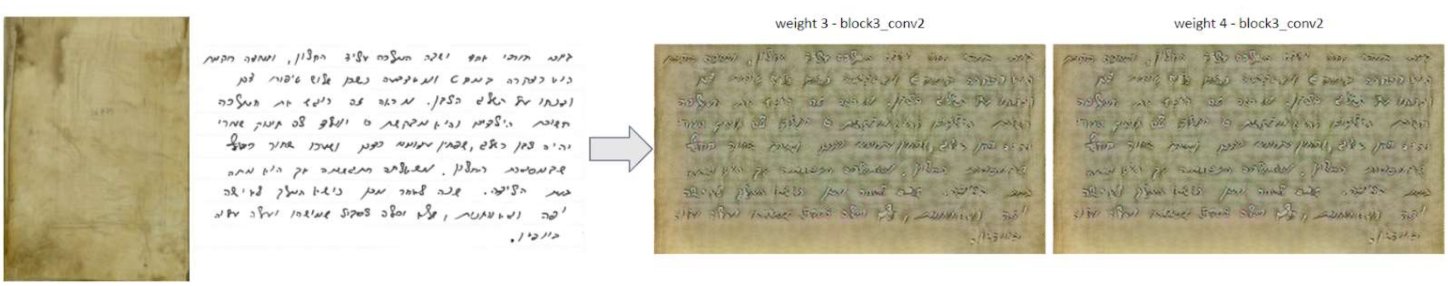
1. Original content image



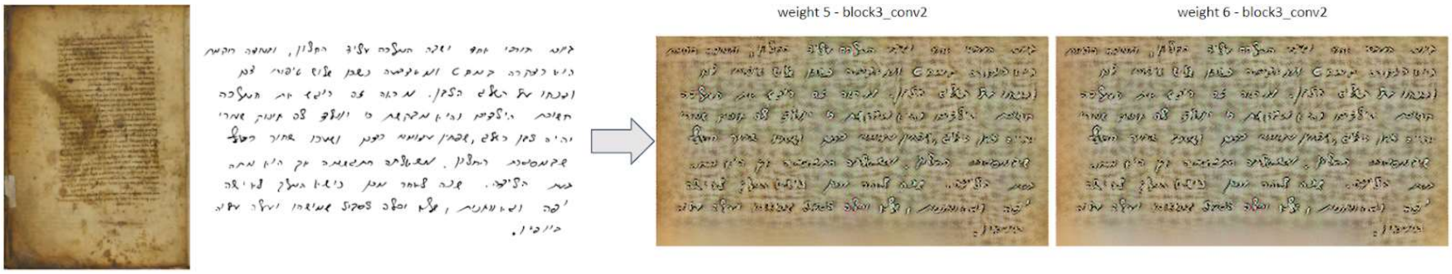


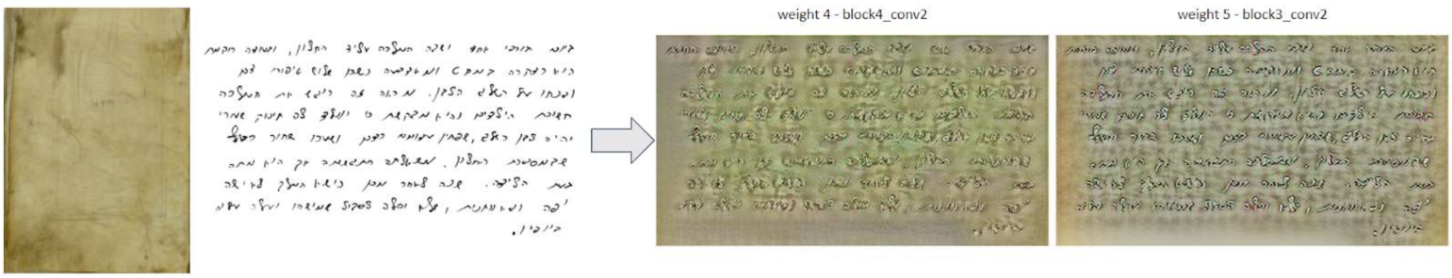
2. Dilate content image



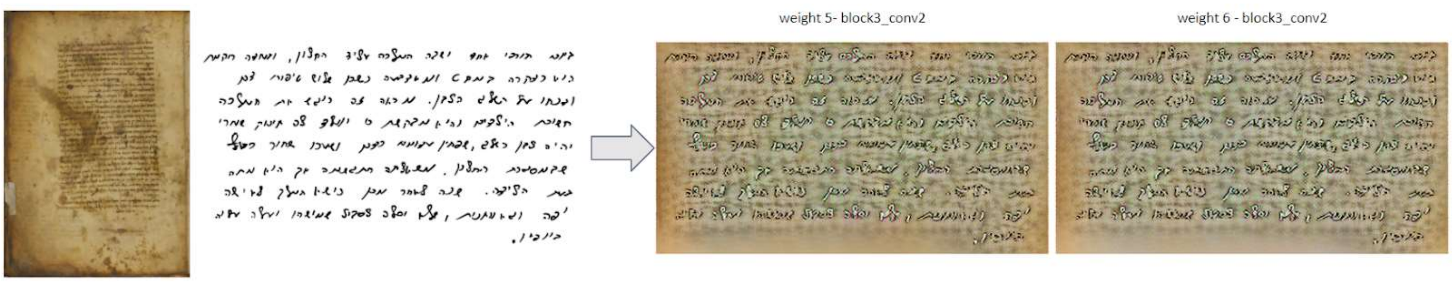


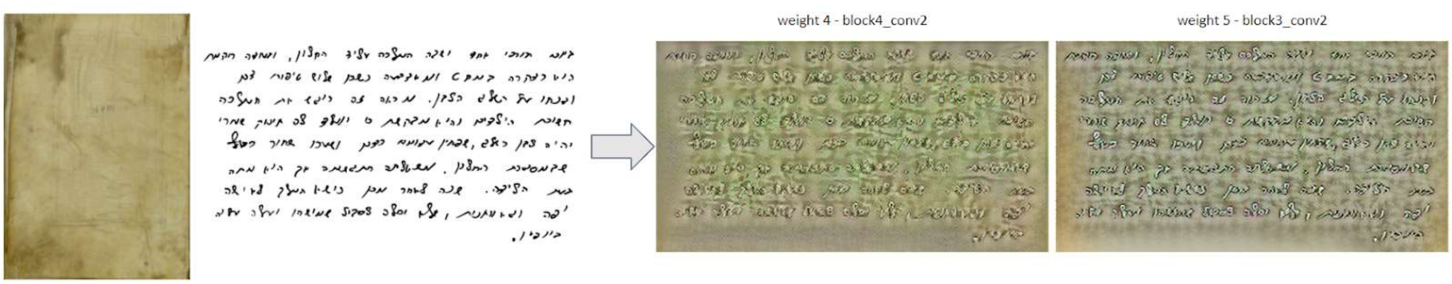
3. Binary content image



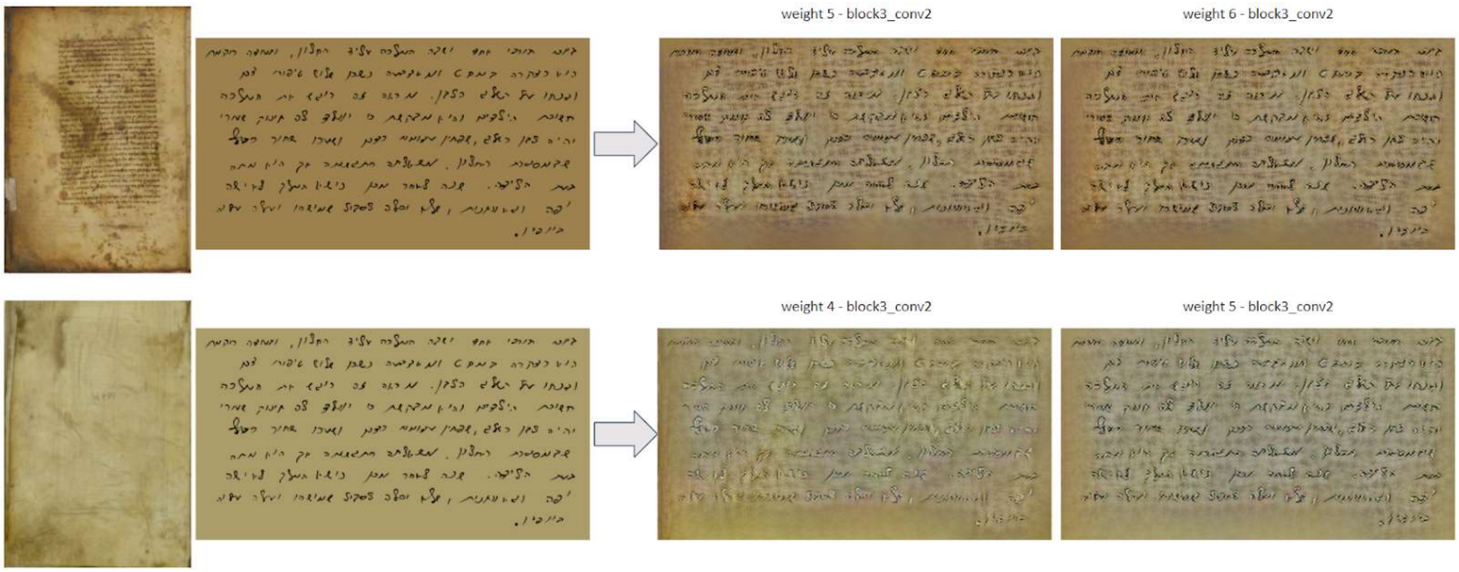


4. Binary-Dilate content image





5. Average historical pixels background for content image



As we can see, we get the best results with a dilated content image. When we apply binarization on a content image, it ruins the results. Also, when we tried to change the content image background color to the average pixels from the style image, it did not improve our results. So to conclude, we managed to transfer historical style on modern content image and get good results.

References

[1] L. A. Gatys, A. S. Ecker and M. Bethge, "A Neural Algorithm of Artistic Style," *CoRR,* 2015.

[2] L. A. Gatys, A. S. Ecker and M. Bethge, "Image Style Transfer Using Convolutional Neural Networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2414-2423.

[3] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama and T. J. Darrell, "Caffe: Convolutional Architecture for Fast Feature Embedding," in *Proceedings of the 22nd ACM International Conference on Multimedia*, New York, NY, USA, Association for Computing Machinery, 2014, p. 675–678.

[4] A. Ter-Sarkisov, "Network of Steel: Neural Font Style Transfer from Heavy Metal to Corporate Logos," in *Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, Valletta, Malta, 2020, pp. 621-629.

[5] G. Atarsaikhan, B. K. Iwana, A. Narusawa, K. Yanai and S. Uchida, "Neural Font Style Transfer," in *14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Kyoto, Japan, 2017, pp. 51-56.

[6] R. Gomez, A. F. Biten, L. Gomez, J. Gibert, D. Karatzas and M. Rusiñol, "Selective Style Transfer for Text," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, Sydney, Australia, 2019, pp. 805-812.

[7] M. Guan, H. Ding, K. Chen and Q. Huo, "Improving Handwritten OCR with Augmented Text Line Images Synthesized from Online Handwriting Samples by Style-Conditioned GAN," in *17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, Dortmund, Germany, 2020.

[8] L. Kang, P. Riba, M. Rusiñol, A. Fornés and M. Villegas, "Distilling Content from Style for Handwritten," in *17th International Conference on Frontiers in Handwriting Recognition (ICFHR).*, Dortmund, Germany, 2020.

[9] H. Liu and A. Zhu, "Synthesizing Scene Text Images for Recognition with Style Transfer," in *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, Sydney, Australia, 2019, pp. 8-13.

[10] M. Mayr, M. Stumpf, A. Nikolaou, M. Seuret, A. Maier and V. Christlein, "Spatio-Temporal Handwriting Imitation," 2020.

[11] B. Davis, C. Tensmeyer, B. Price, C. Wigington, B. Morse and R. Jain, "Text and Style Conditioned GAN for Generation of Offline Handwriting Lines," 2020.

[12] V. Pondenkandath, M. Alberti, M. Diatta, R. Ingold and M. Liwicki, "Historical Document Synthesis with Generative Adversarial Networks," in *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, Sydney, Australia, 2019, pp. 146-151.

[13] V. C. Kieu, M. Visani, N. Journet, J. P. Domenger and R. Mullot, "A character degradation model for grayscale ancient document images," in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR)*, Tsukuba, Japan, 2012, pp. 685-688.

[14] D. T. Luyen, E. Carel, J. M. Ogier and J. C. Burie, "A Character Degradation Model for Color Document Images," in *13th International Conference on Document Analysis and Recognition (ICDAR)*, Tunis, Tunisia, 2015, pp. 806-810.

[15] GitHub repository of *HStyle*.

[16] "IEEE Standard for Software Project Management Plans," IEEE Std 1058-1998, pp. 1-28, 1998.

[17] "IEEE Guide for Software Requirements Specifications," IEEE Std 830-1984, pp. 1-26, 1984.

Appendix A - Experiments Analysis

1. Fast arbitrary image style transfer

Our first attempt in order to see if an images style approach can give good results was to try the fast image style transfer model. This model is pre trained that performs fast artistic style transfer that may work on arbitrary painting styles. In this model we cannot control the content and style weights but only give a content and style image.

1.1. Experiment 1

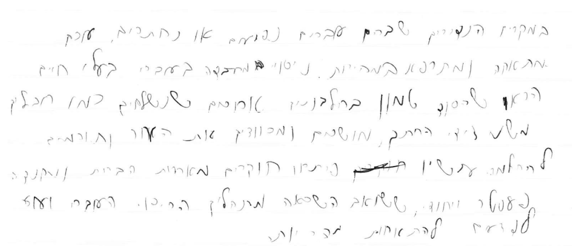
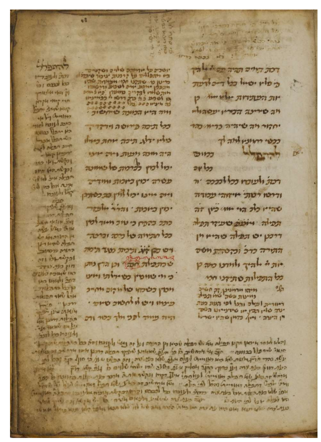
1.1.1. Content Input

This experiment used a modern hebrew handwritten document for content.

1.1.2. Style Input

This experiment used a hebrew middle age document with text for style.

1.1.3. Results

Content image	Style image
	
result	
	

1.1.4. *Discussion*

As we can see the results are not good, we get the style of the document but we also get a lot of the text in patterns.

1.2. Experiment 2

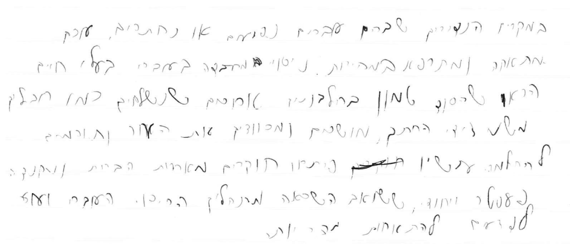
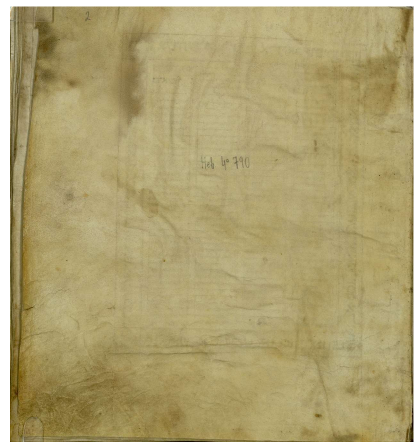
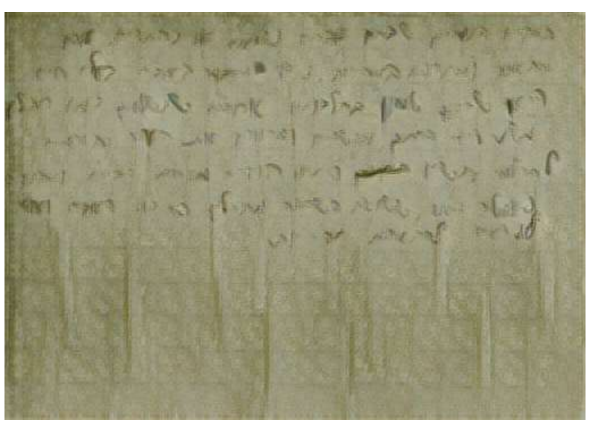
1.2.1. *Content Input*

This experiment used a modern hebrew handwritten document for content.

1.2.2. *Style Input*

This experiment used a hebrew middle age document without text for style.

1.2.3. *Results*

Content image	Style image
	
result	
	

1.2.4. *Discussion*

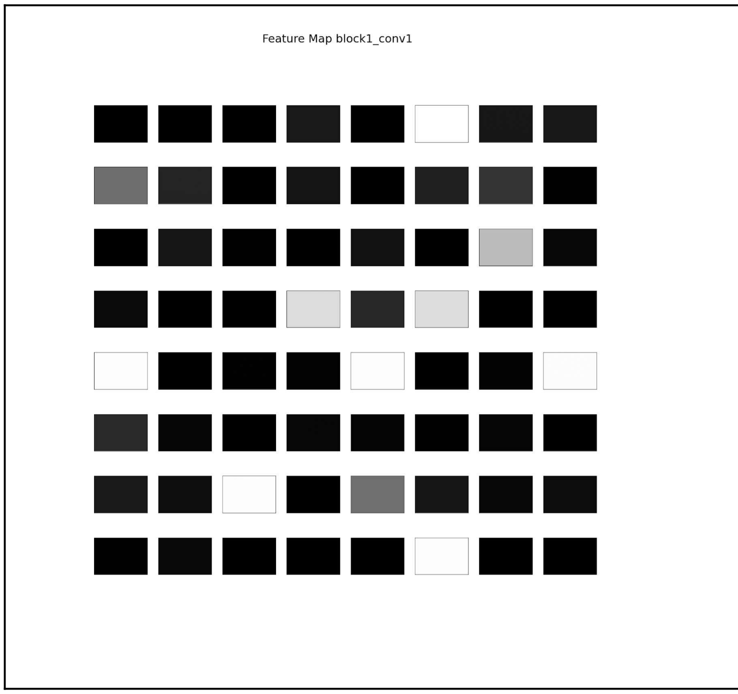
As we can see the results are better but still they are not very good, we get a style but it's not really the document style and text is a little unreadable.

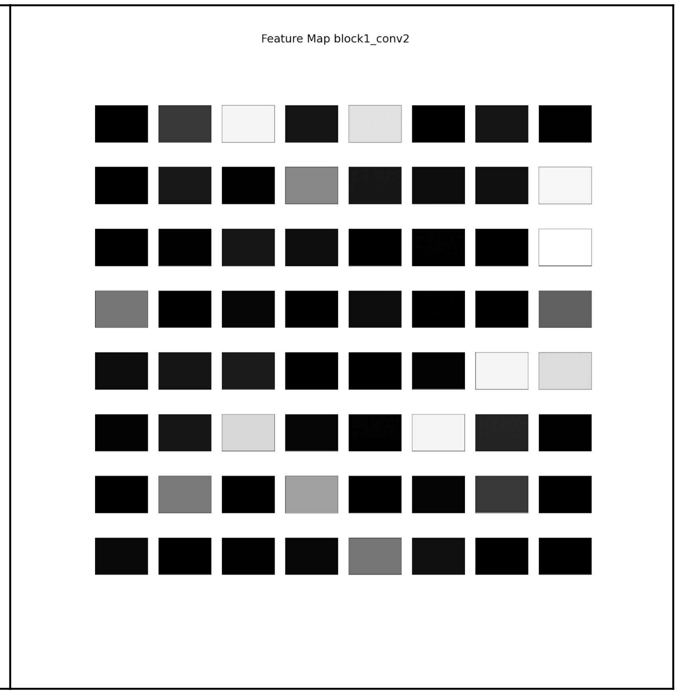
2. Neural Style Transfer

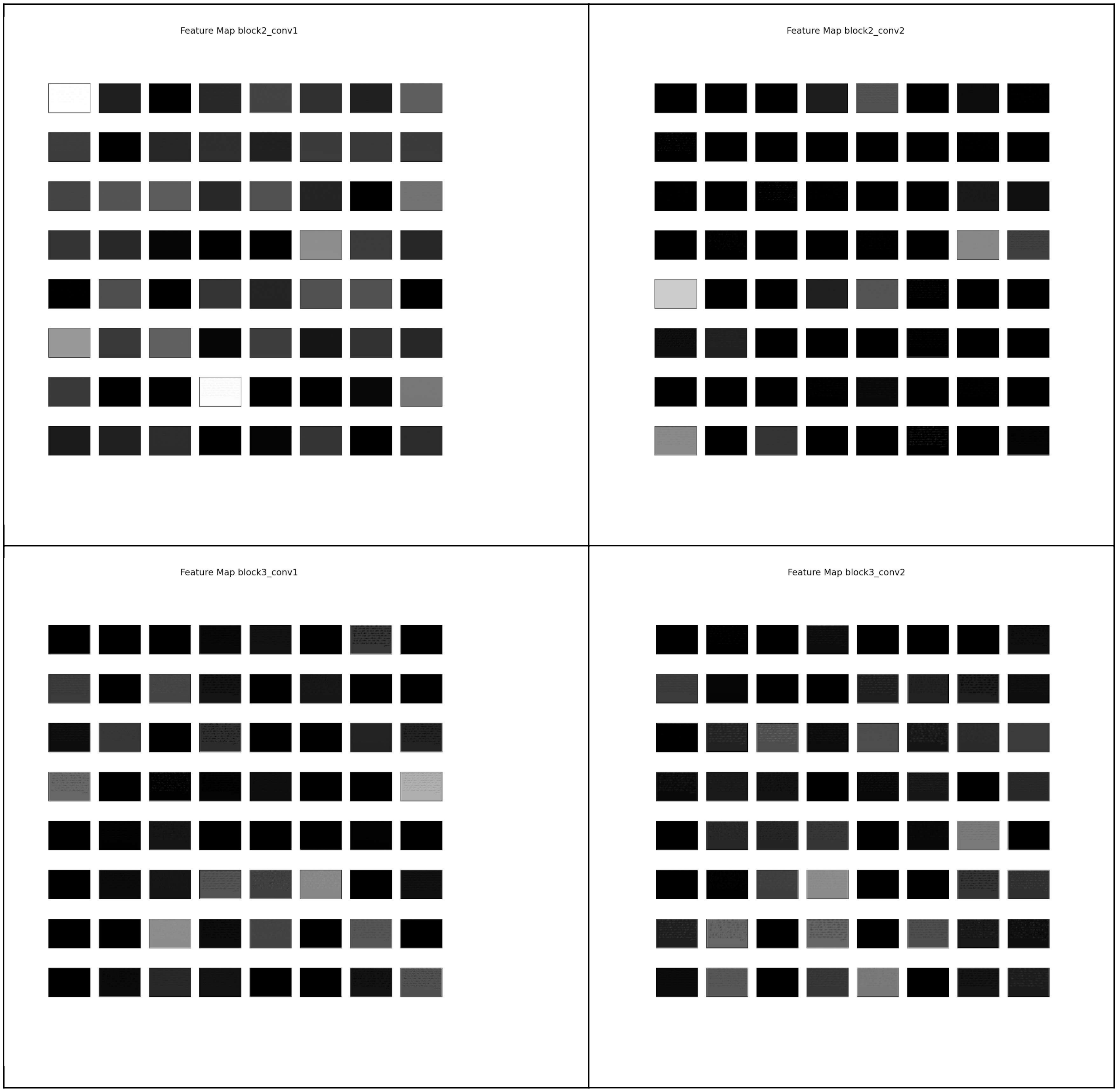
After we saw in the fast arbitrary image style transfer that we can get a decent result but we need more control on the parameters we decided to try the Neural style transfer technique using a deep learning model. We experimented on different content layers and weights.

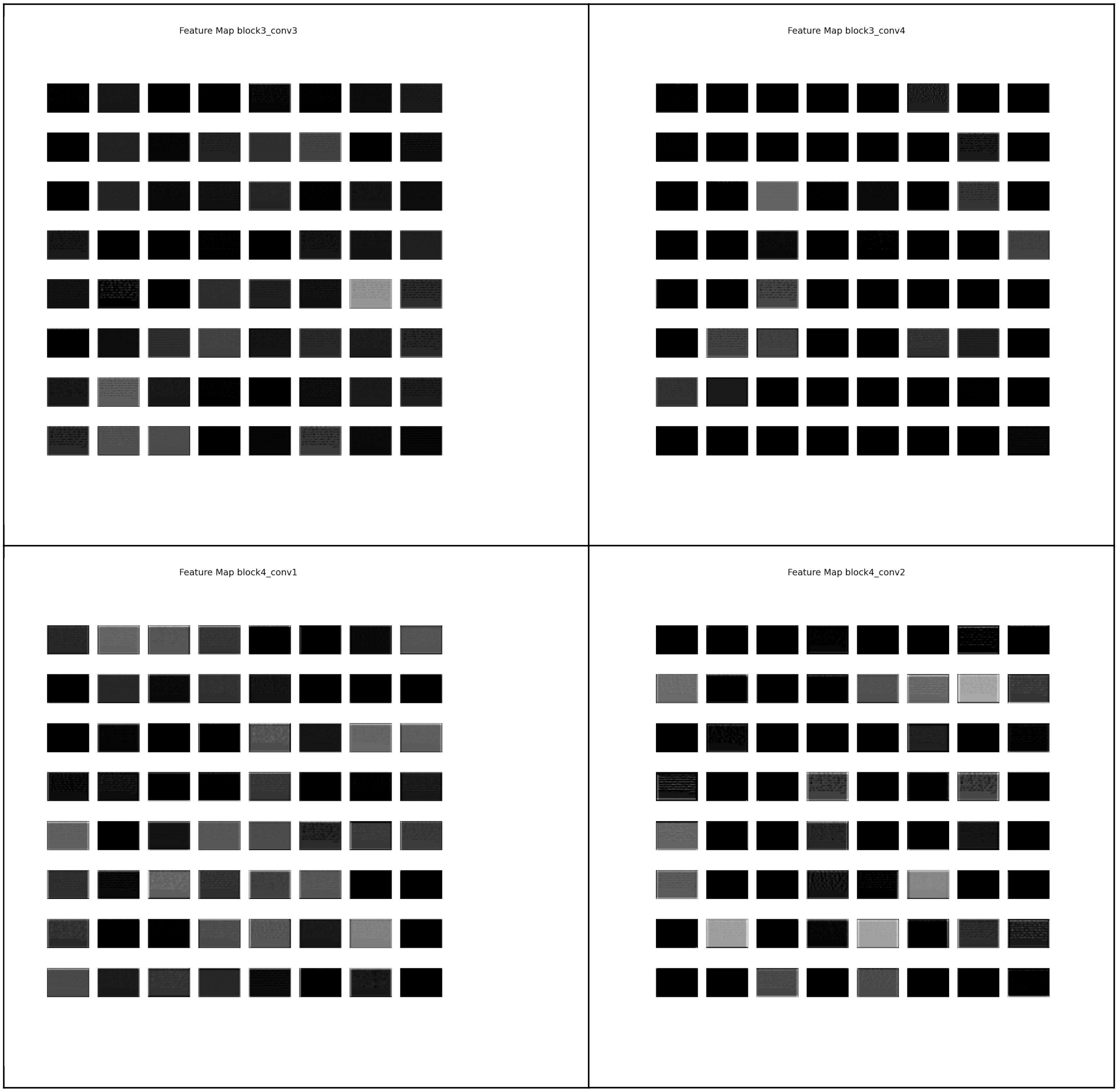
2.1. Content layers

We visualized VGG19 layers feature maps in order to understand where we should take content. we visualized the output of relevant intermediate layers. The results are the following (pay attention that some layers have more than 64 feature maps but we displayed only 64 to get the intuition):

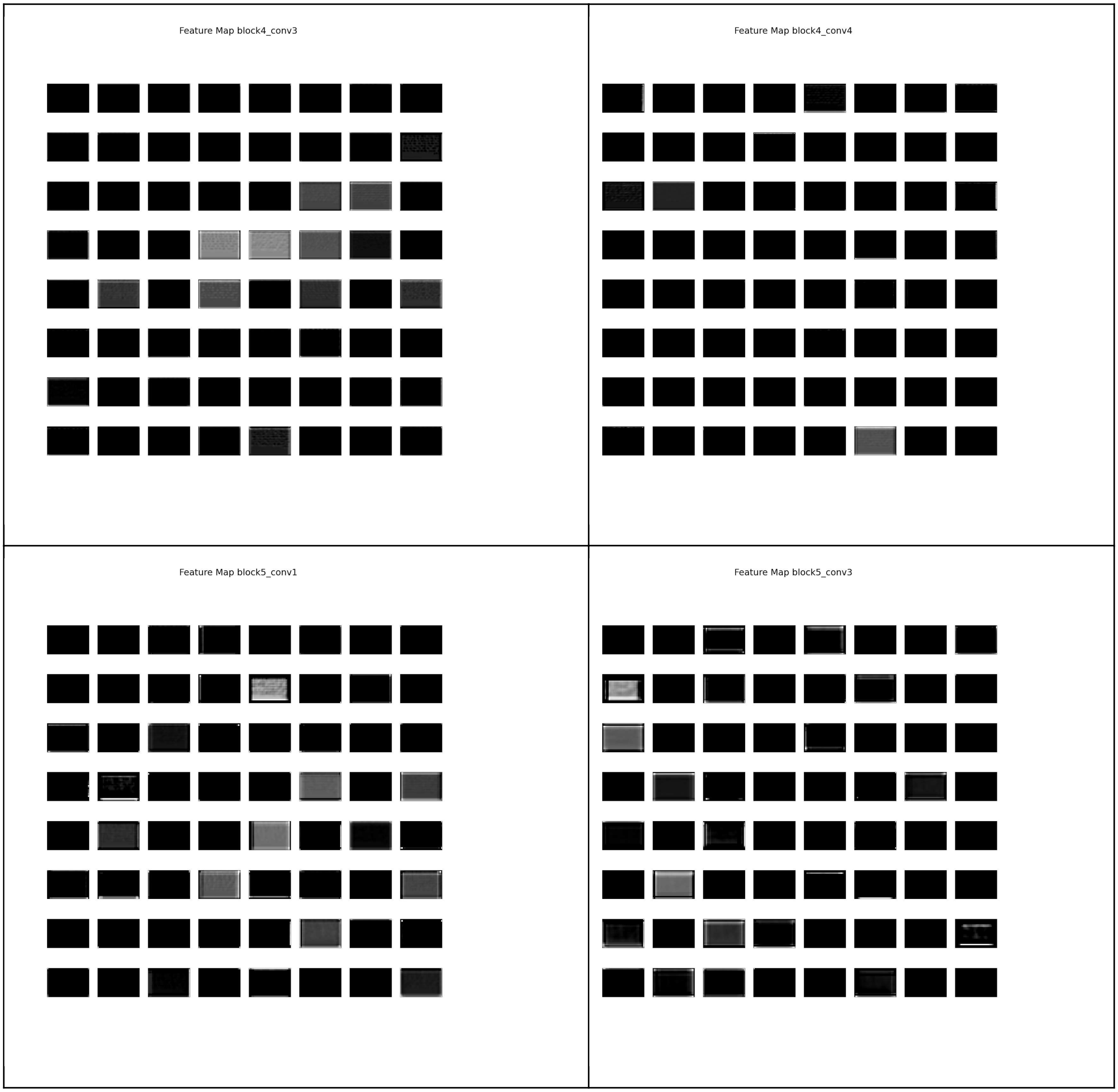





Feature Map block2_conv1
Feature Map block2_conv2
Feature Map block3_conv1
Feature Map block3_conv2



Feature Map block3_conv3
Feature Map block3_conv4
Feature Map block4_conv1
Feature Map block4_conv2



As we can see these layers are best to extract content from:
block4_conv2, block3_conv4, block3_conv3, block3_conv2
In addition we will test **block5_conv2** because it was the best in the original paper.

2.2. Style layers

We used the original paper style layers:
block1_conv1, block2_conv1, block3_conv1, block4_conv1, block5_conv1

2.3. Weights

In each experiment we used the following Weights:

Index	Content weight (α)	Style weight (β)	Ratio (α/β)
0	1.00E+01	1.00E-02	1.00E+03
1	1.00E+02	1.00E-02	1.00E+04
2	1.25E+02	1.00E-02	1.25E+04
3	1.50E+02	1.00E-02	1.50E+04
4	1.75E+02	1.00E-02	1.75E+04
5	1.00E+03	1.00E-02	1.00E+05
6	1.50E+03	1.00E-02	1.50E+05
7	1.00E+04	1.00E-02	1.00E+06
8	1.00E+05	1.00E-02	1.00E+07

2.4. Experiment 1

2.4.1. Content Input

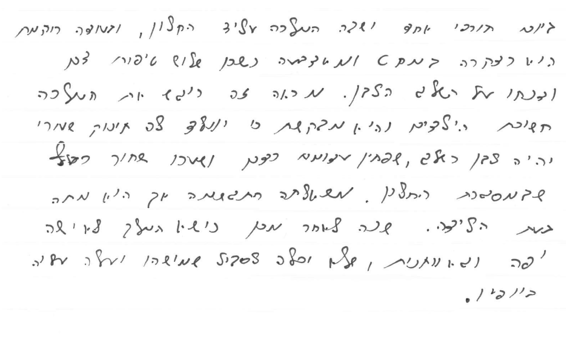
This experiment used a modern hebrew handwritten document for content.

2.4.2. Style Input

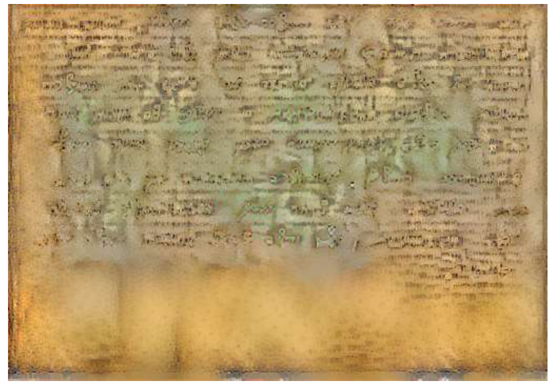
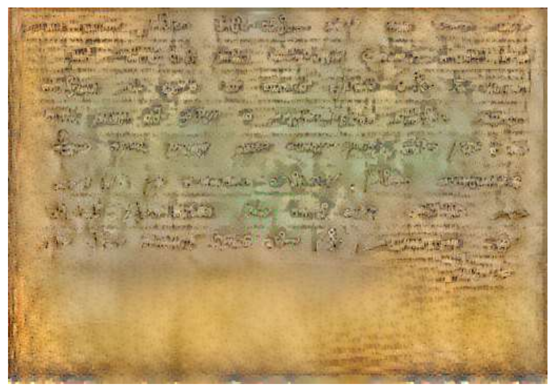
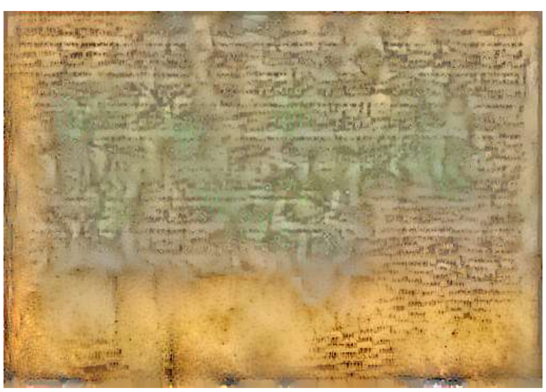
This experiment used a hebrew middle age document with text for style.

2.4.3. Results

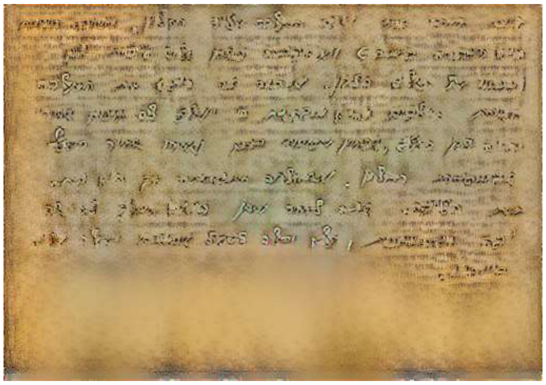
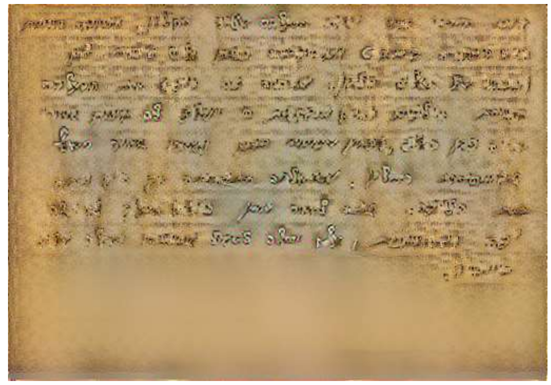
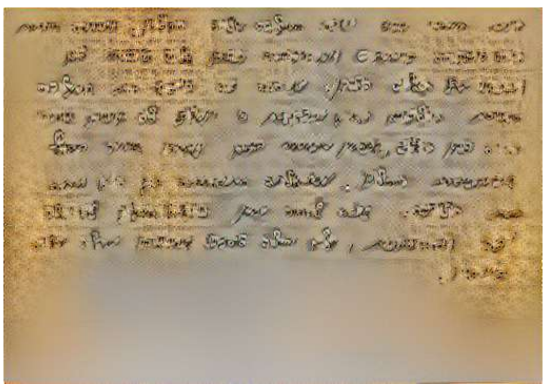
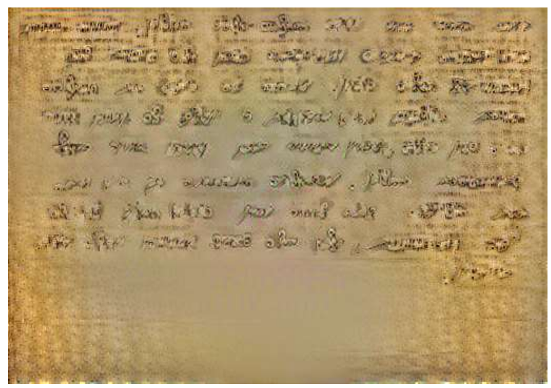
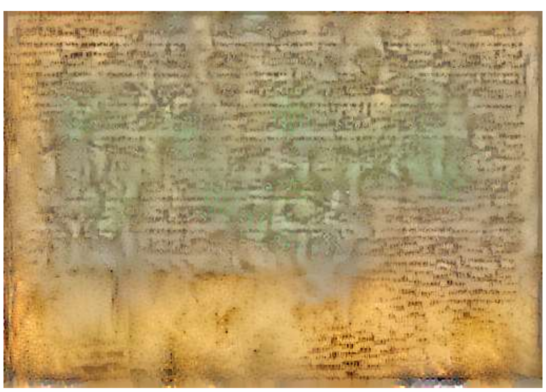
We will show samples of the results with each weight and content layer (style layers are the same).

Content image	Style image
	

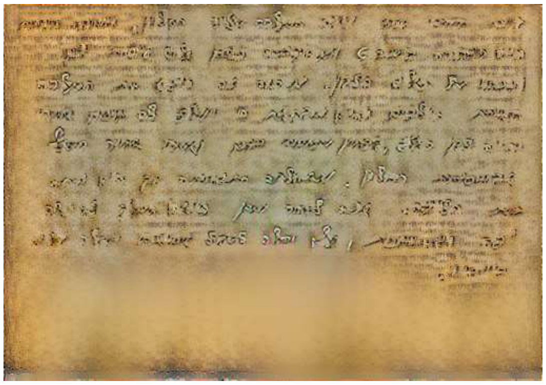
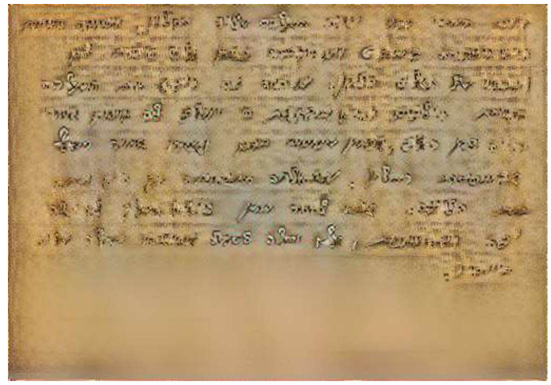
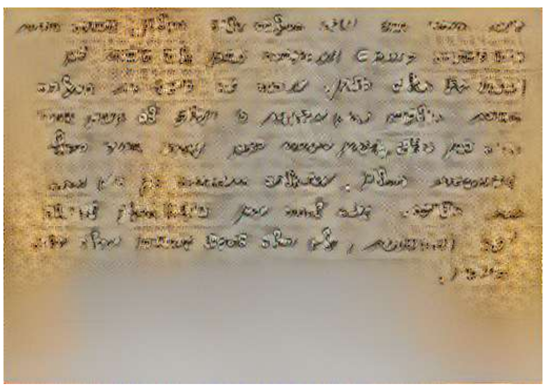
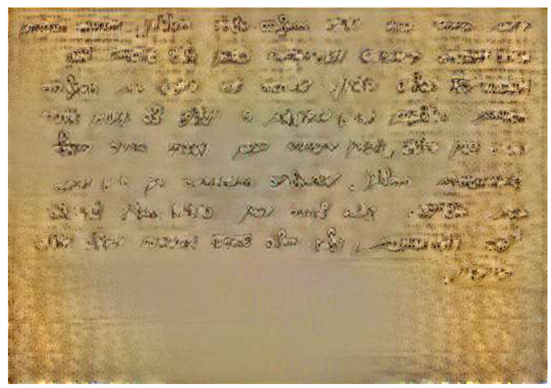
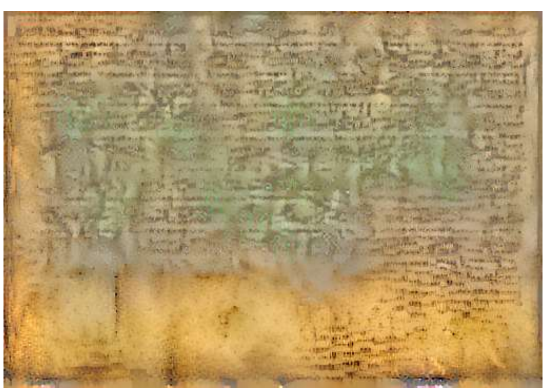
2.4.3.1. Weight 0

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

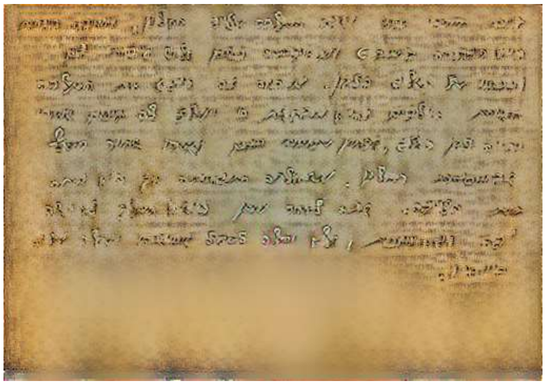
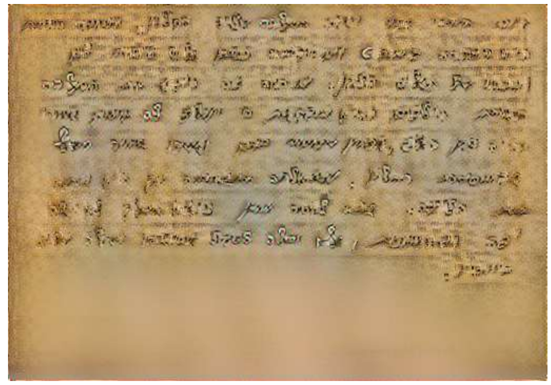
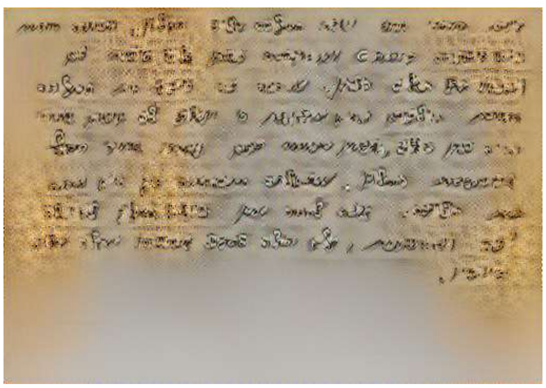
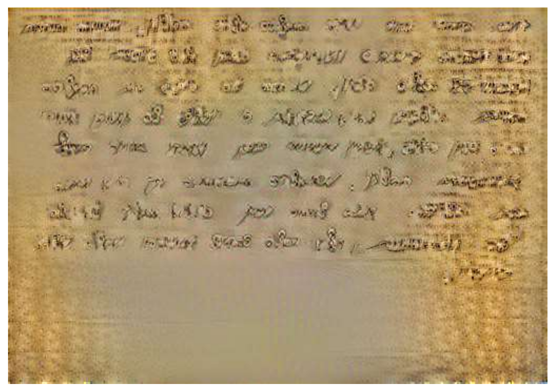
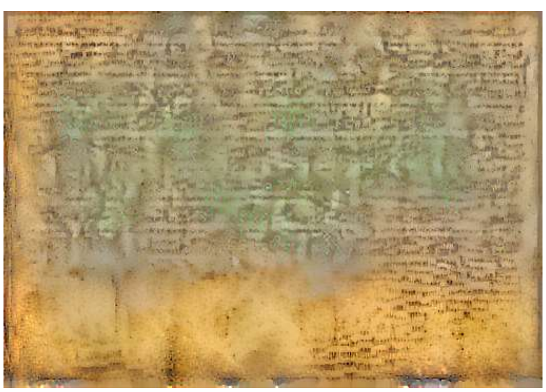
2.4.3.2. Weight 1

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

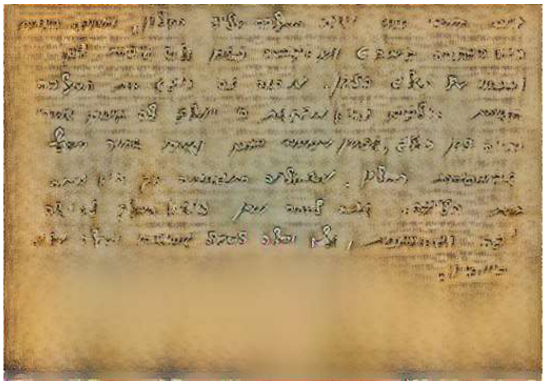
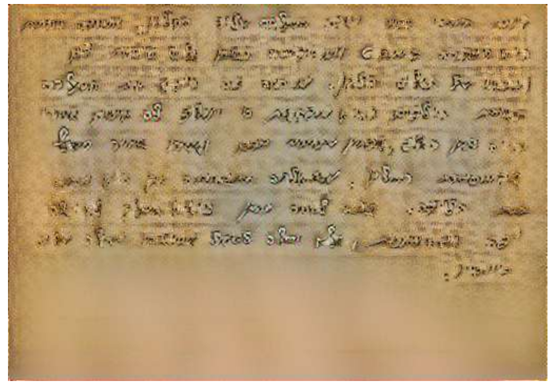
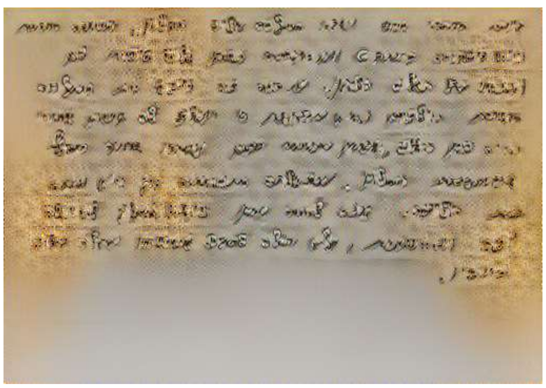
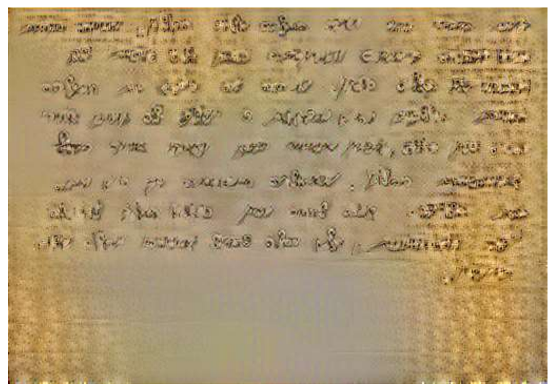
2.4.3.3. Weight 2

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

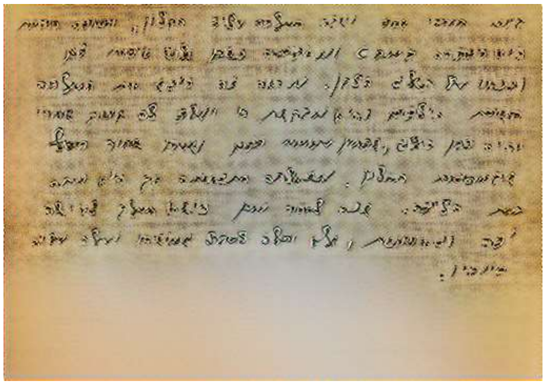
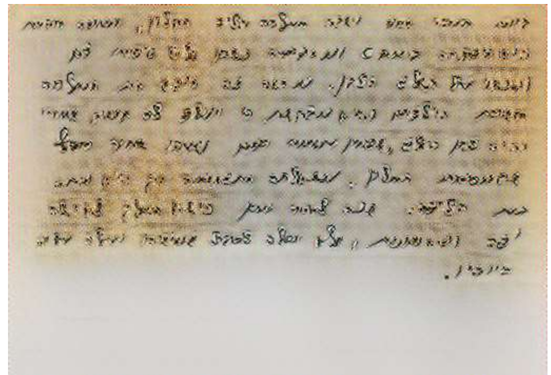
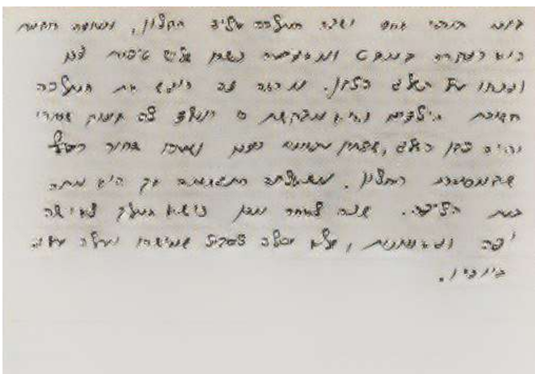
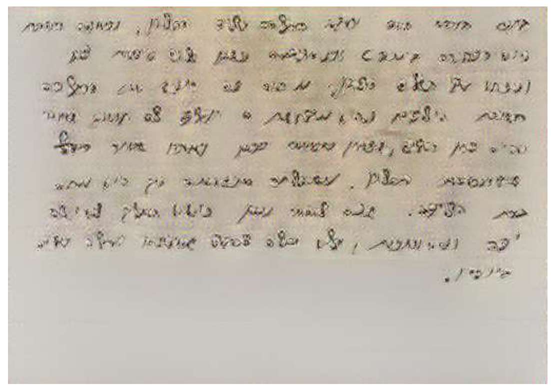
2.4.3.4. Weight 3

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

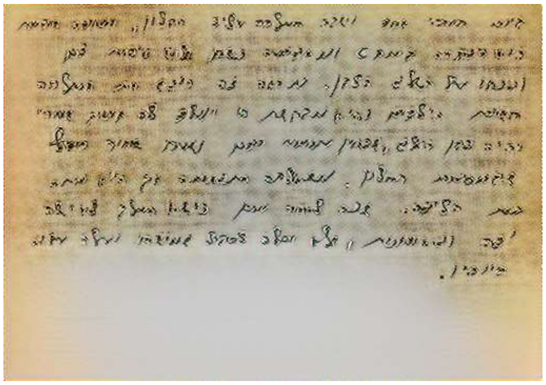
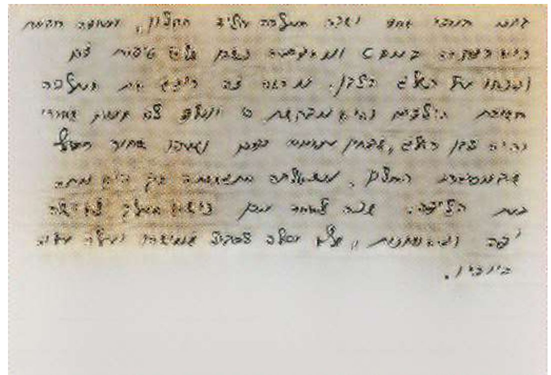
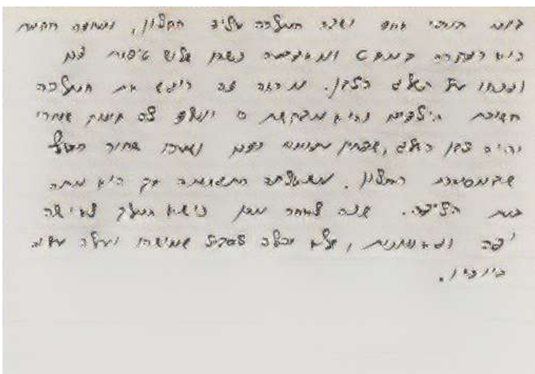
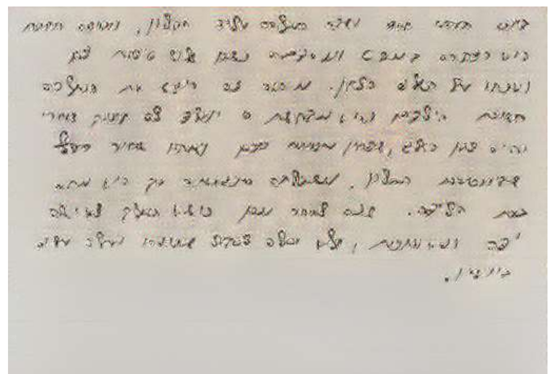
2.4.3.5. Weight 4

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

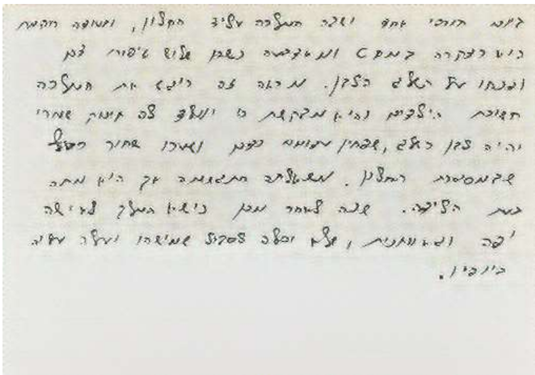
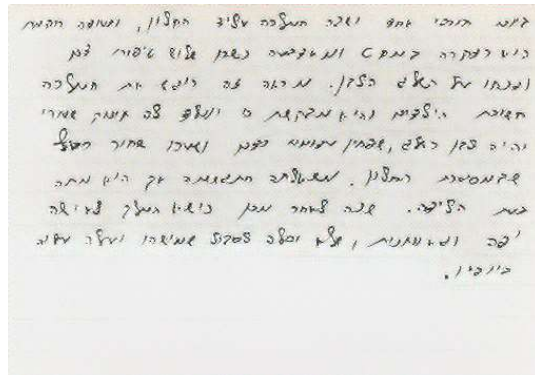
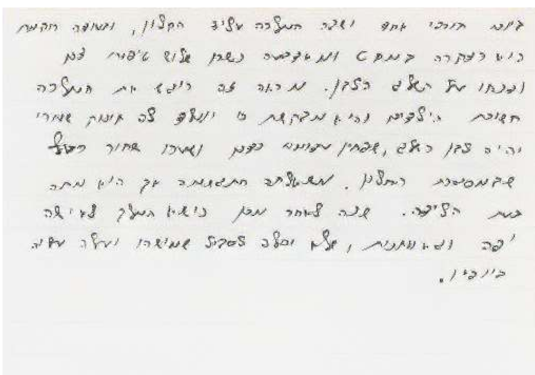
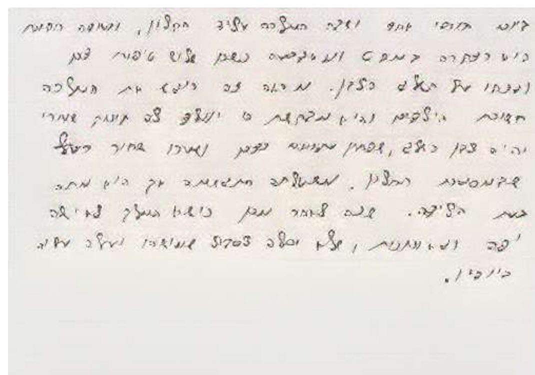
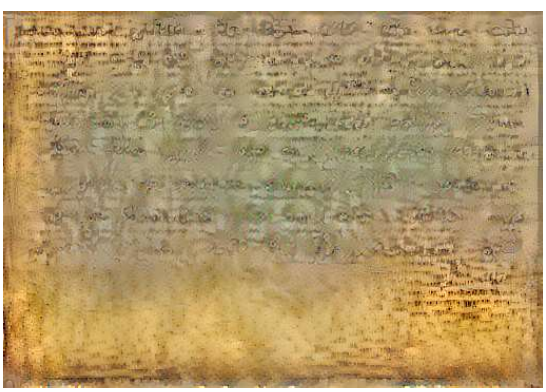
2.4.3.6. Weight 5

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

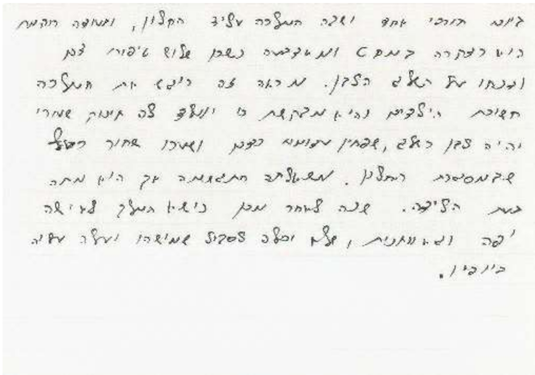
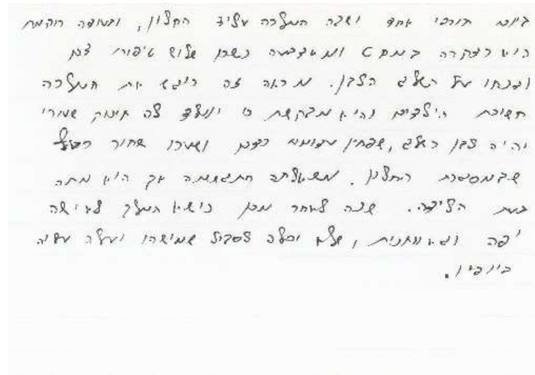
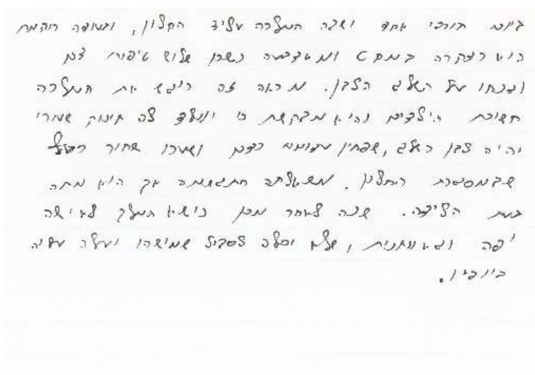
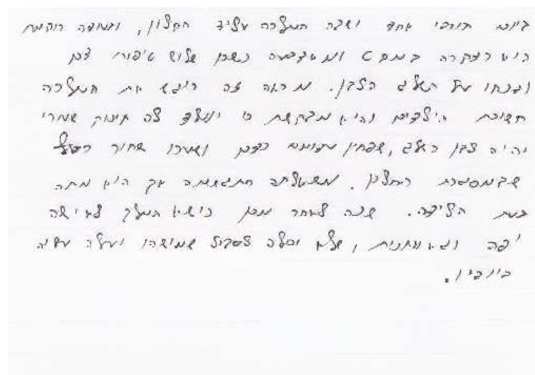
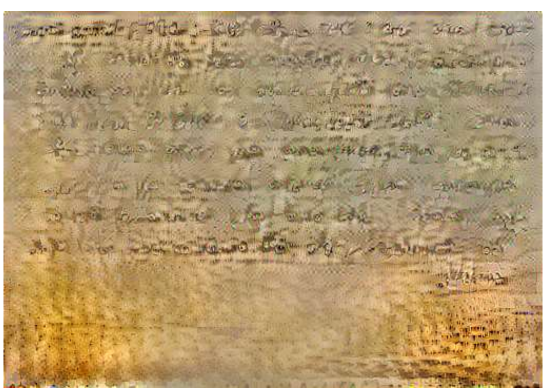
2.4.3.7. Weight 6

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.4.3.8. Weight 7

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.4.3.9. Weight 8

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.4.4. Discussion

As we can see, all content layers give good results except block5_conv2, content layer block3_conv2 gives best results for this experiment. we can see if content weight is too high we don't get a good style representation (weights 5-8) and if content weight is too low we don't get a good content representation (weight 0) . Also the text is a little blurry but overall decent results.

2.5. Experiment 2

We saw good results in experiment 1 but the text is a little bit blurry, in order to improve these results we want to test the effect of text in the style image. because the style image contains text it may disrupt the content image text.

2.5.1. Content Input

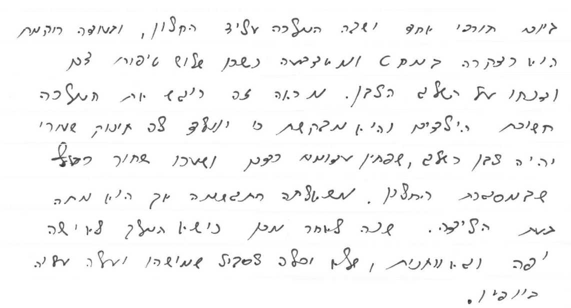
This experiment used a modern hebrew handwritten document for content.

2.5.2. Style Input

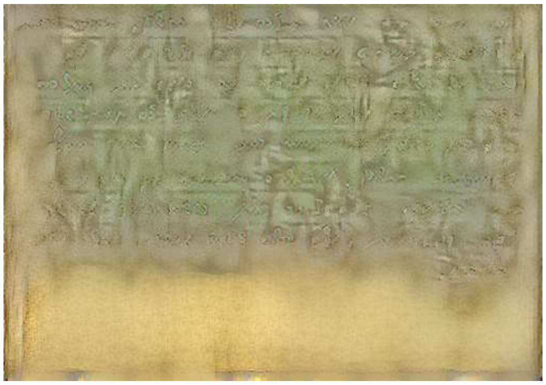
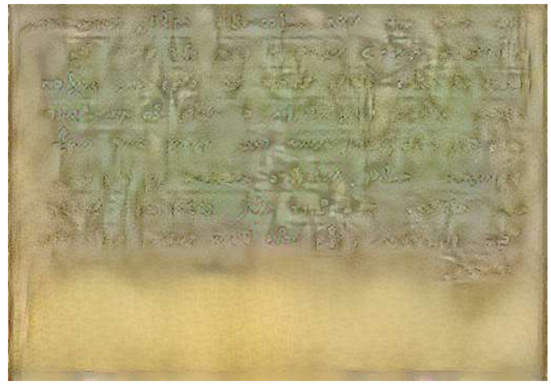
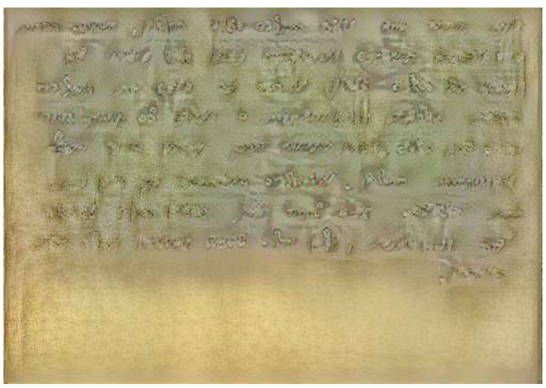
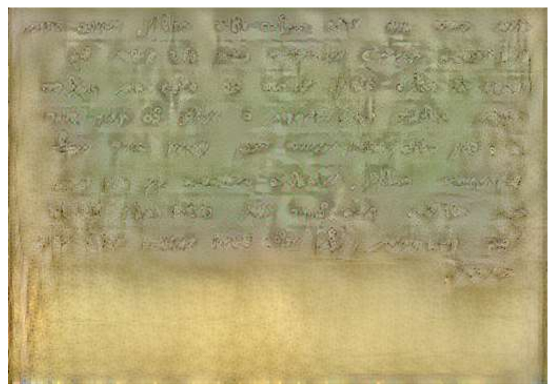
This experiment used a hebrew middle age document without text for style.

2.5.3. Results

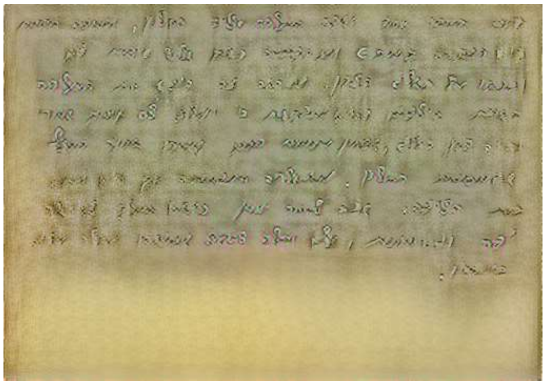
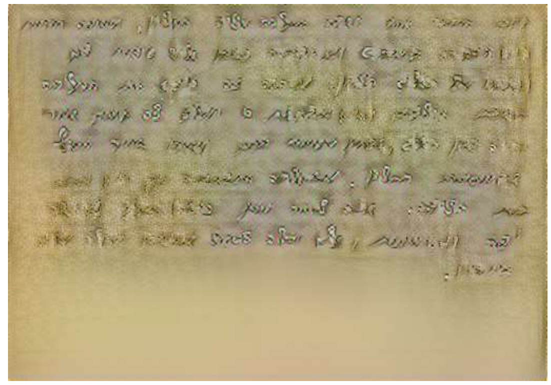
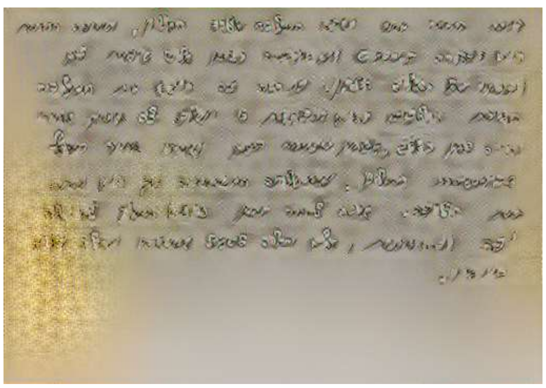
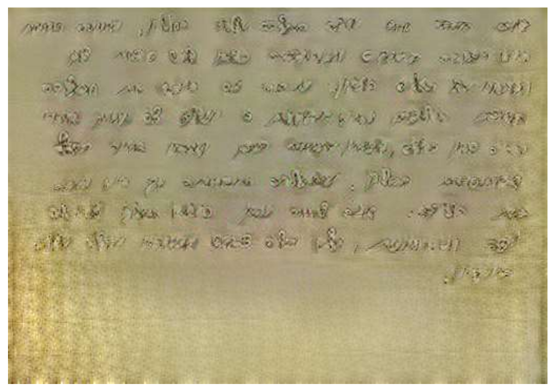
We will show samples of the results with each weight and content layer (style layers are the same).

Content image	Style image
	

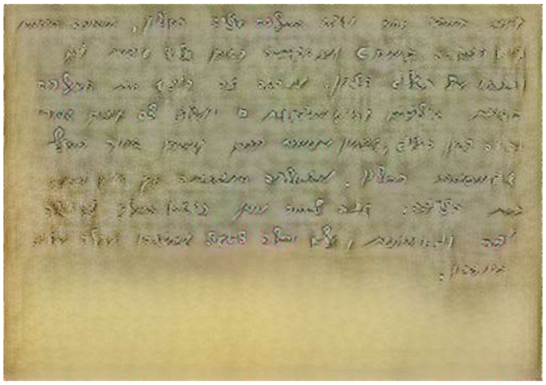
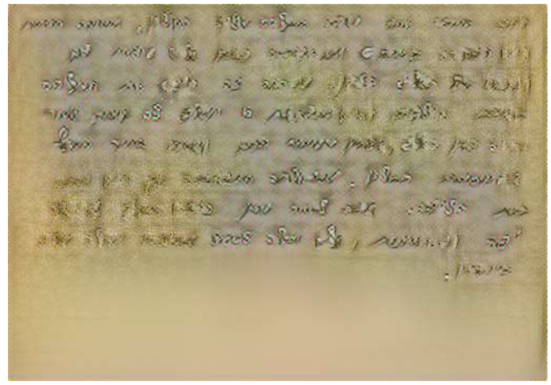
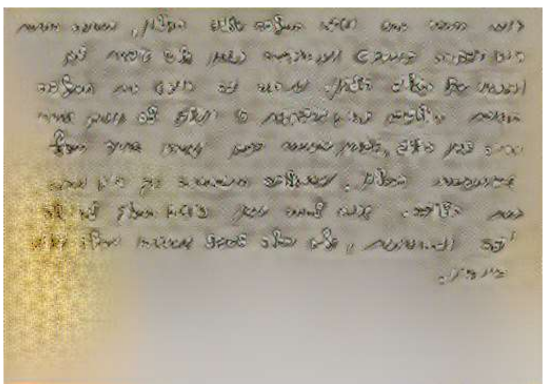
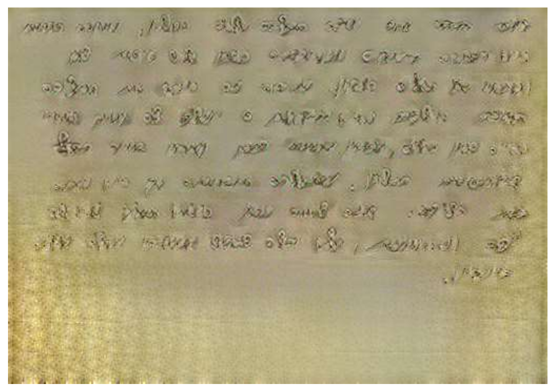
2.5.3.1. Weight 0

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

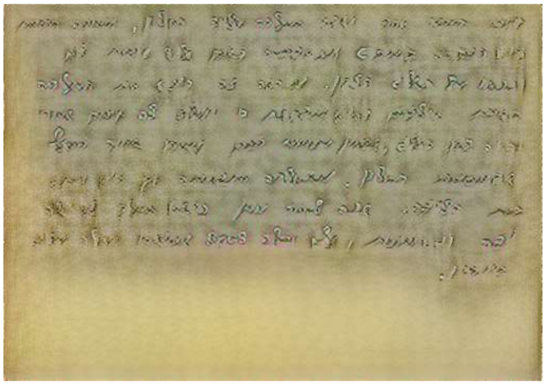
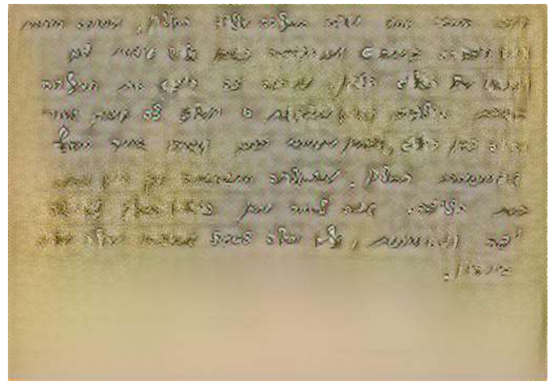
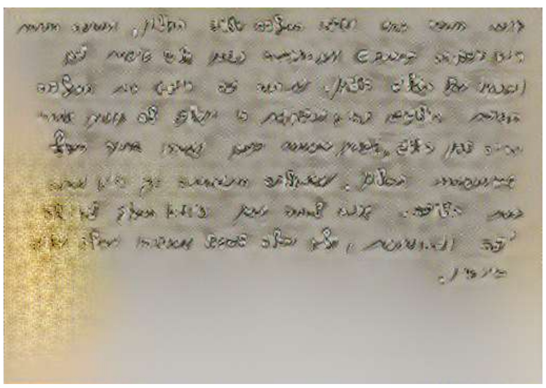
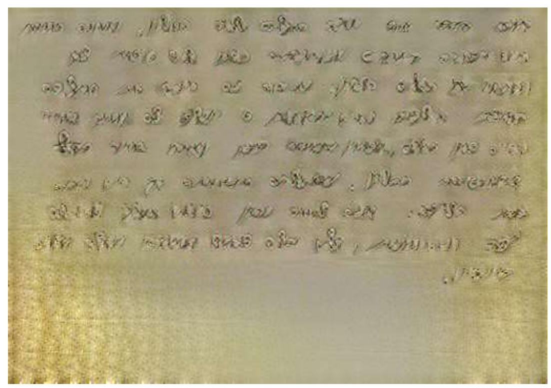
2.5.3.2. Weight 1

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

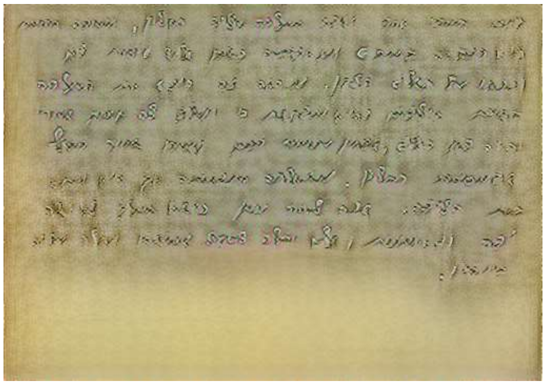
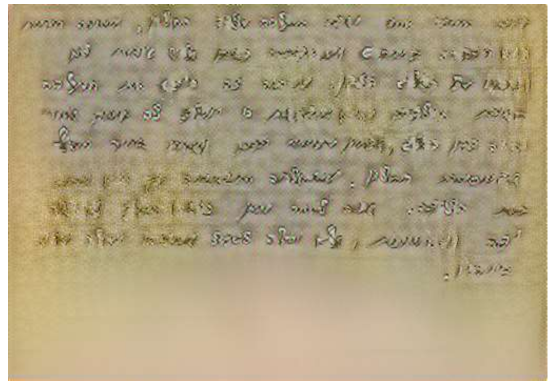
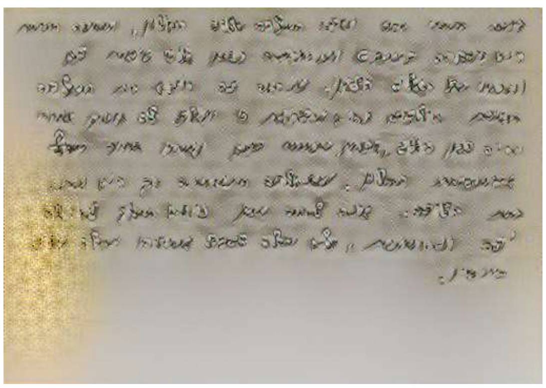
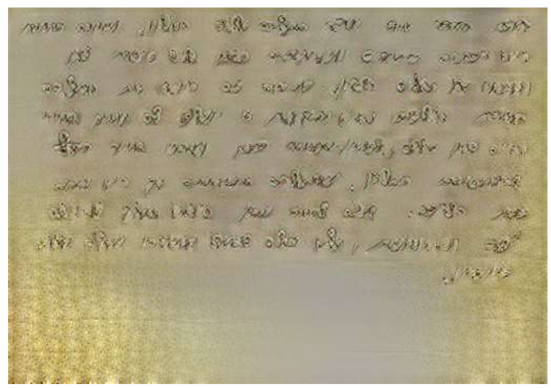
2.5.3.3. Weight 2

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

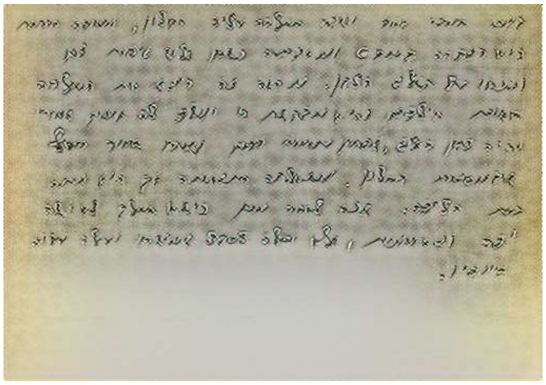
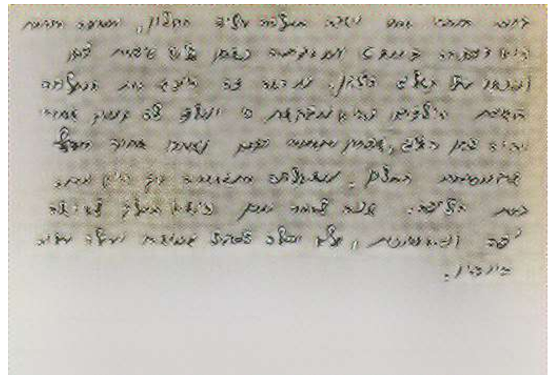
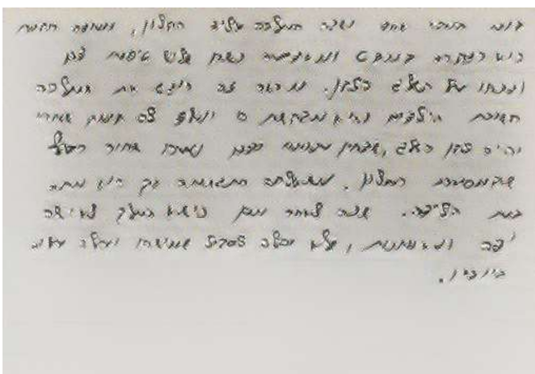
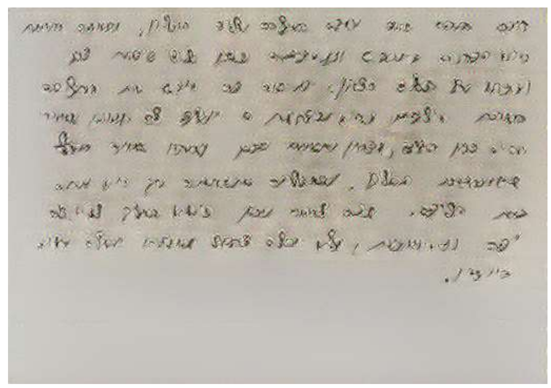
2.5.3.4. Weight 3

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

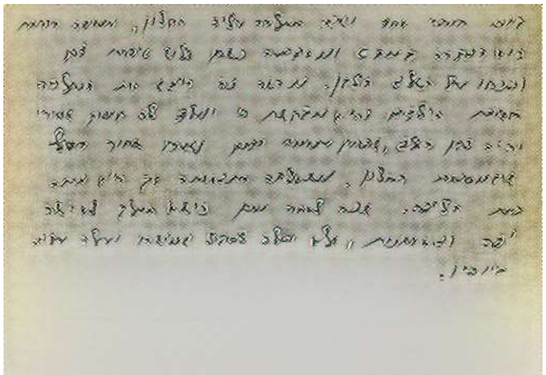
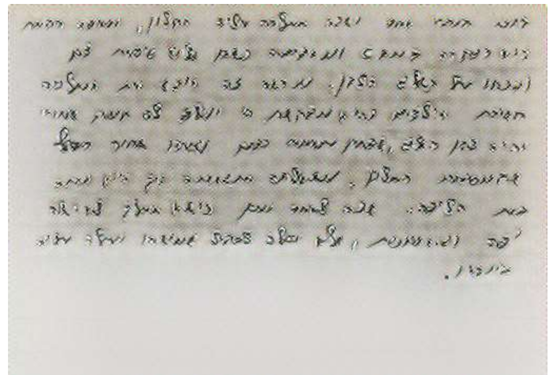
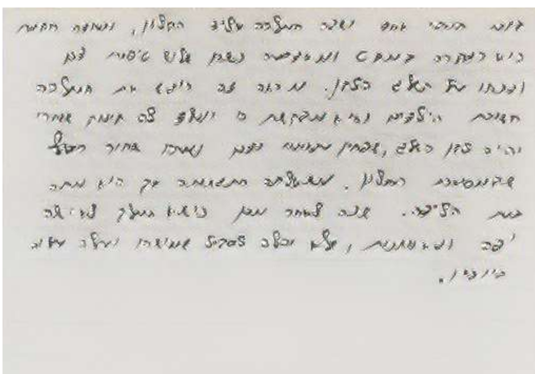
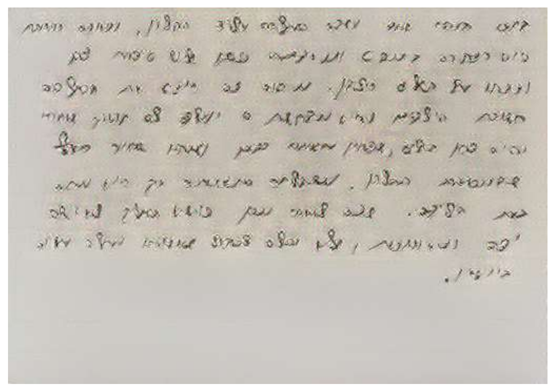
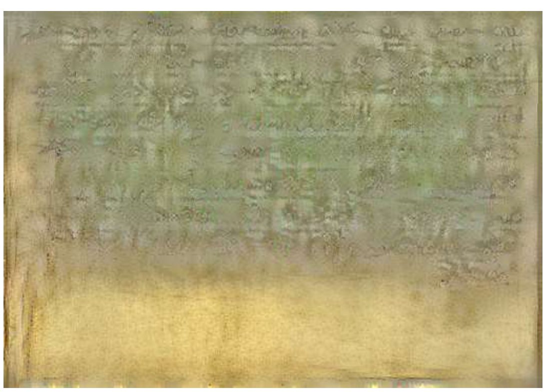
2.5.3.5. Weight 4

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

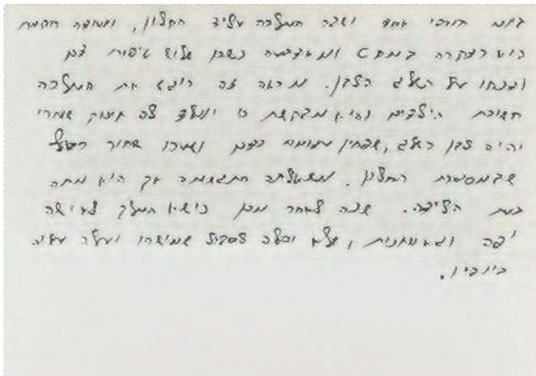
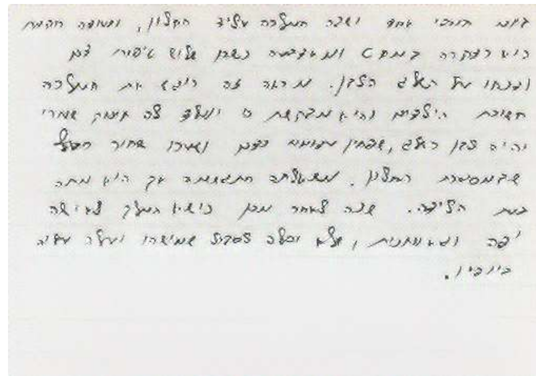
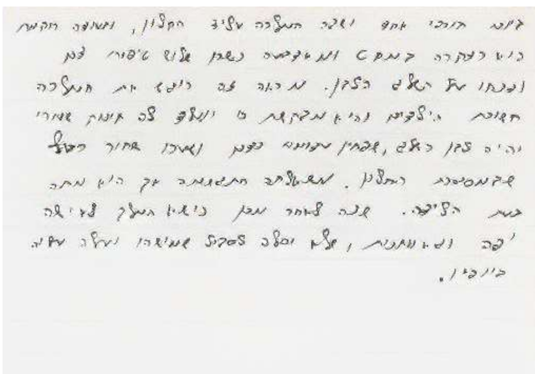
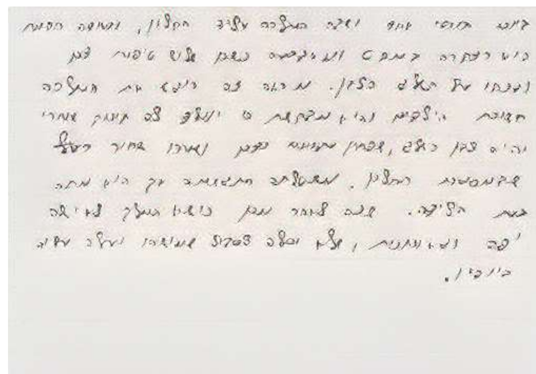
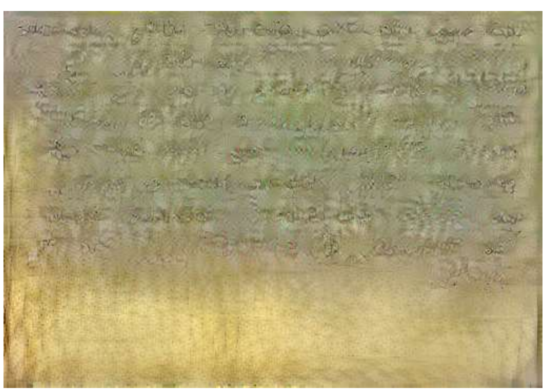
2.5.3.6. Weight 5

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

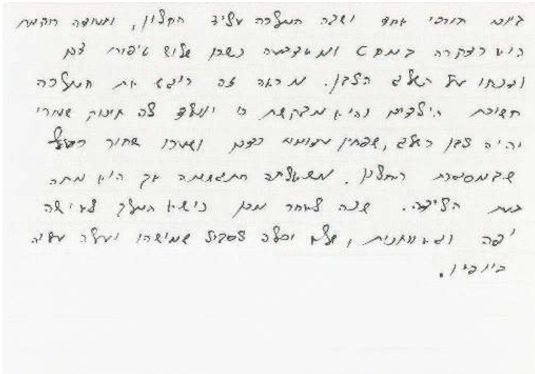
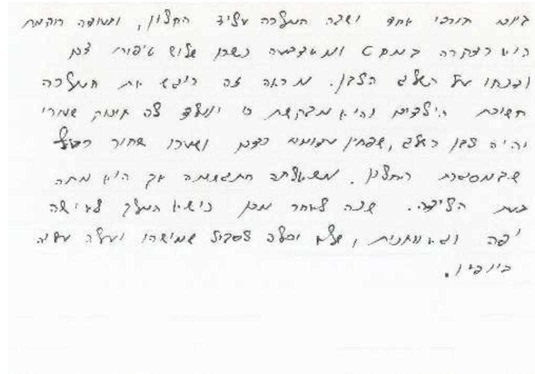
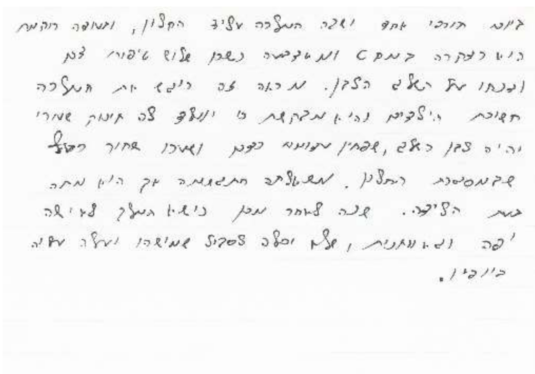
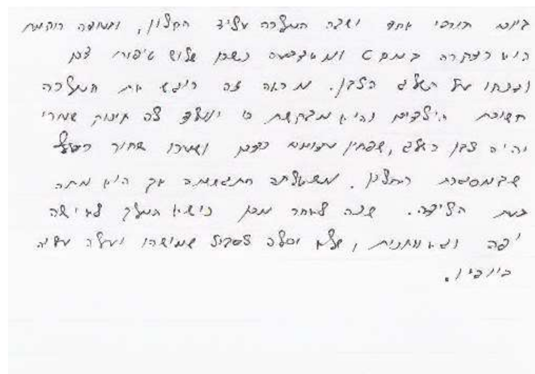
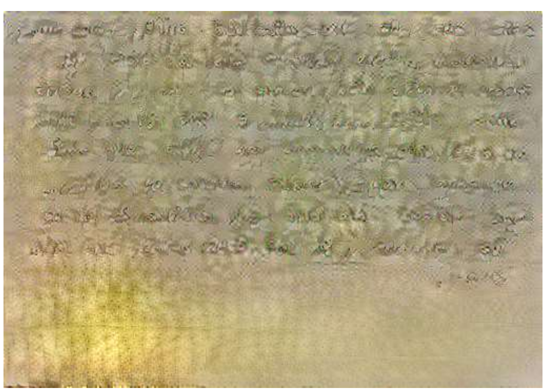
2.5.3.7. Weight 6

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.5.3.8. Weight 7

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.5.3.9. Weight 8

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.5.4. Discussion

As we can see, this experiment has the same results as experiment 1 but content representation is better than experiment 1 and text is a lot less blurry meaning we get improved results if the style does not include text.

2.6. Experiment 3

We saw good results in experiment 1 and 2. Next we will try to improve both experiments by applying dilation on the content image to make text more thicker.

2.6.1. Content Input

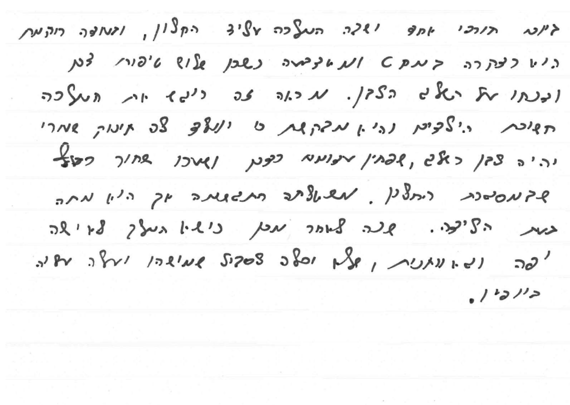
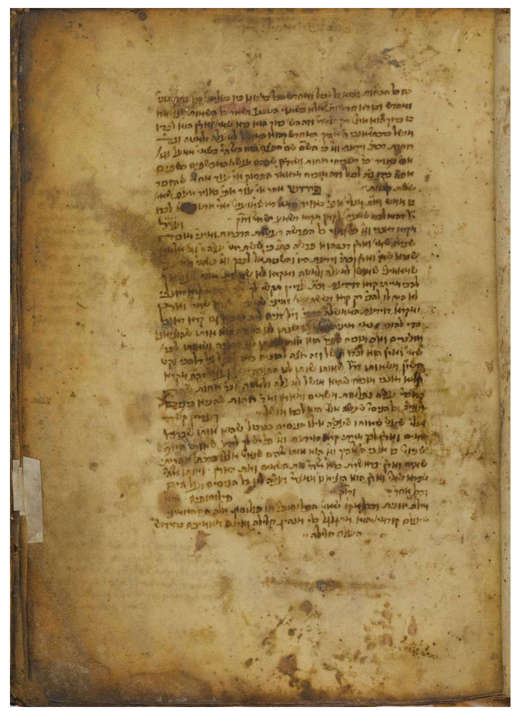
This experiment used a dilated modern hebrew handwritten document for content.

2.6.2. Style Input

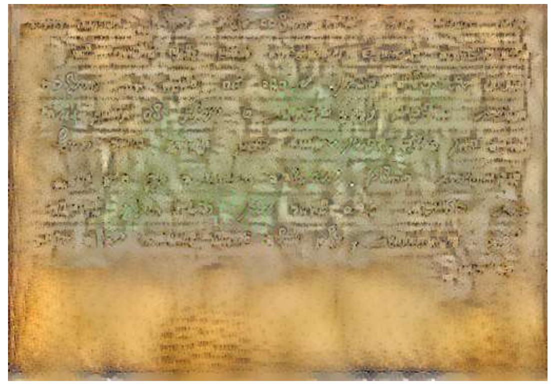
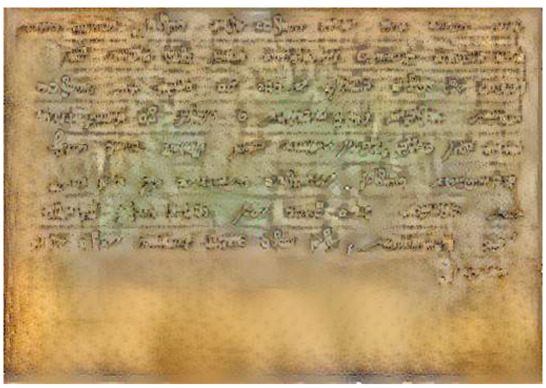
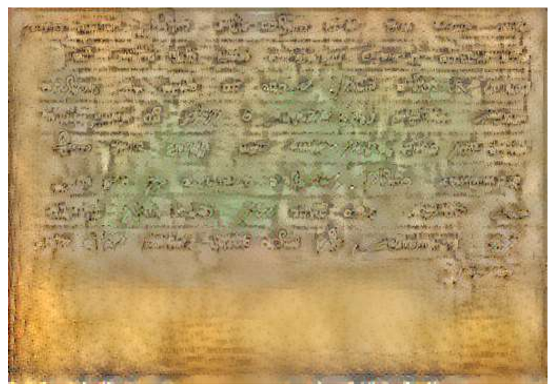
This experiment used a hebrew middle age document with text for style.

2.6.3. Results

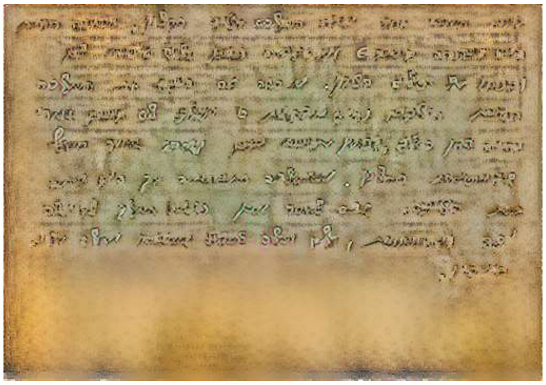
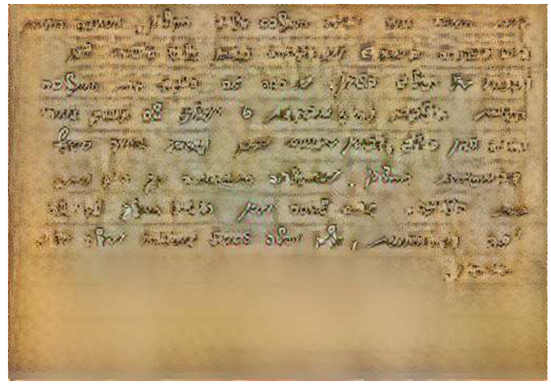
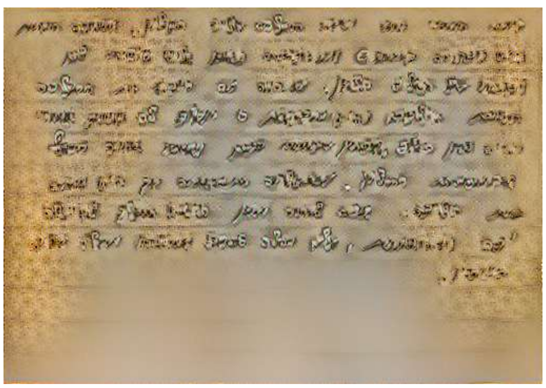
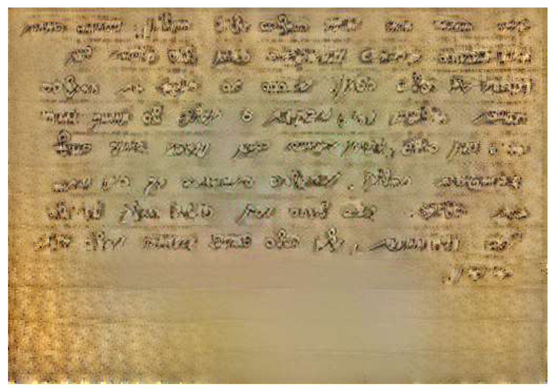
We will show samples of the results with each weight and content layer (style layers are the same).

Content image	Style image
	

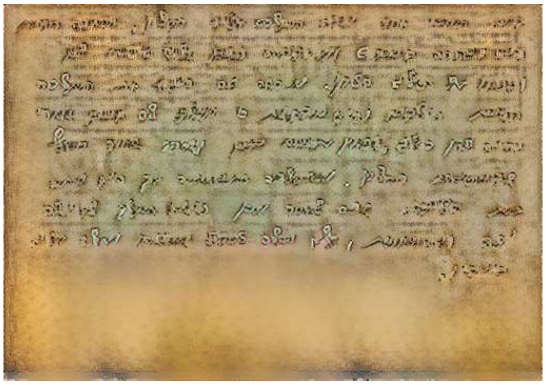
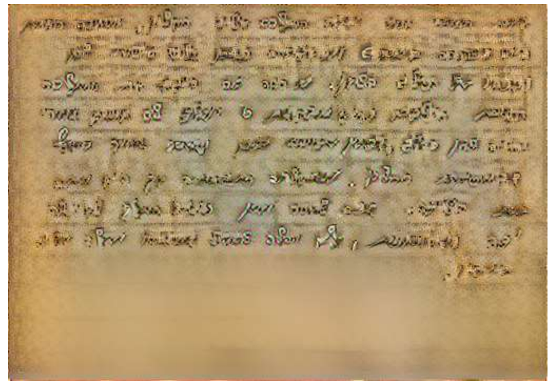
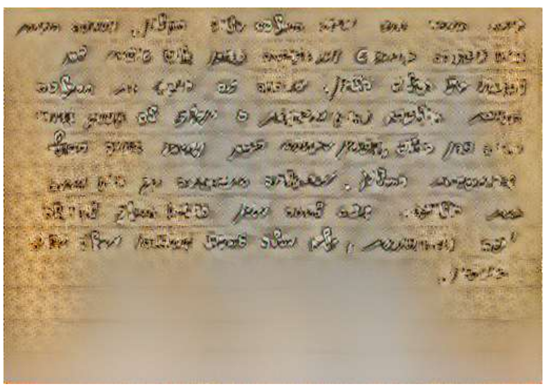
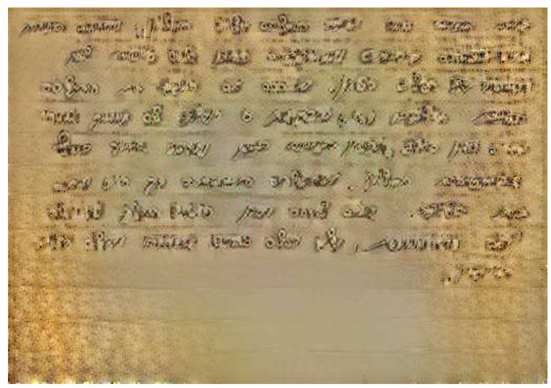
2.6.3.1. Weight 0

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

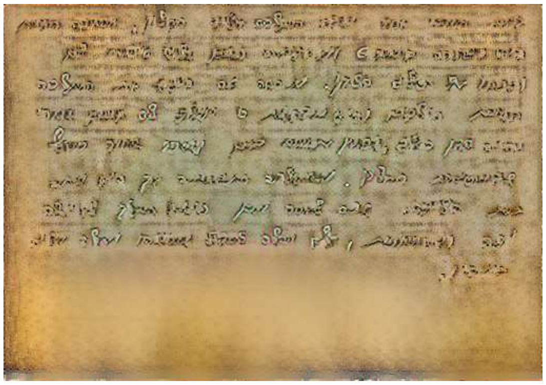
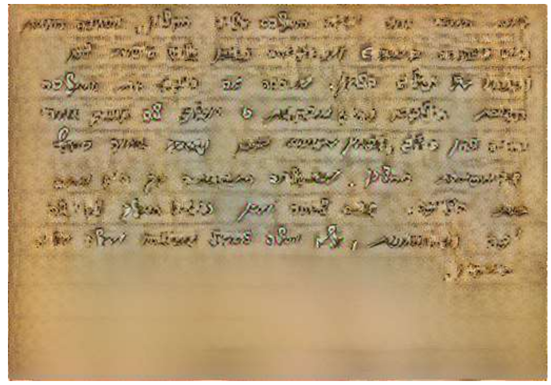
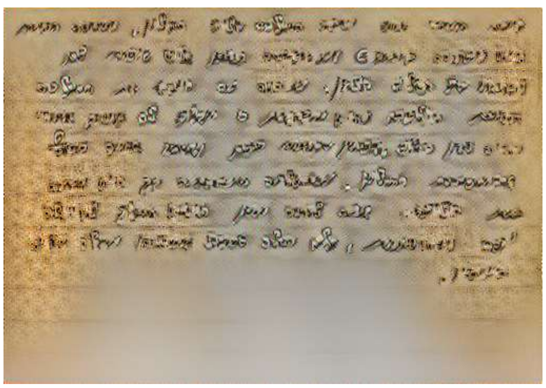
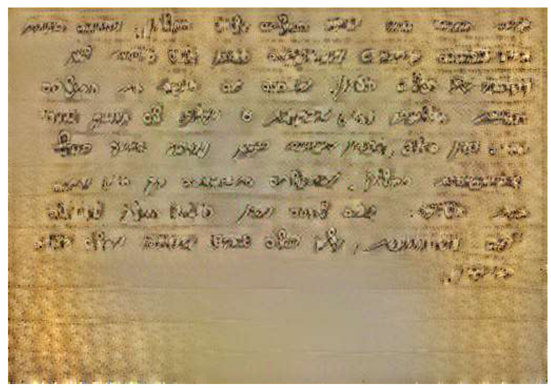
2.6.3.2. Weight 1

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

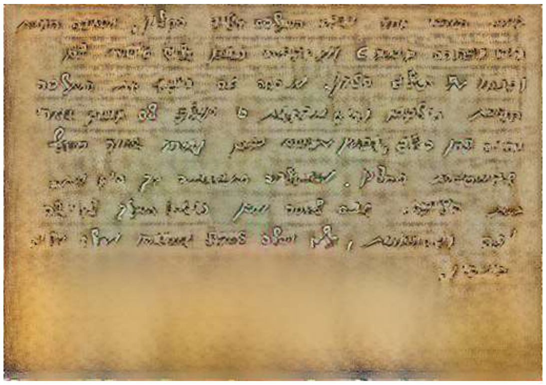
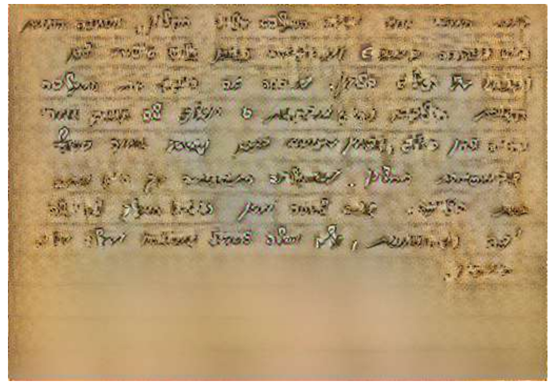
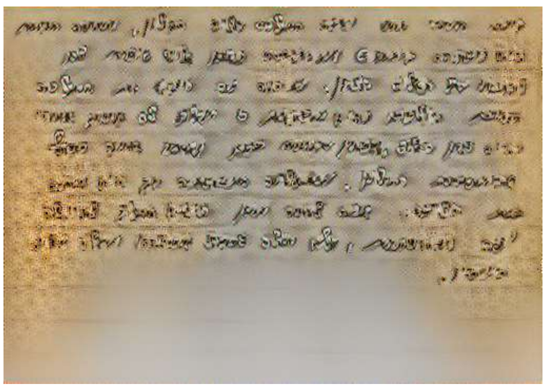
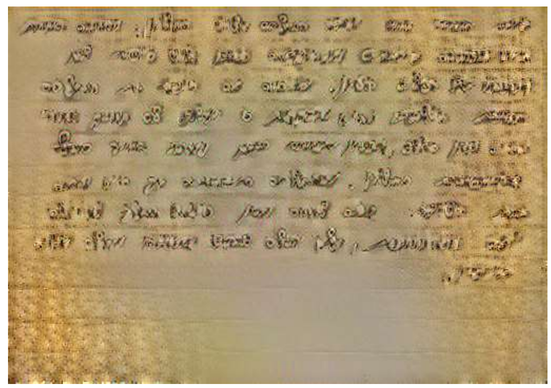
2.6.3.3. Weight 2

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

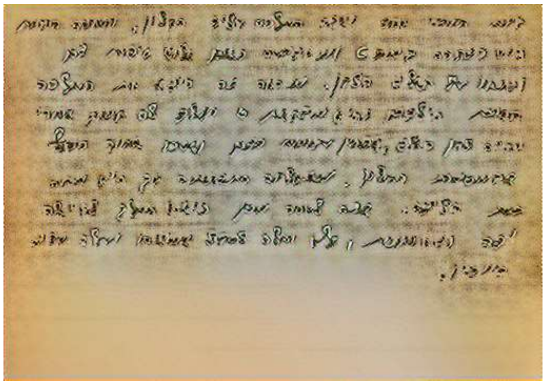
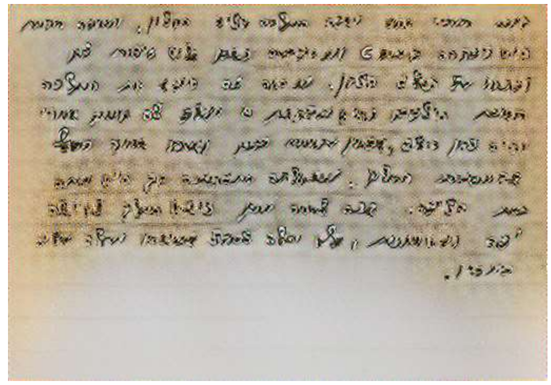
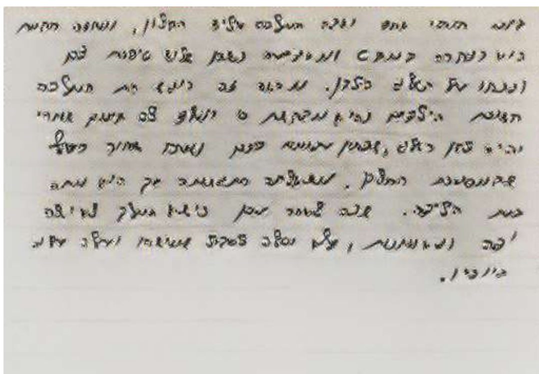
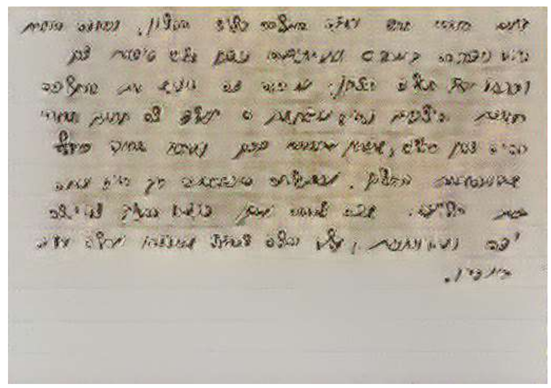
2.6.3.4. Weight 3

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

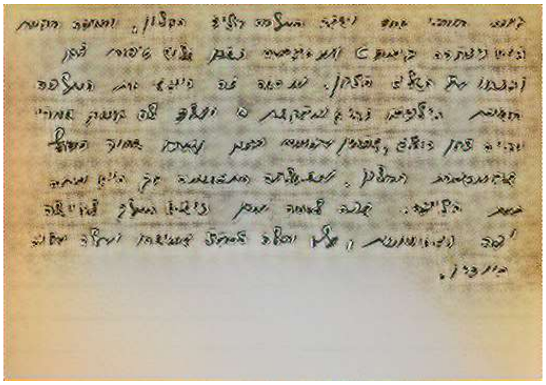
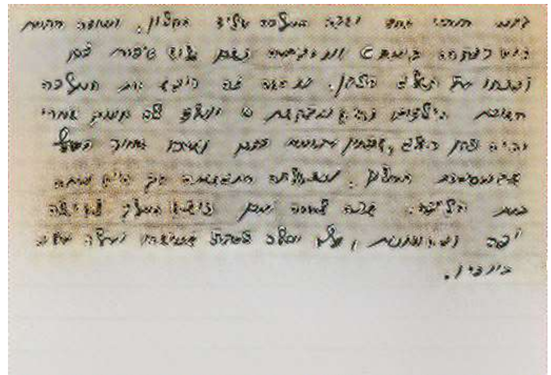
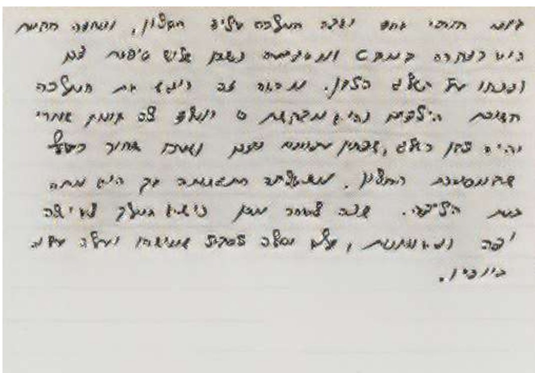
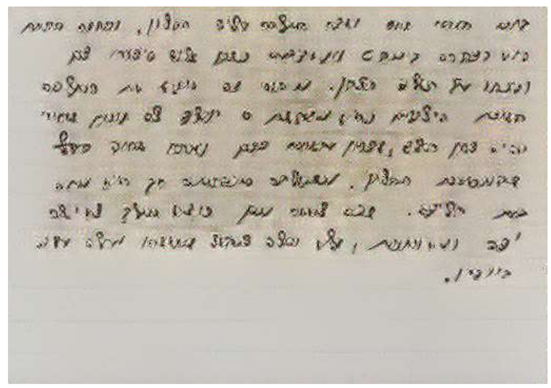
2.6.3.5. Weight 4

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

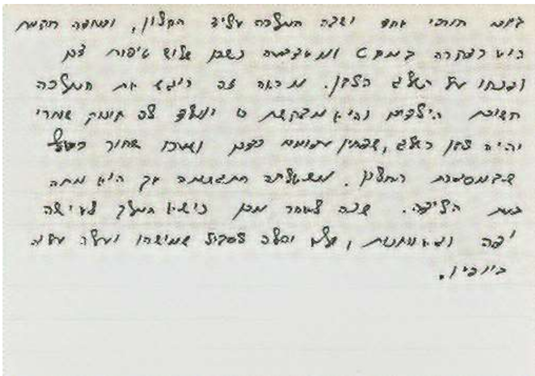
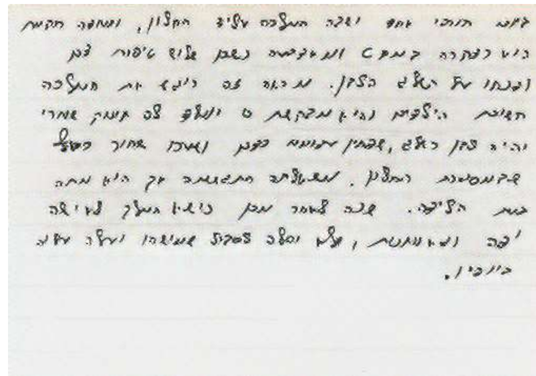
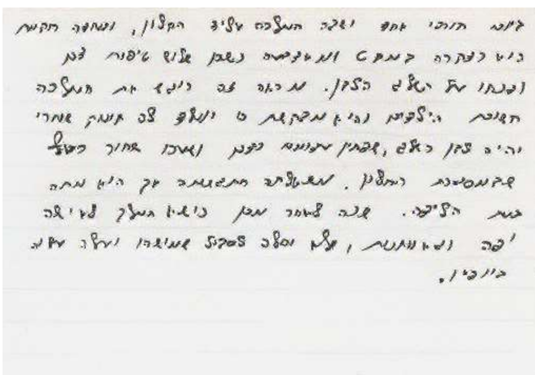
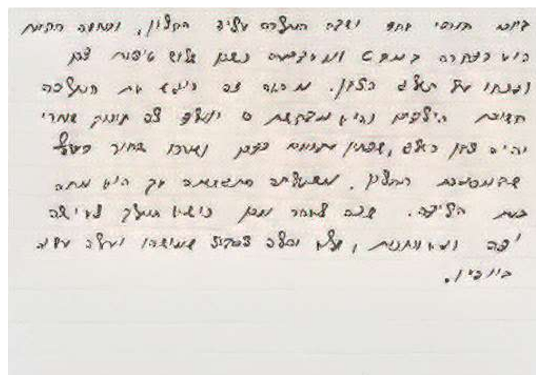
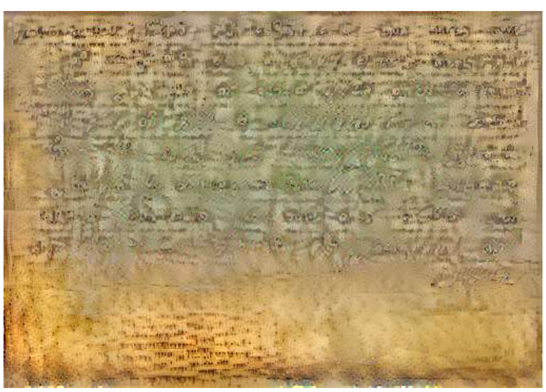
2.6.3.6. Weight 5

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

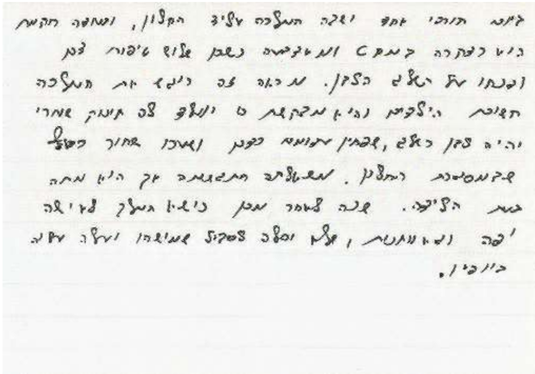
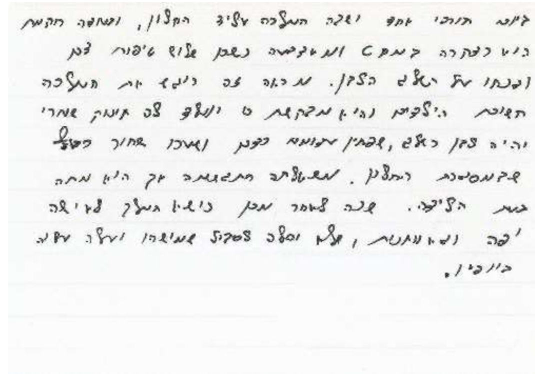
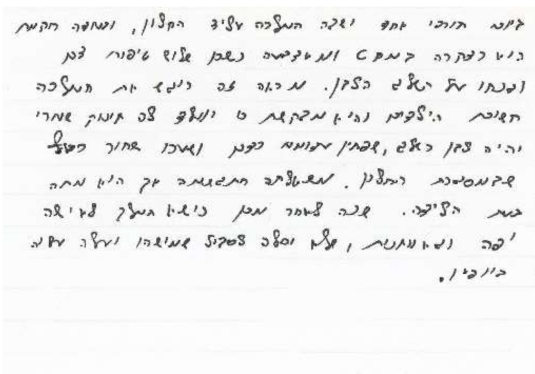
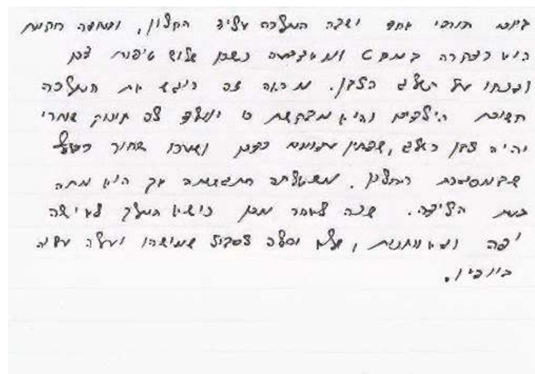
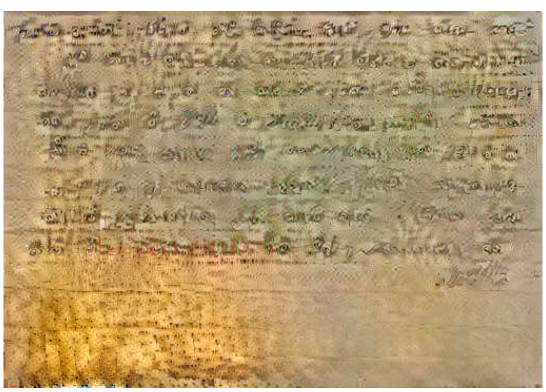
2.6.3.7. Weight 6

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.6.3.8. Weight 7

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.6.3.9. Weight 8

block3_conv2	block3_conv3
	
block3_conv4	**block4_conv2**
	
block5_conv2	
	

2.6.4. Discussion

As we can see, this experiment has the same results as experiment 1 but content representation is better than experiment 1 and the text is blurry.

2.7. Experiment 4

We saw better results in experiment 3 than 1 but the text is a little bit blurry. In experiment 2 we saw that text in style image has an effect on this so lets test without text in order to improve results.

2.7.1. Content Input

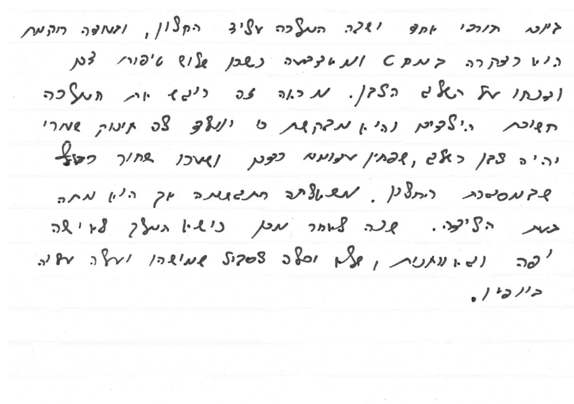
This experiment used a dilated modern hebrew handwritten document for content.

2.7.2. Style Input

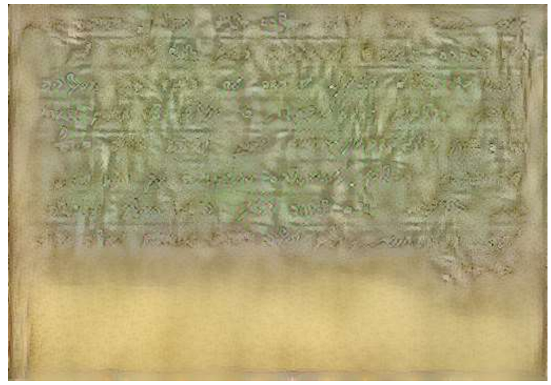
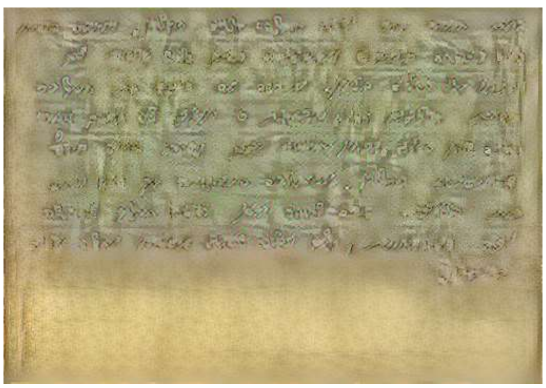
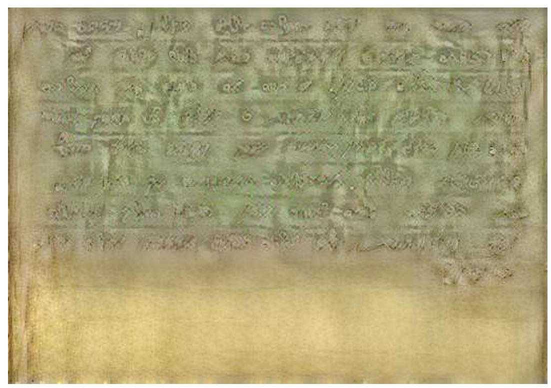
This experiment used a hebrew middle age document without text for style.

2.7.3. Results

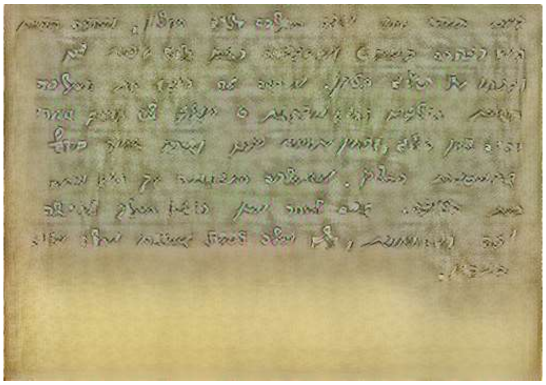
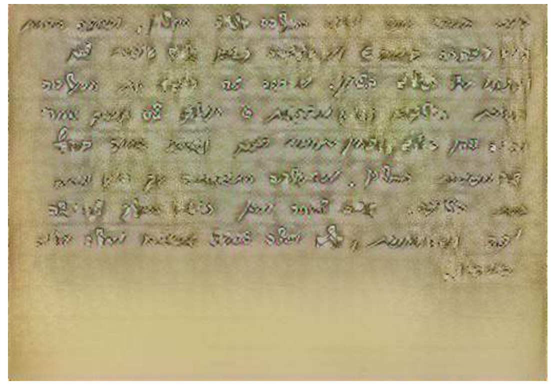
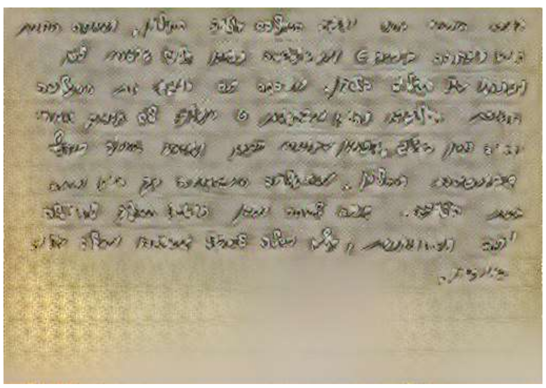
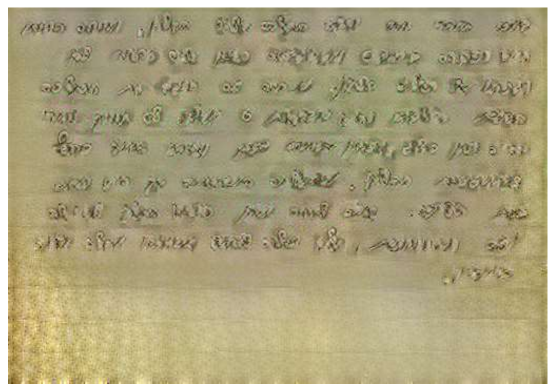
We will show samples of the results with each weight and content layer (style layers are the same).

Content image	Style image
	

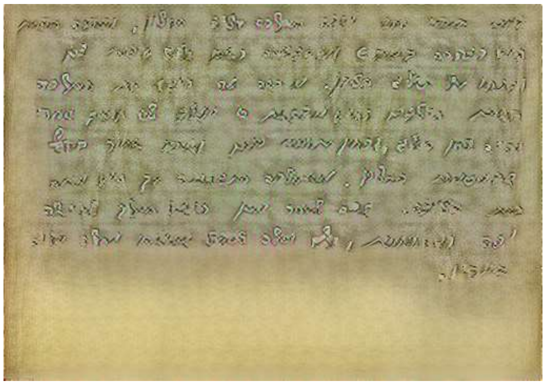
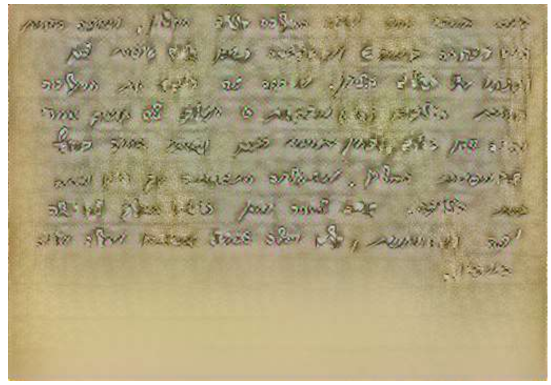
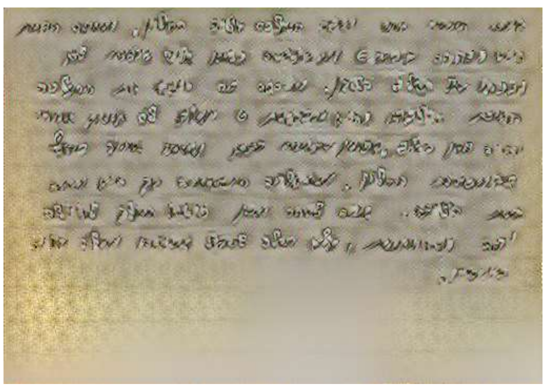
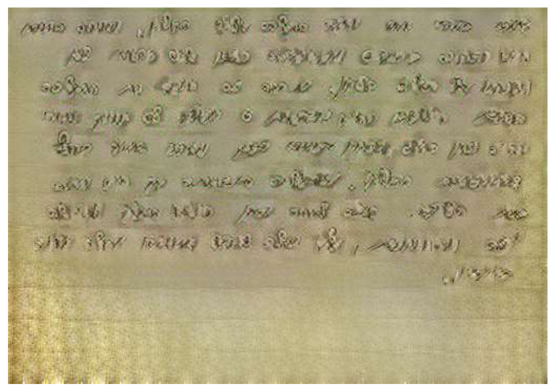
2.7.3.1. Weight 0

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

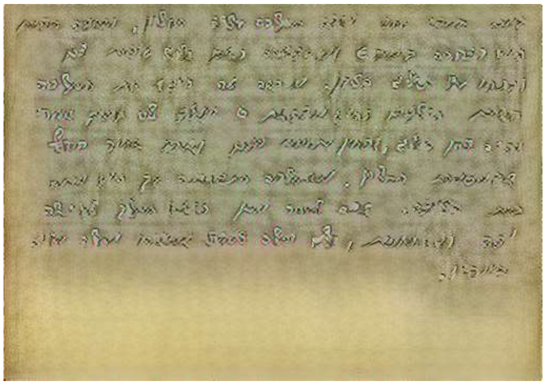
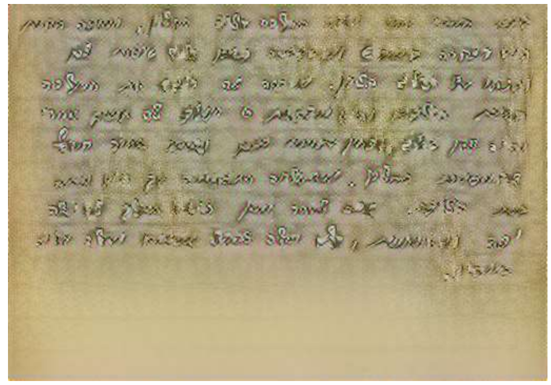
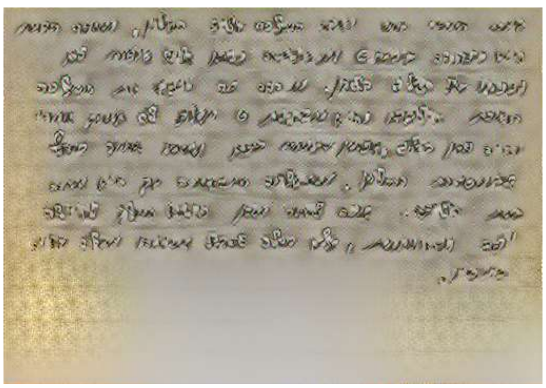
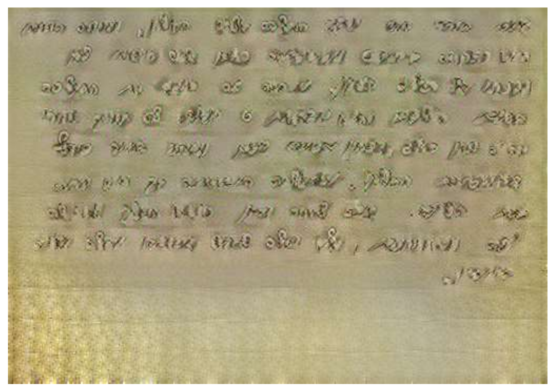
2.7.3.2. Weight 1

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

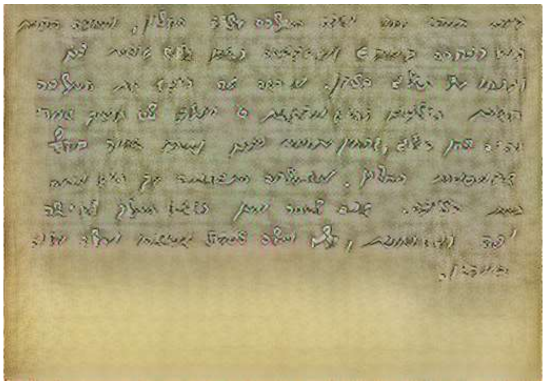
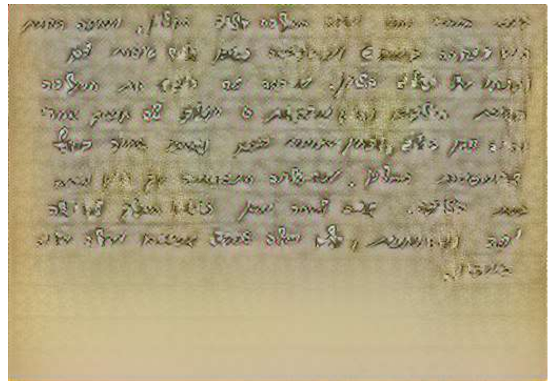
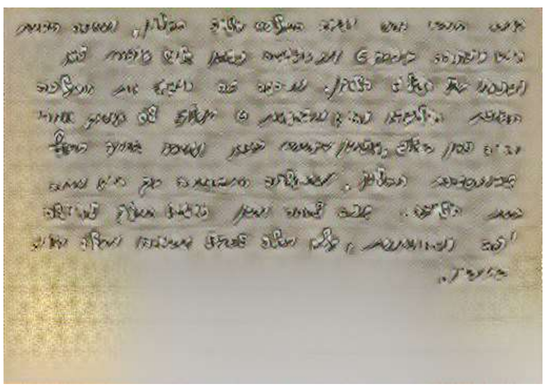
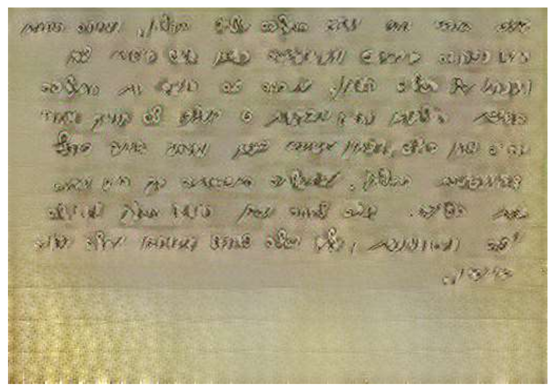
2.7.3.3. Weight 2

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

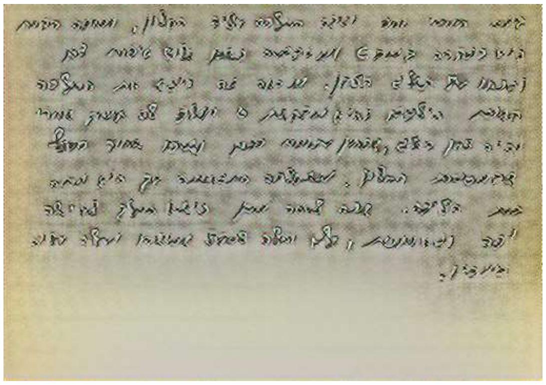
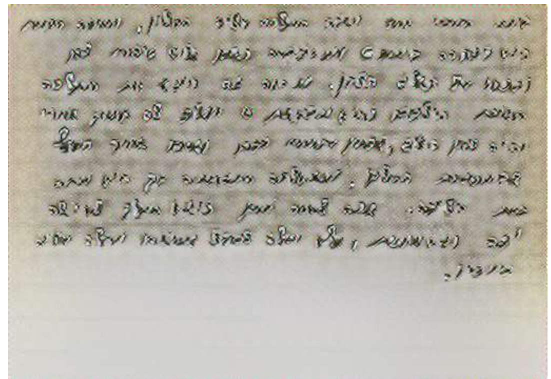
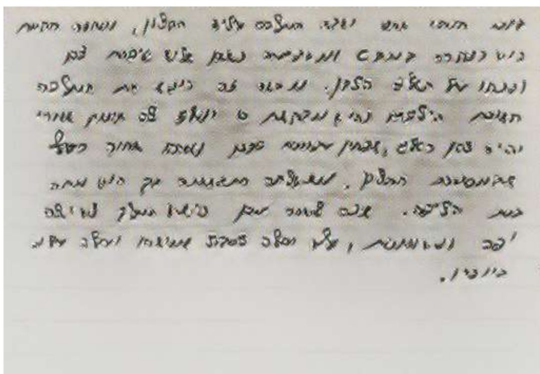
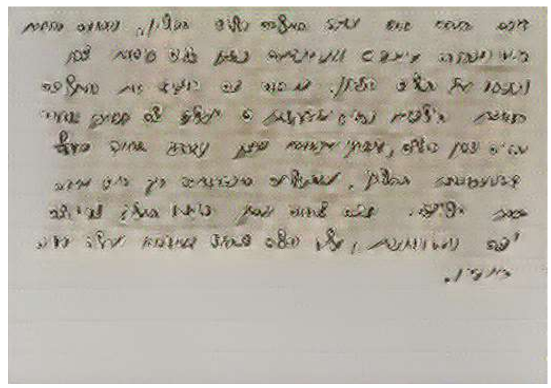
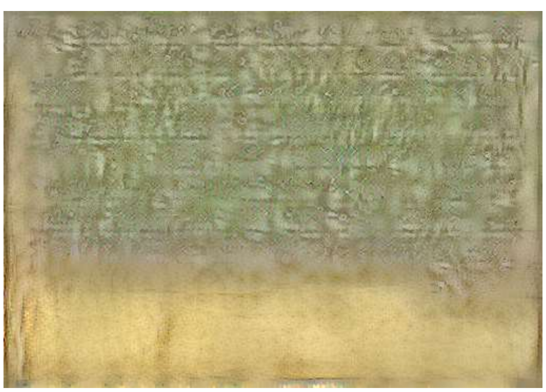
2.7.3.4. Weight 3

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

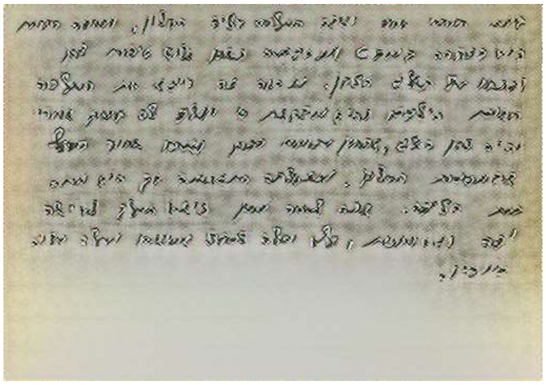
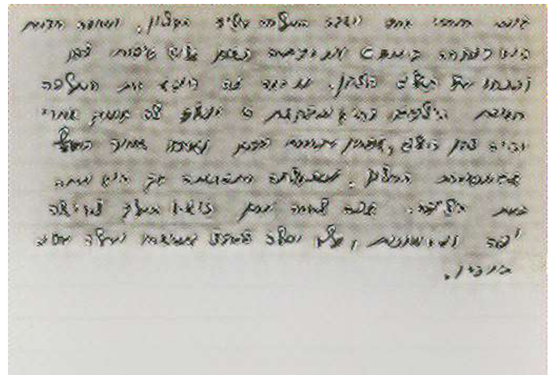
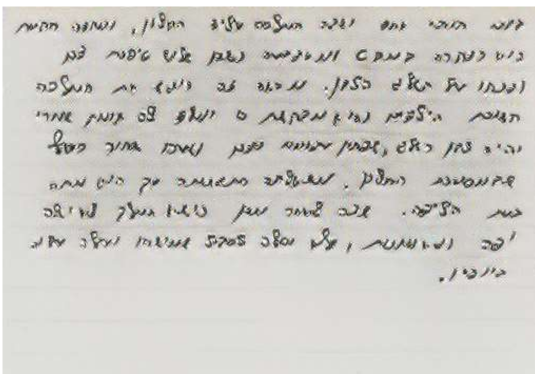
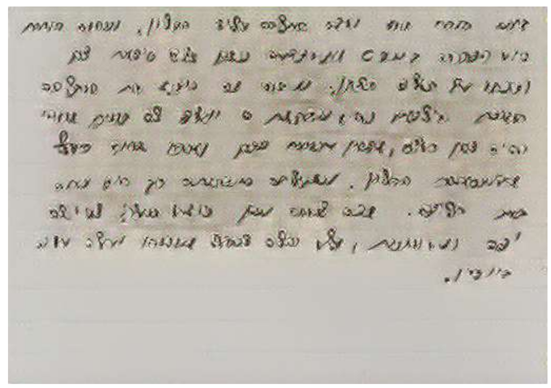
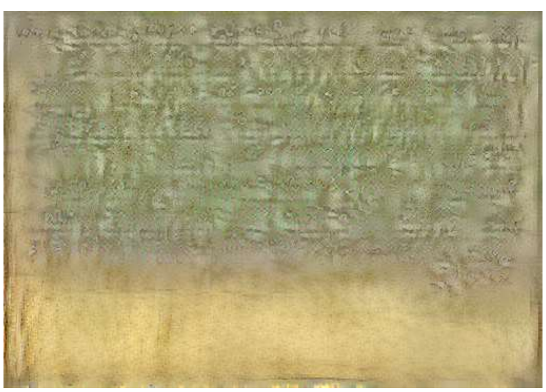
2.7.3.5. Weight 4

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

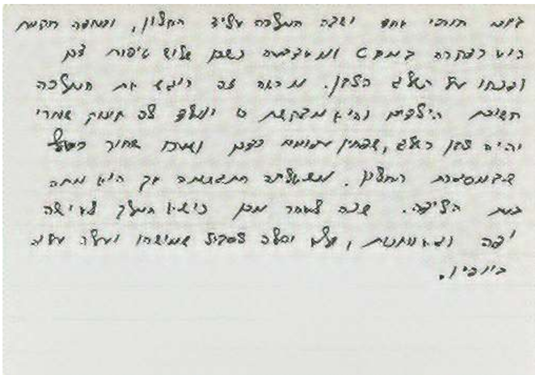
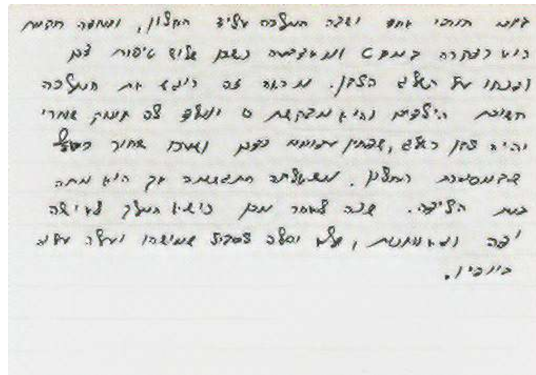
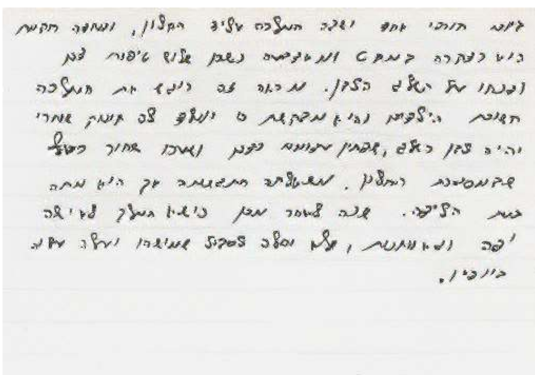
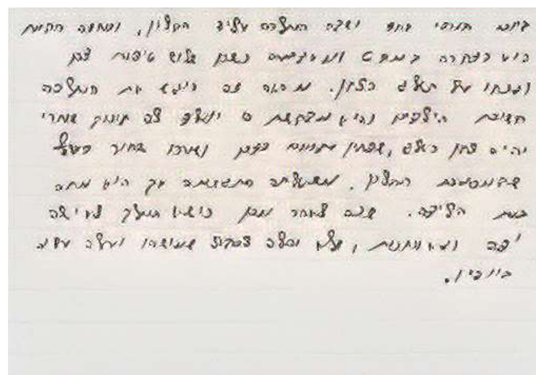
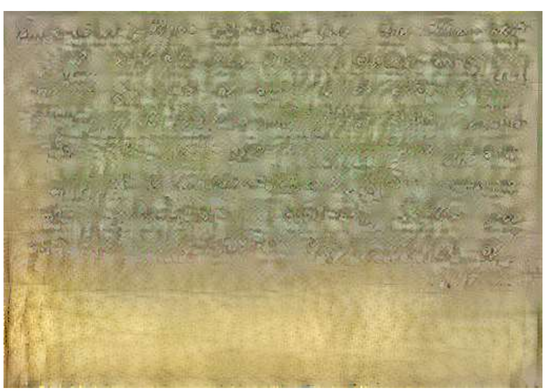
2.7.3.6. Weight 5

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

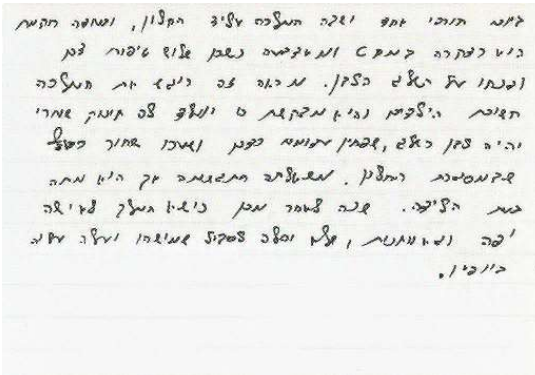
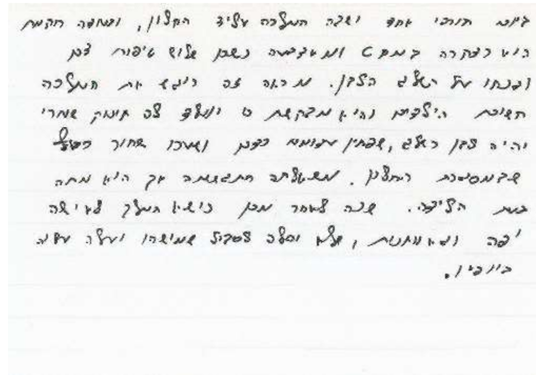
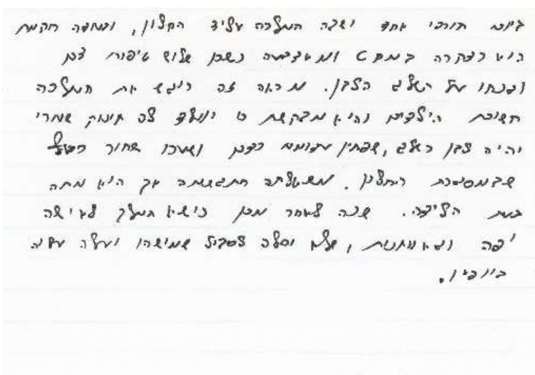
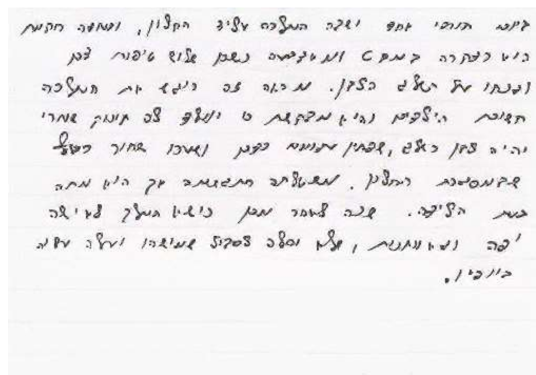
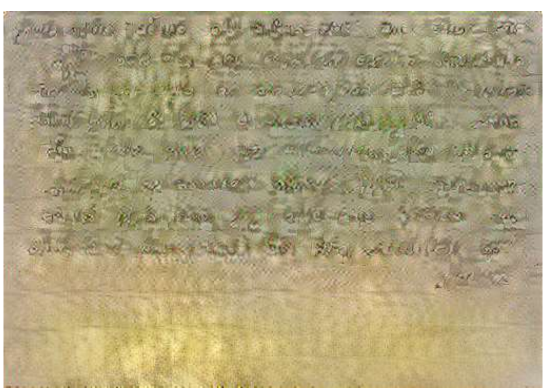
2.7.3.7. Weight 6

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.7.3.8. Weight 7

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.7.3.9. Weight 8

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.7.4. *Discussion*

As we can see, in this experiment we get better results than experiments 1, 2 and 3, meaning the dilation improves our results.

2.8. Experiment 5

We saw good results in experiment 3 and 4. Next we will try to apply binarization on content image to see if this can also improve experiments 1 and 2.

2.8.1. Content Input

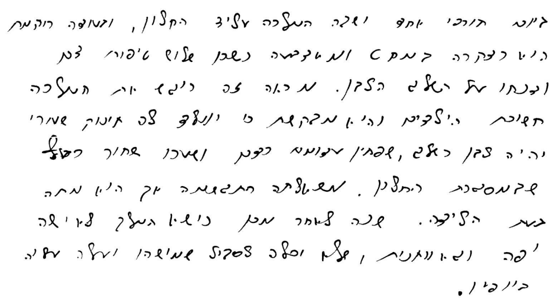
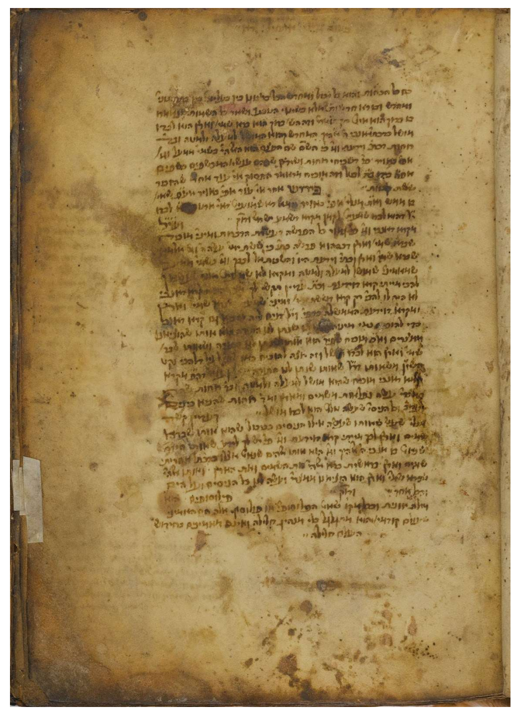
This experiment used a binary modern hebrew handwritten document for content.

2.8.2. Style Input

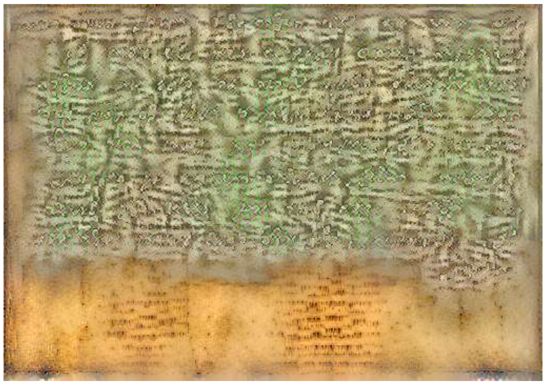
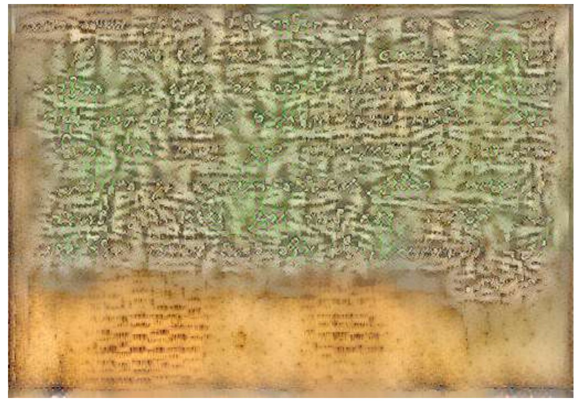
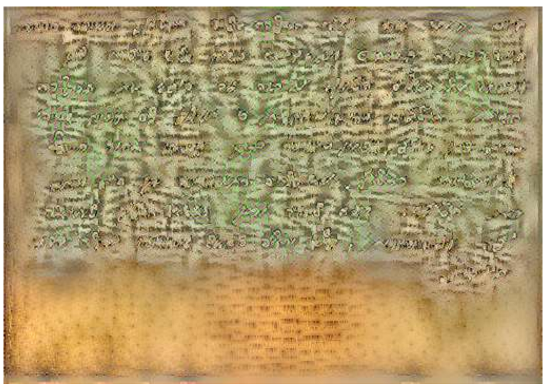
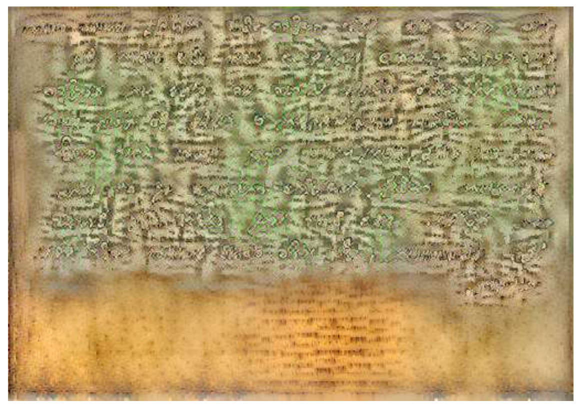
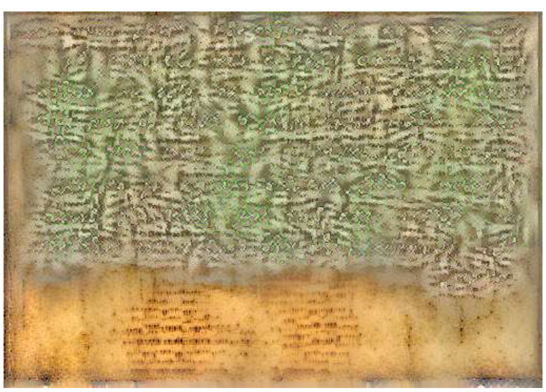
This experiment used a hebrew middle age document with text for style.

2.8.3. Results

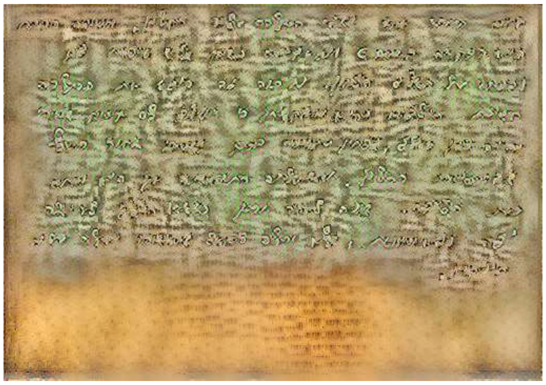
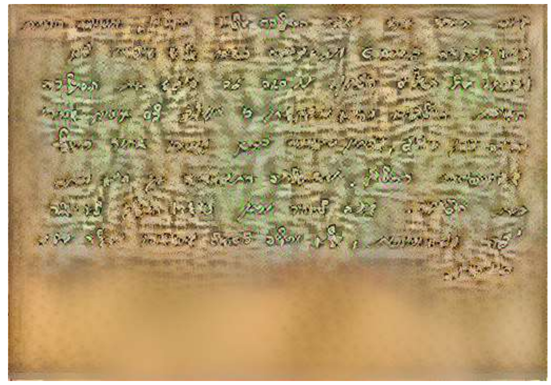
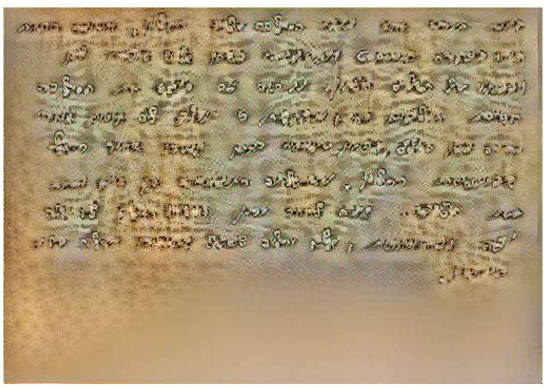
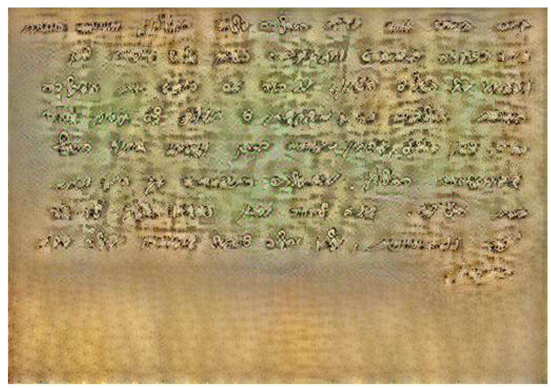
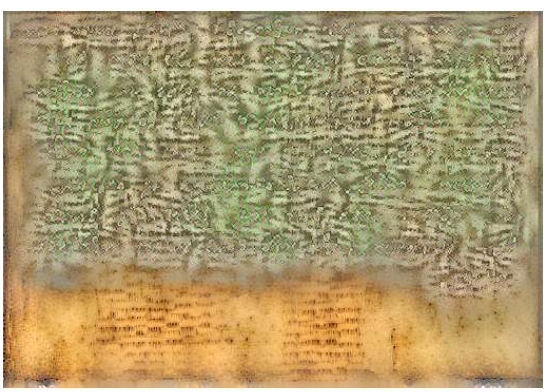
We will show samples of the results with each weight and content layer (style layers are the same).

Content image	Style image
	

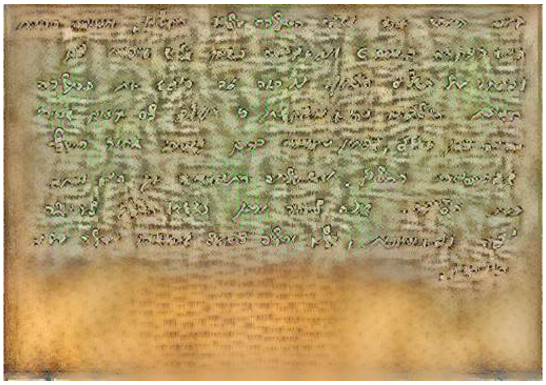
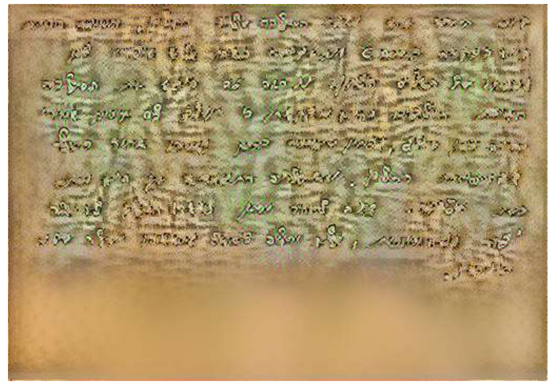
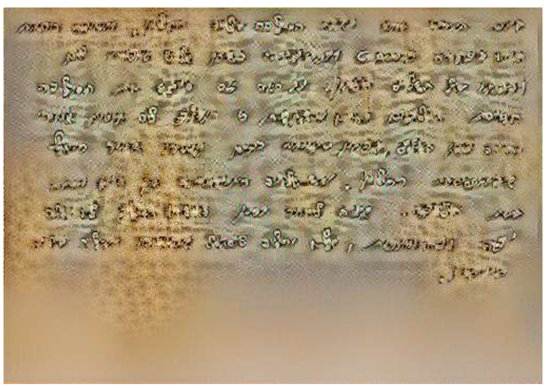
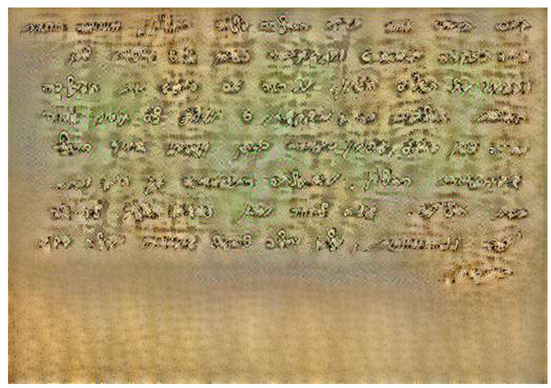
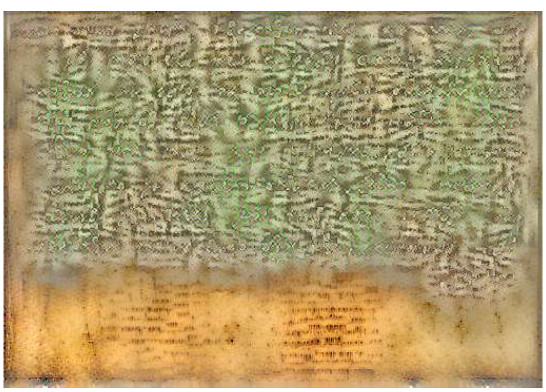
2.8.3.1. Weight 0

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

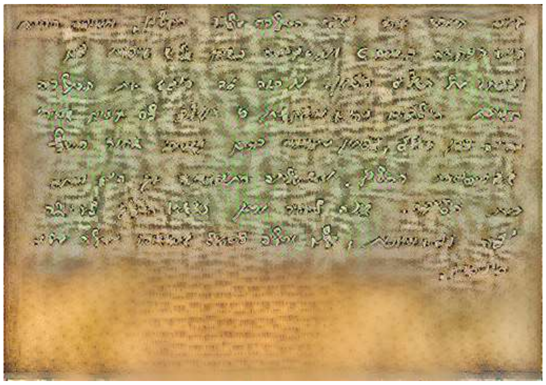
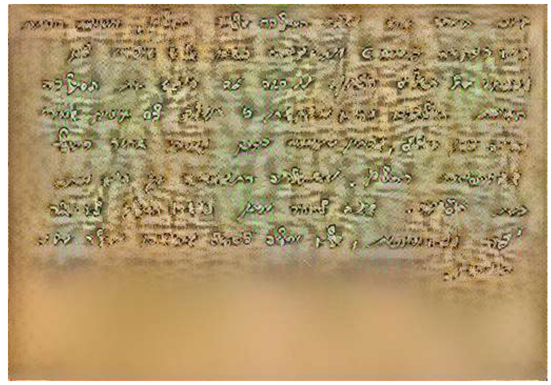
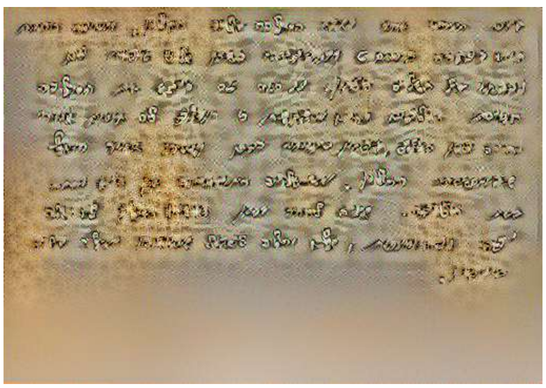
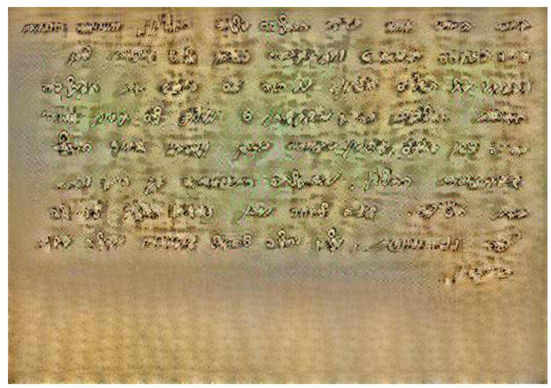
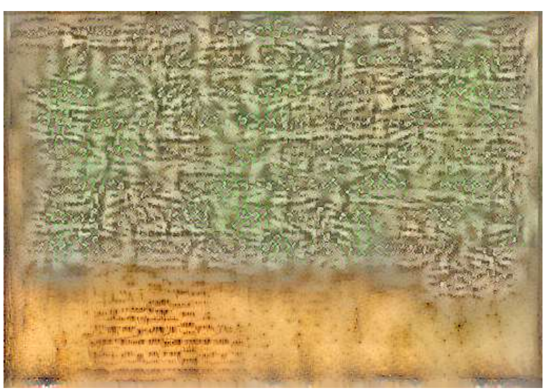
2.8.3.2. Weight 1

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

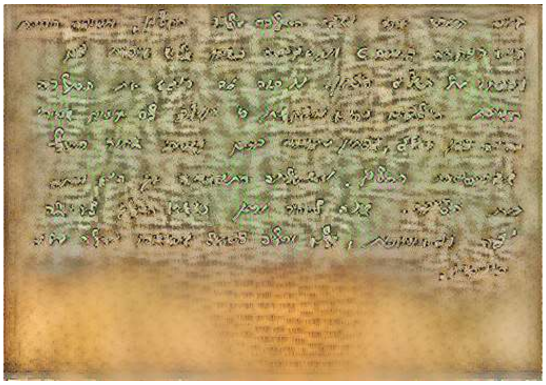
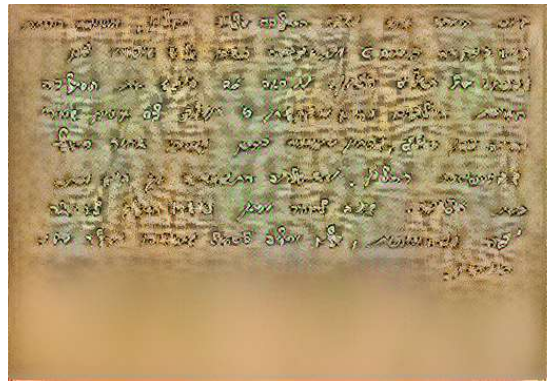
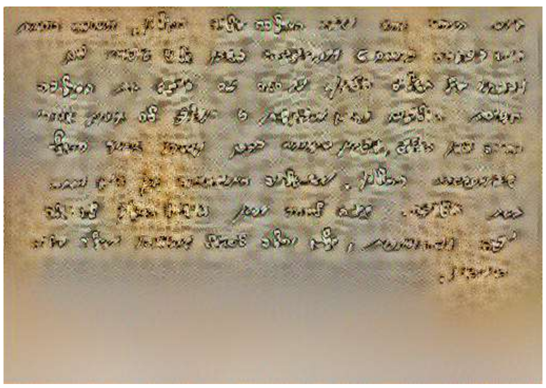
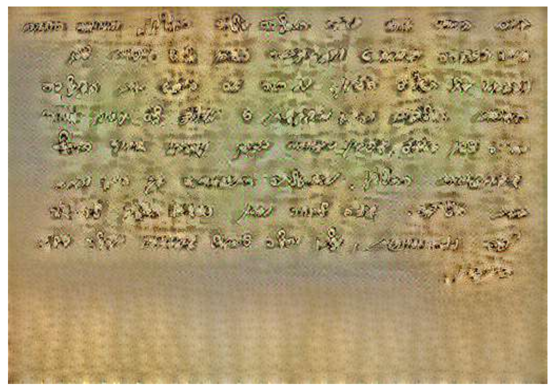
2.8.3.3. Weight 2

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

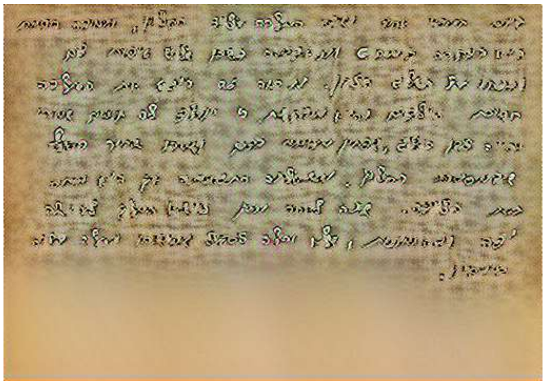
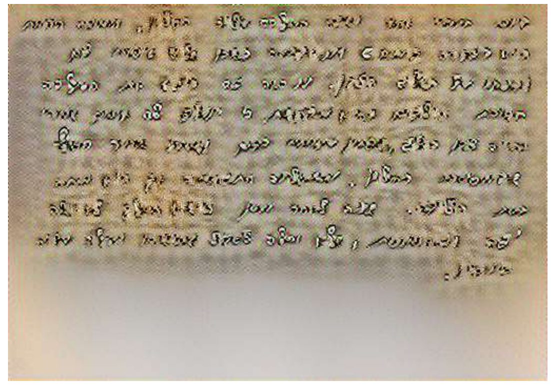
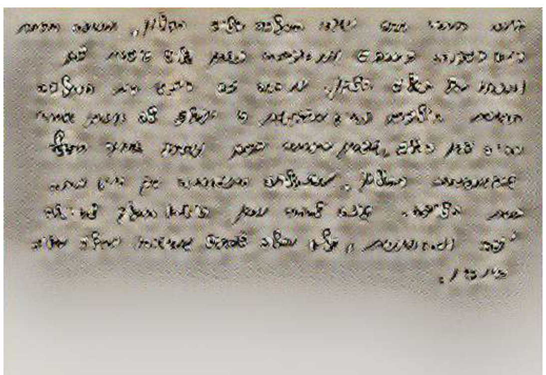
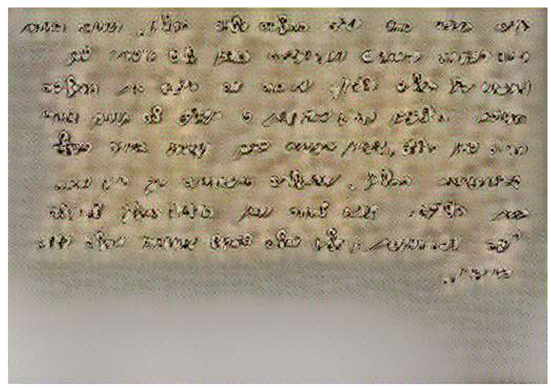
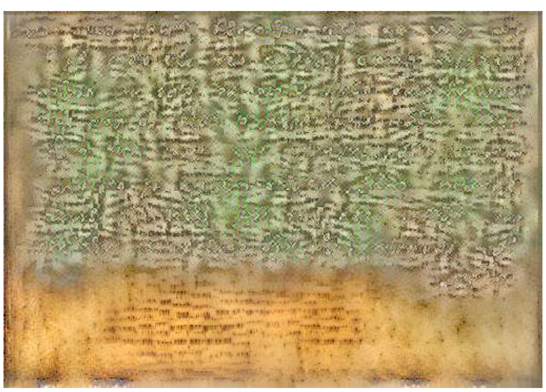
2.8.3.4. Weight 3

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

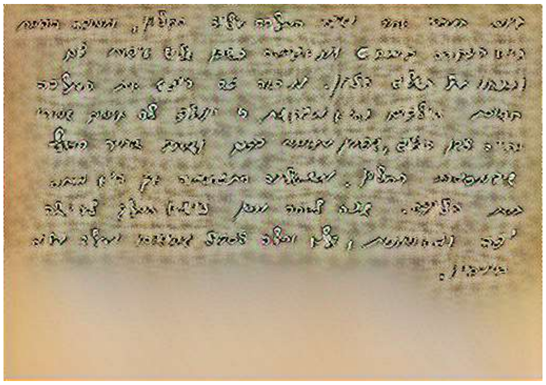
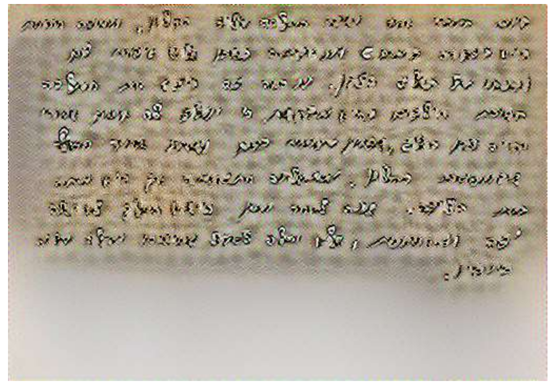
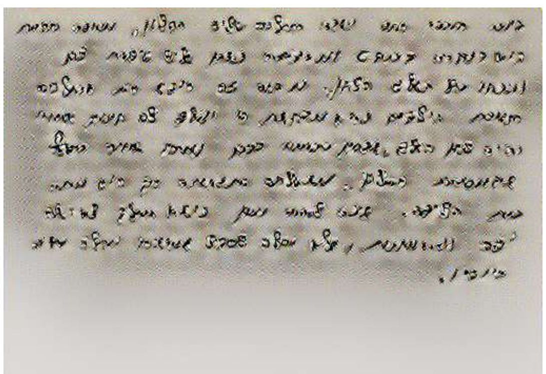
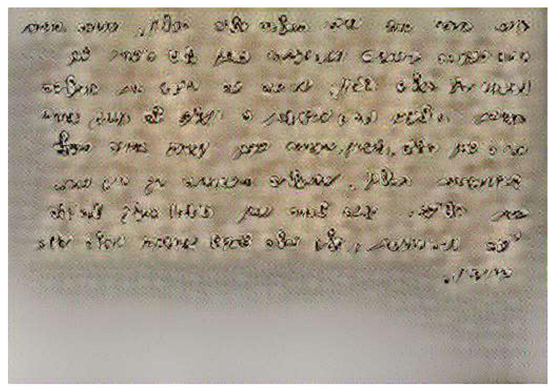
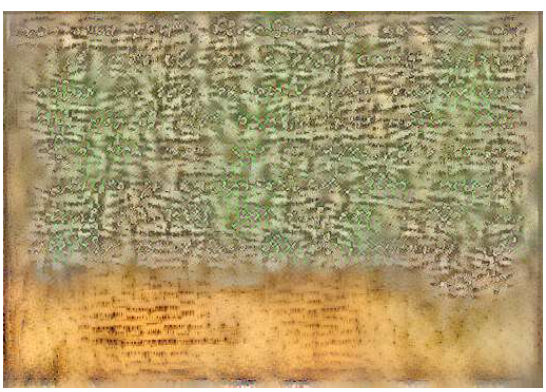
2.8.3.5. Weight 4

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

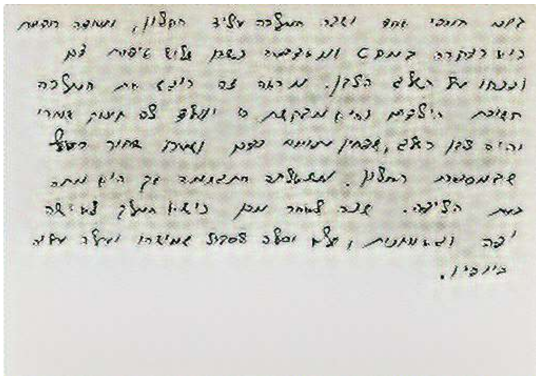
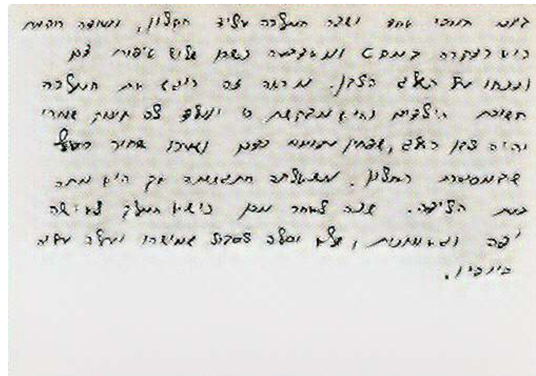
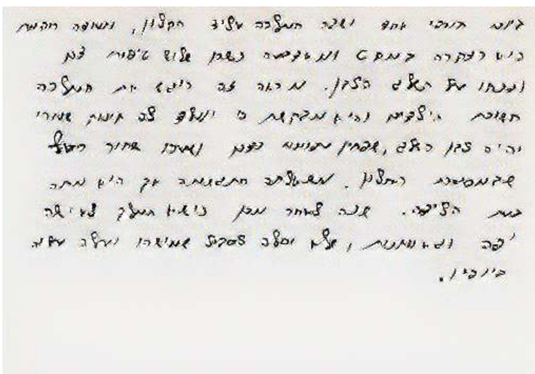
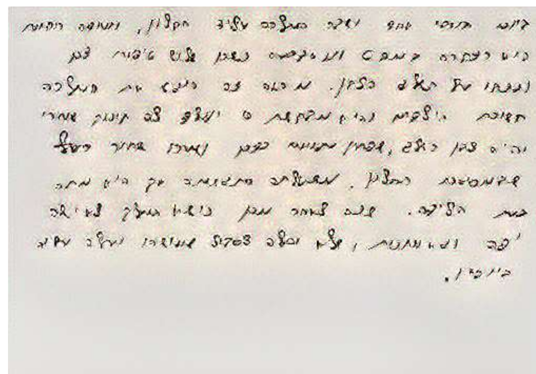
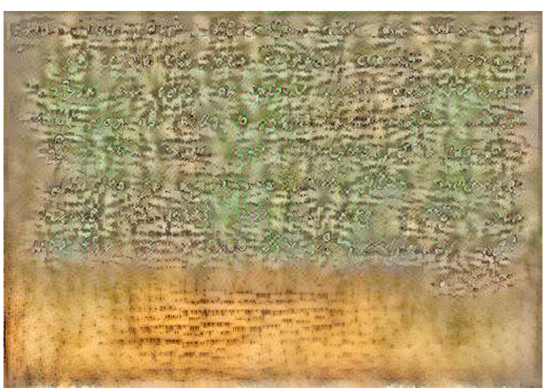
2.8.3.6. Weight 5

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

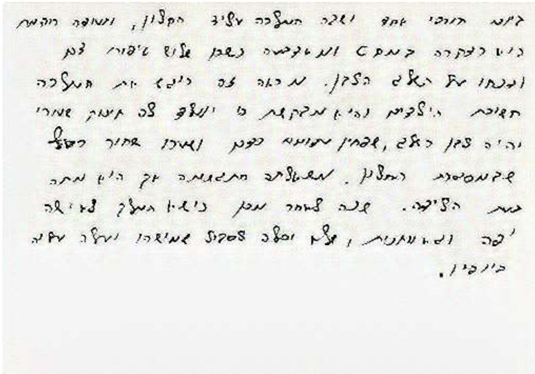
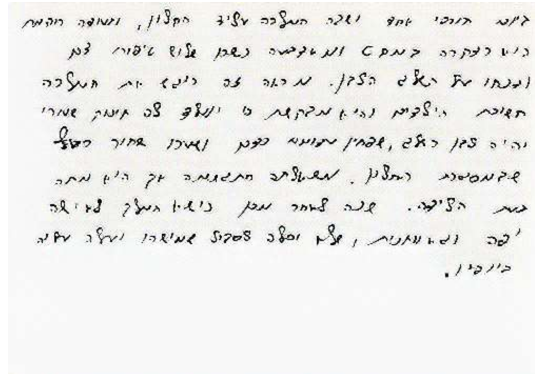
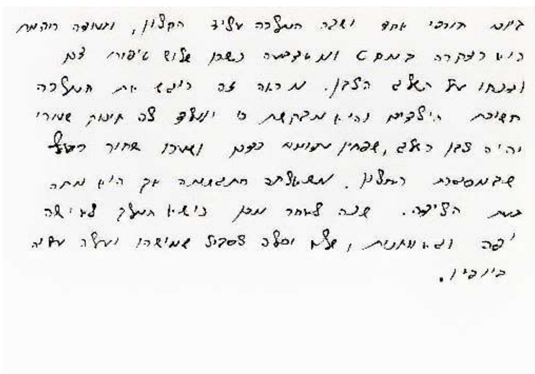
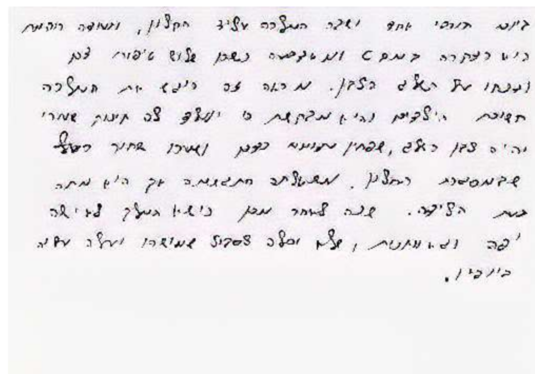
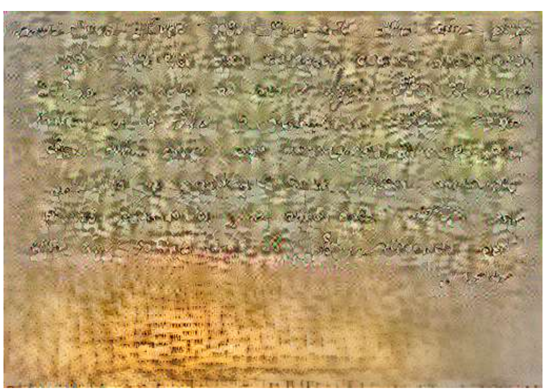
2.8.3.7. Weight 6

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.8.3.8. Weight 7

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.8.3.9. Weight 8

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.8.4. Discussion

As we can see, in this experiment we get worse results than experiments 1, 2 and 3, meaning the binary does not improve our results.

2.9. Experiment 6

As we saw in experiment 5 the results were not that good let's see if the text in the style has an effect on these results.s

2.9.1. Content Input

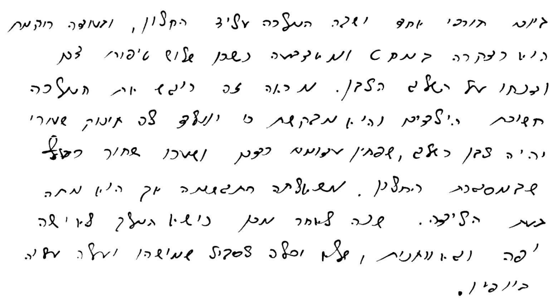
This experiment used a binary modern hebrew handwritten document for content.

2.9.2. Style Input

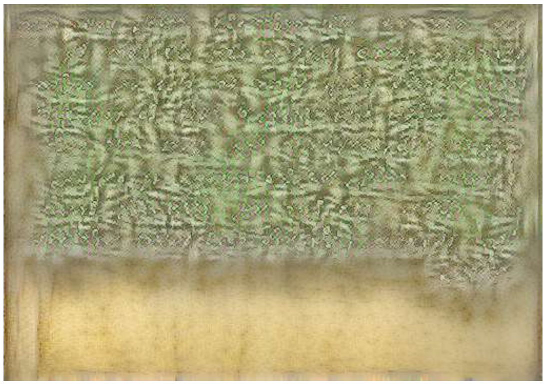
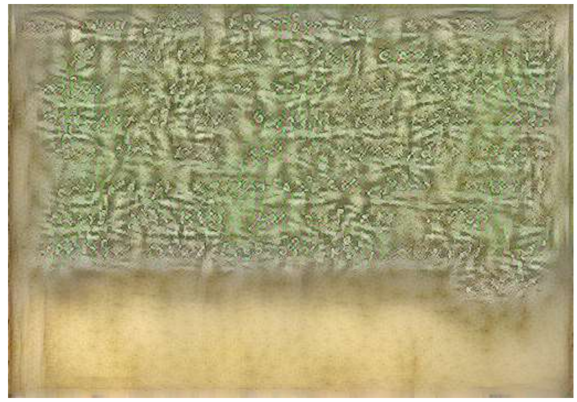
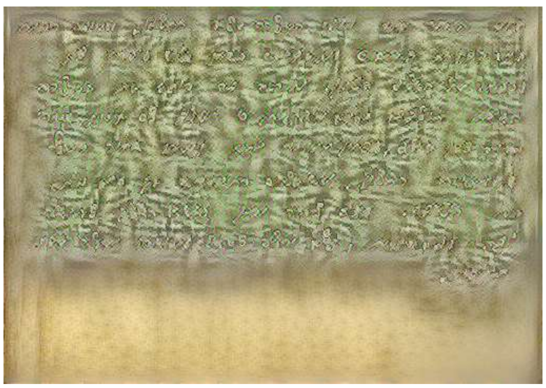
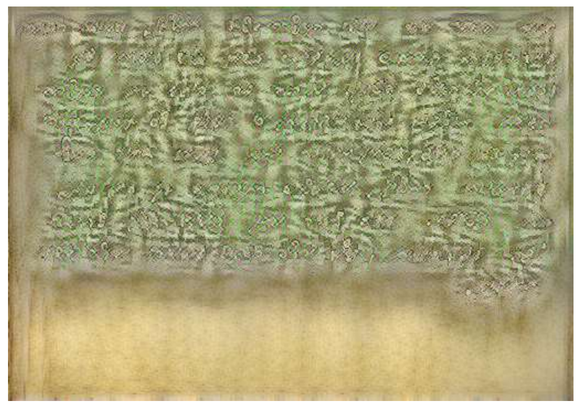
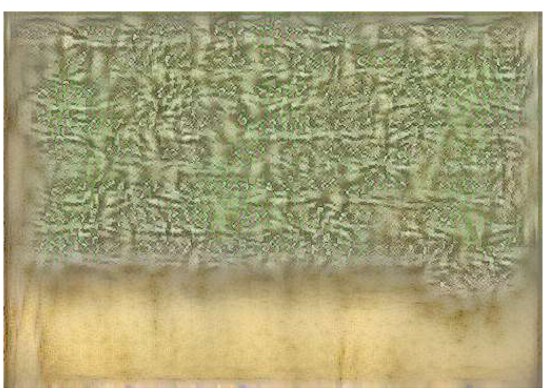
This experiment used a hebrew middle age document without text for style.

2.9.3. Results

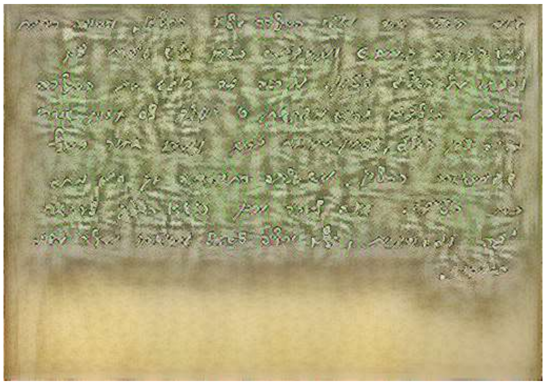
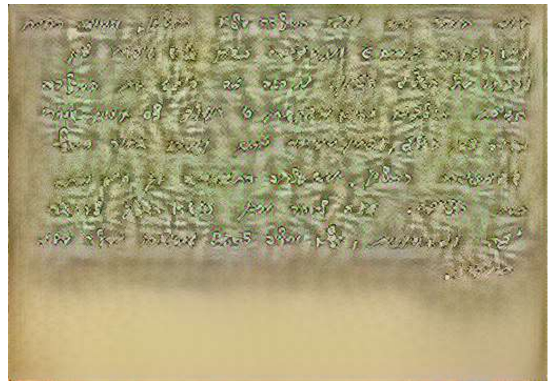
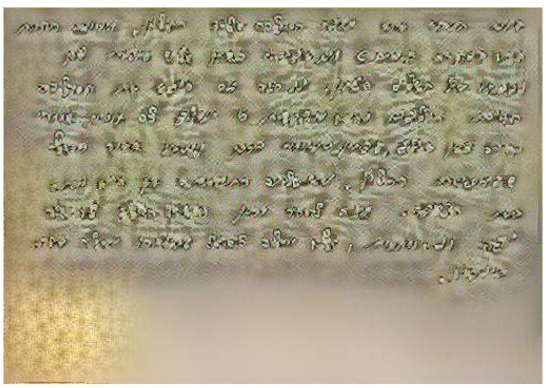
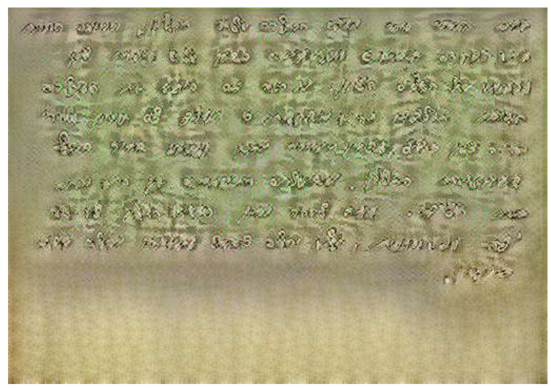
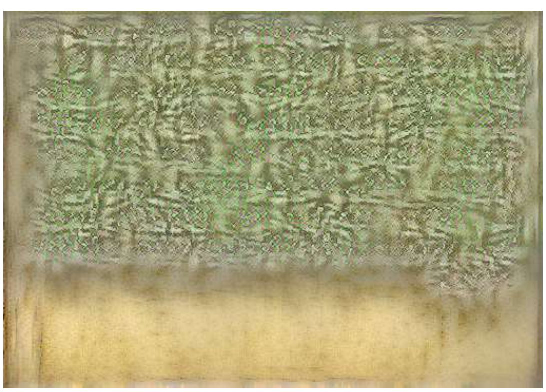
We will show samples of the results with each weight and content layer (style layers are the same).

Content image	Style image
	

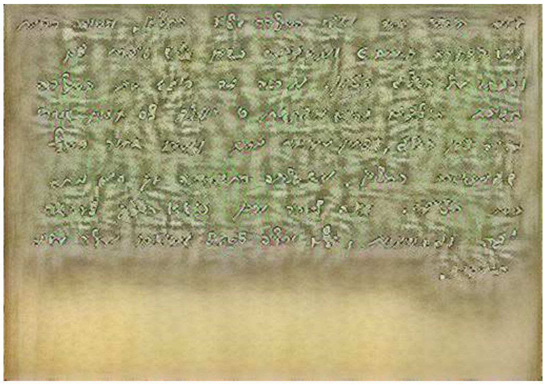
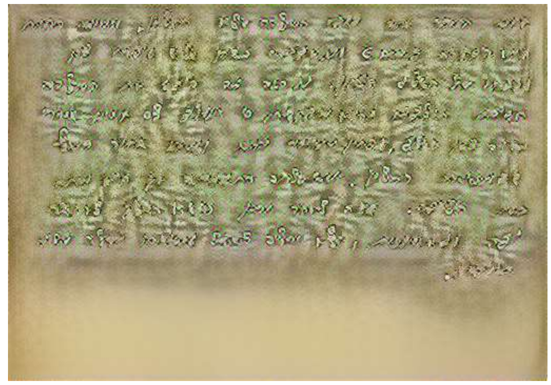
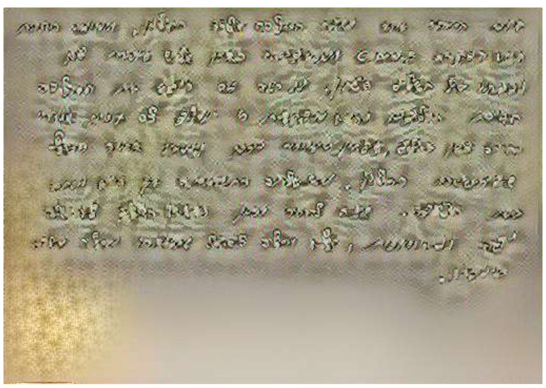
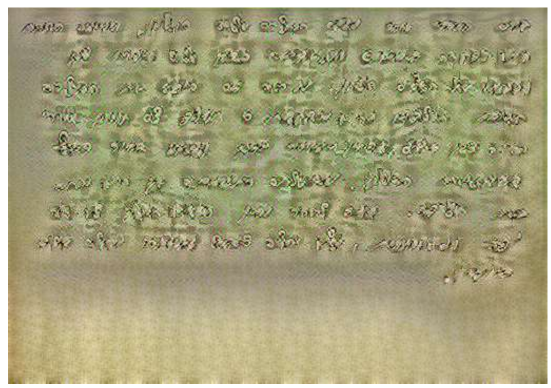
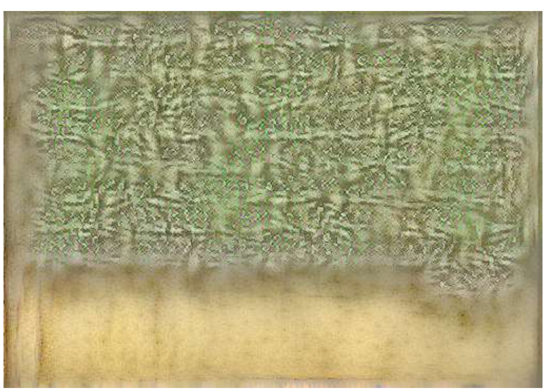
2.9.3.1. Weight 0

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

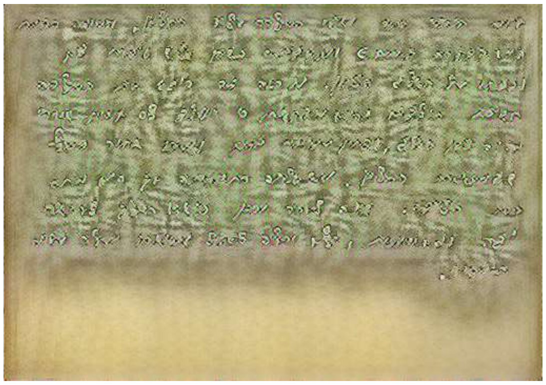
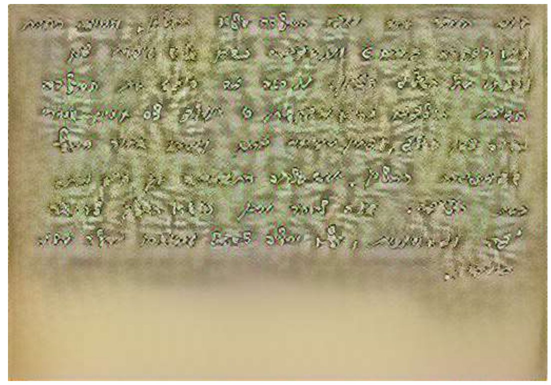
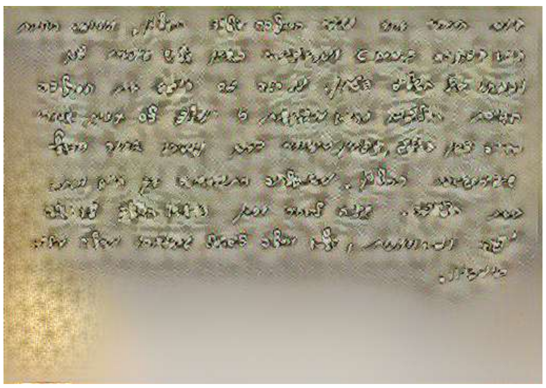
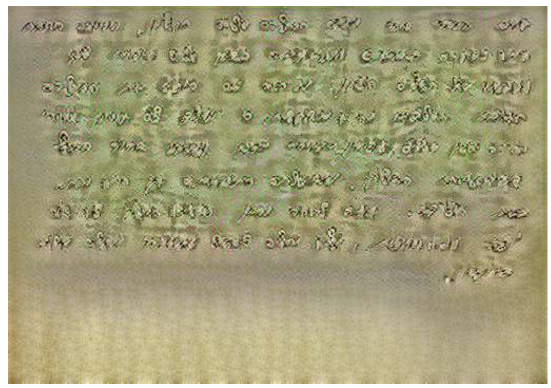
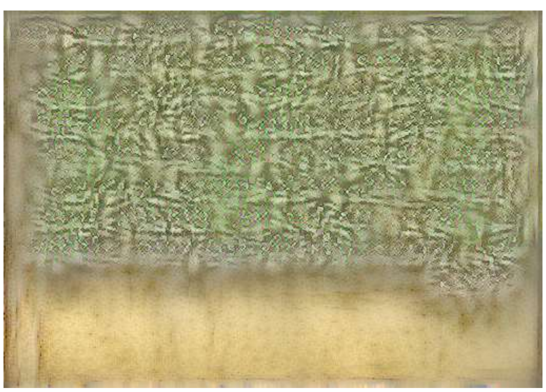
2.9.3.2. Weight 1

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

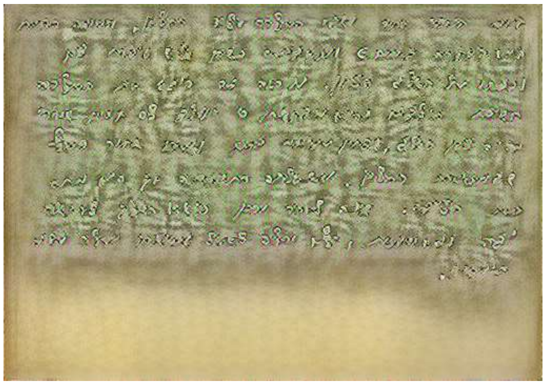
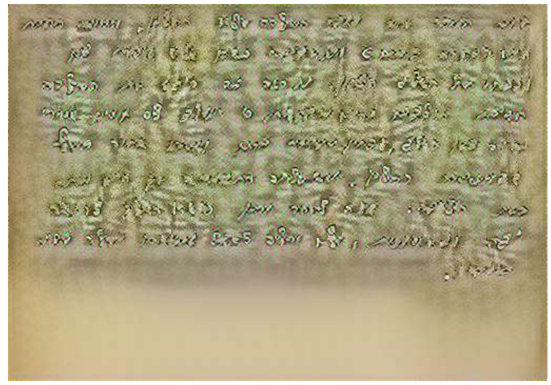
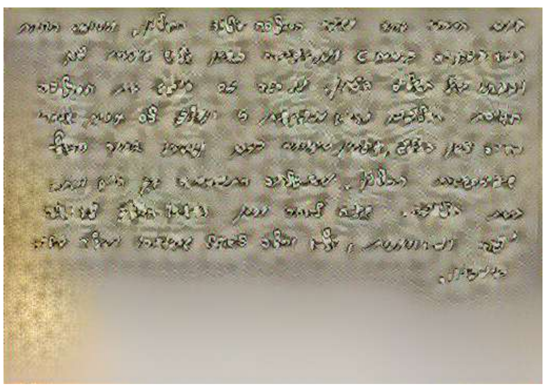
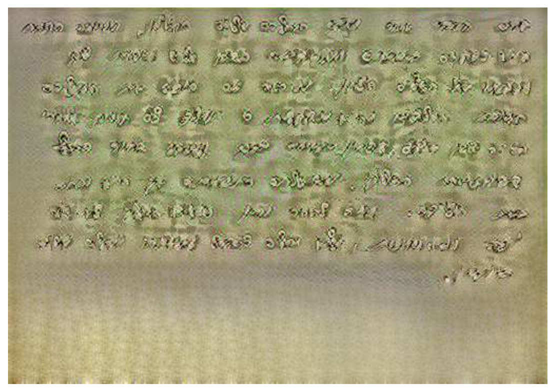
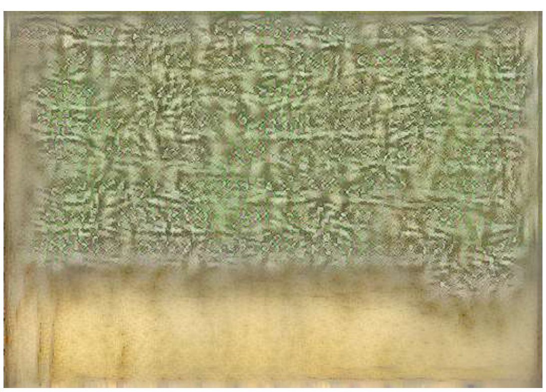
2.9.3.3. Weight 2

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

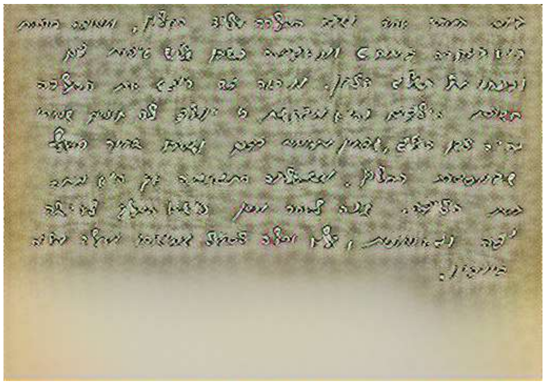
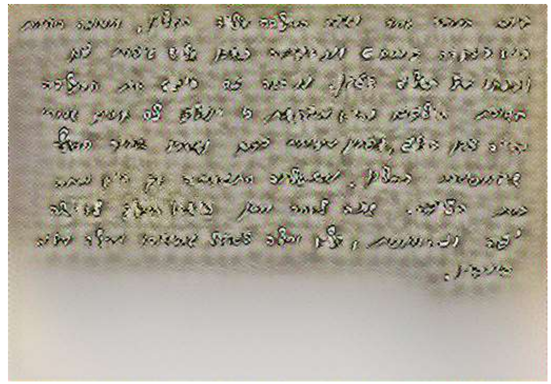
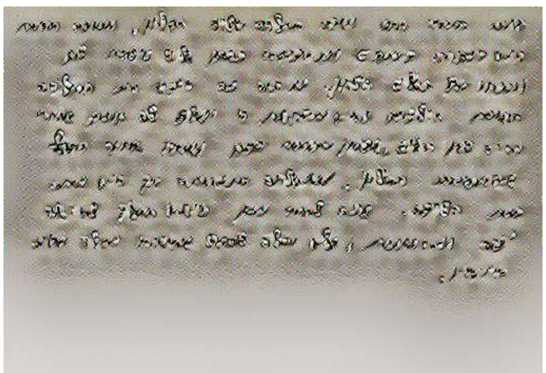
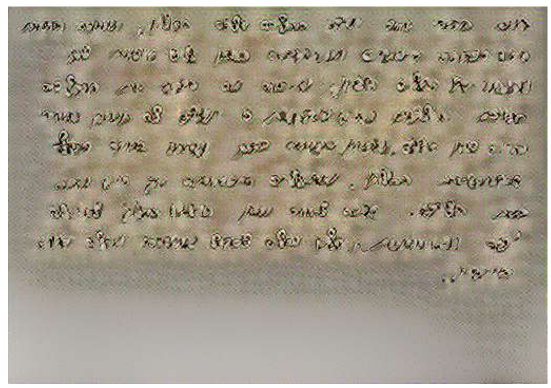
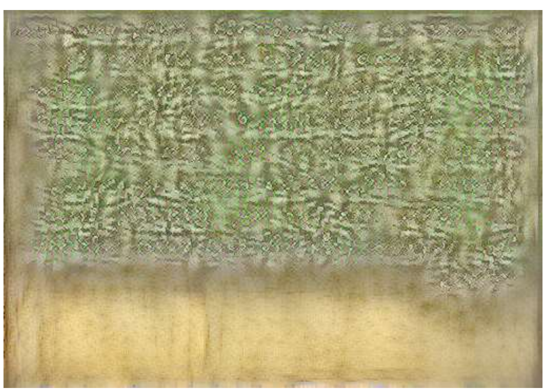
2.9.3.4. Weight 3

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

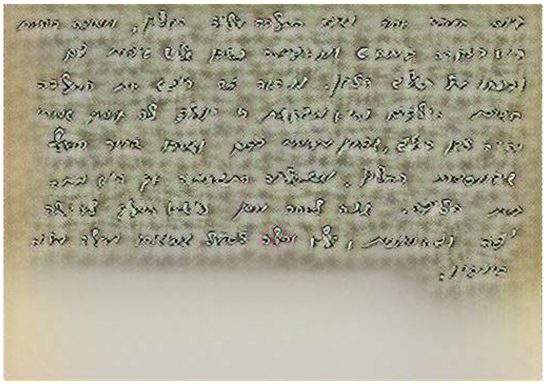
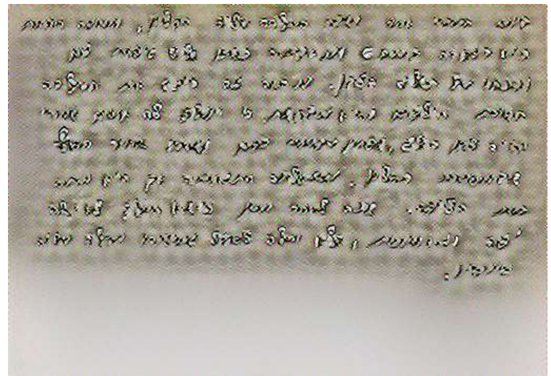
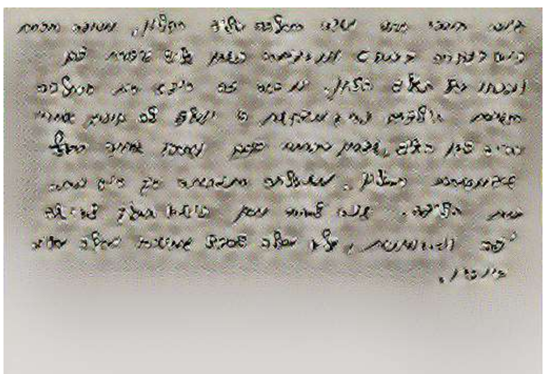
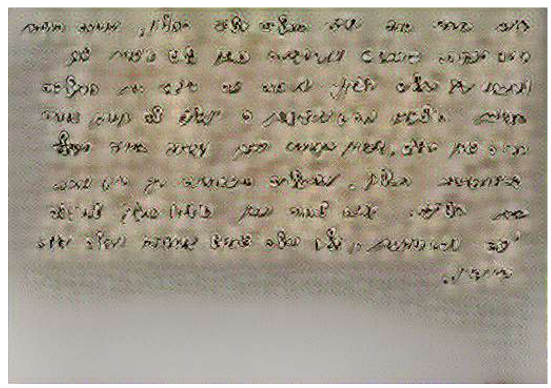
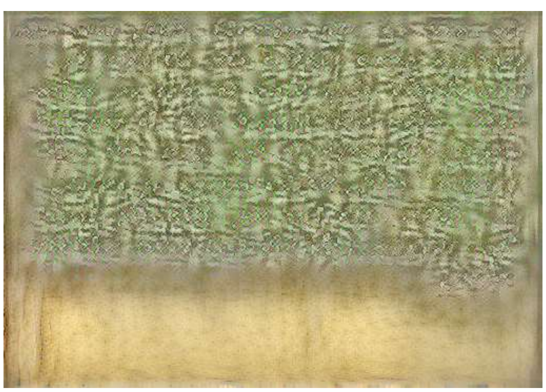
2.9.3.5. Weight 4

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

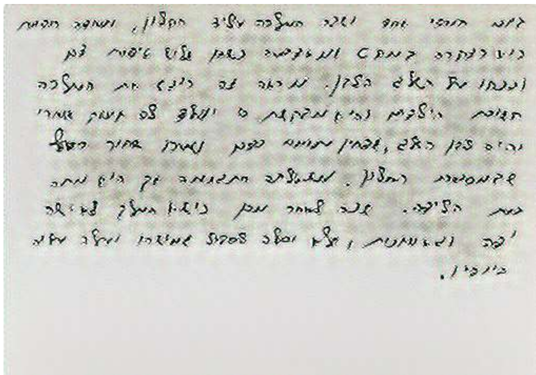
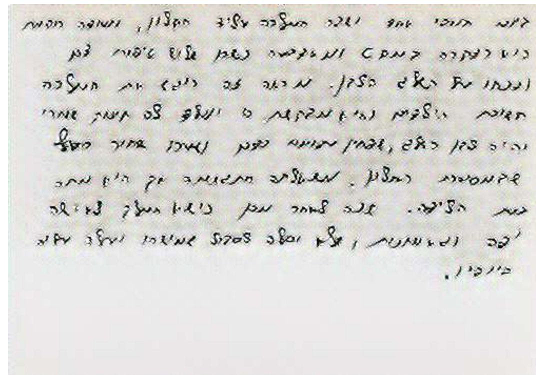
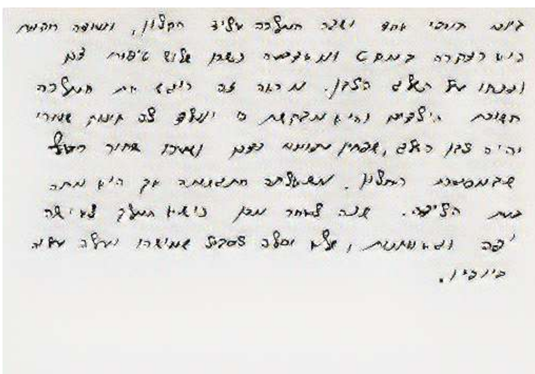
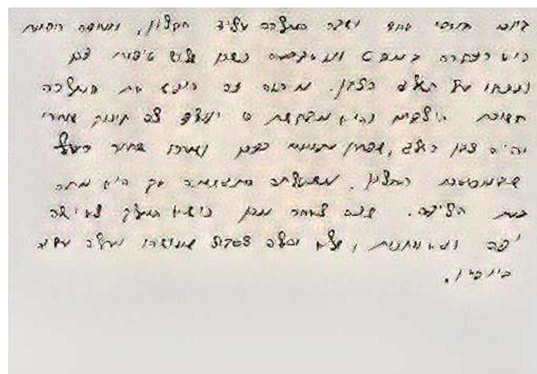
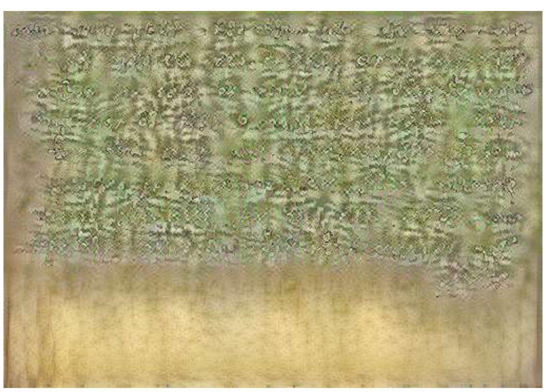
2.9.3.6. Weight 5

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

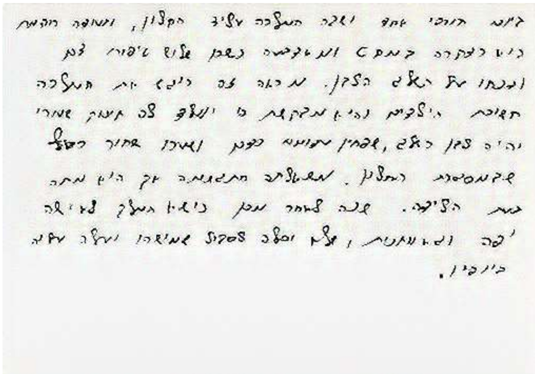
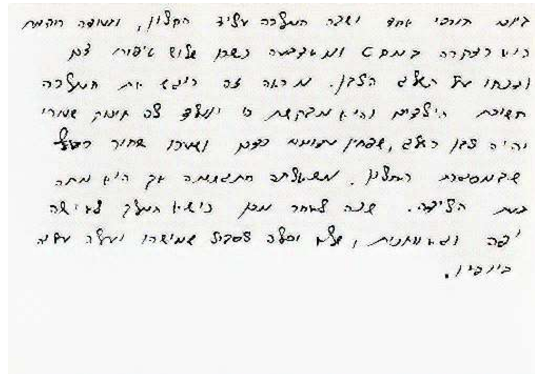
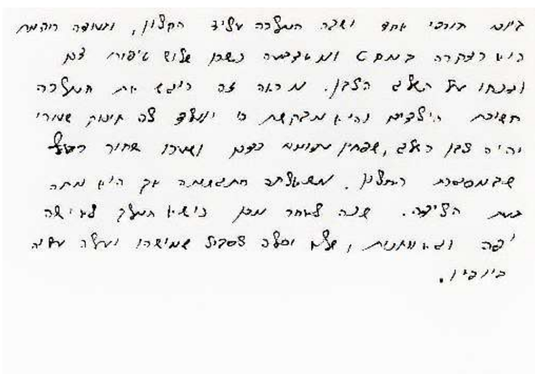
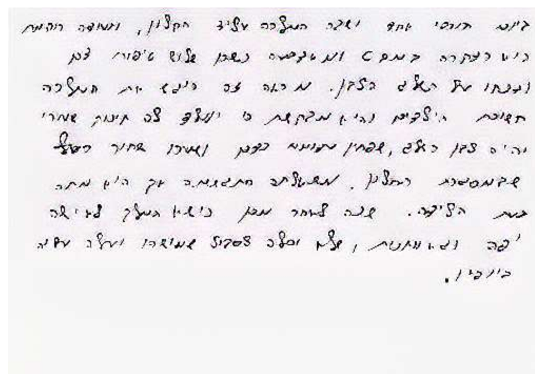
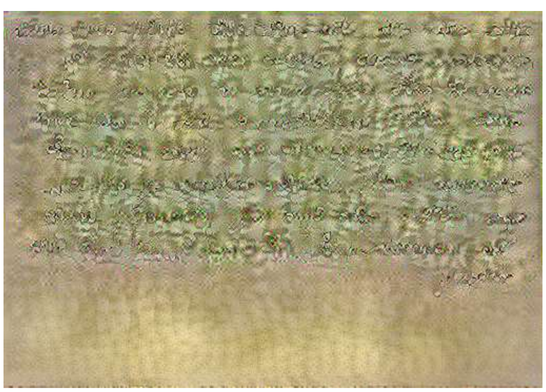
2.9.3.7. Weight 6

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.9.3.8. Weight 7

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.9.3.9. Weight 8

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.9.4. Discussion

As we can see, in this experiment we get better results than experiment 5 but still worse than experiments 1,2 and 3, meaning the binary does not improve our results.

2.10. Experiment 7

Now let's try to combine dilation and binarization to see the effect on results.

2.10.1. Content Input

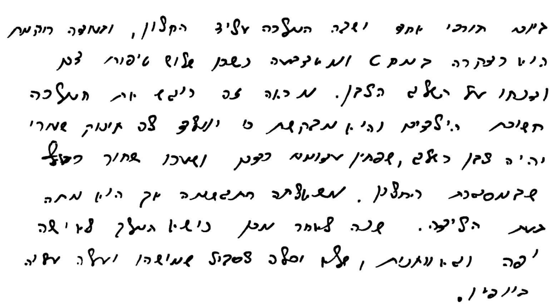
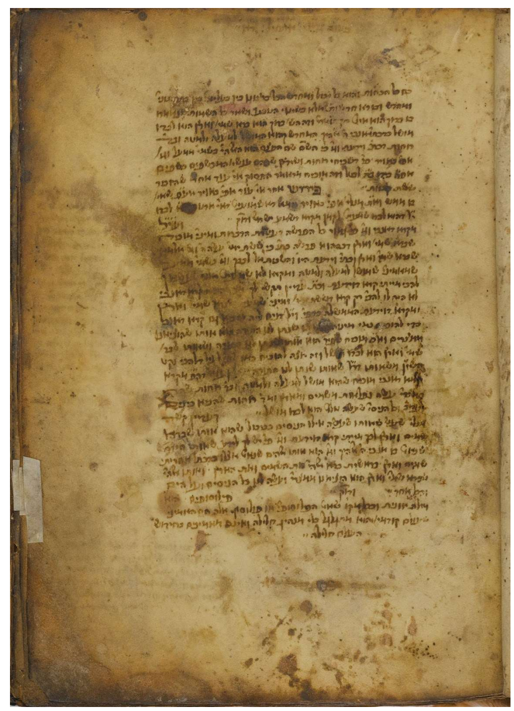
This experiment used a dilated binary modern hebrew handwritten document for content.

2.10.2. Style Input

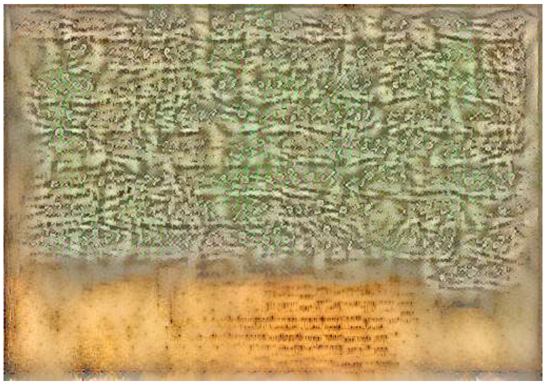
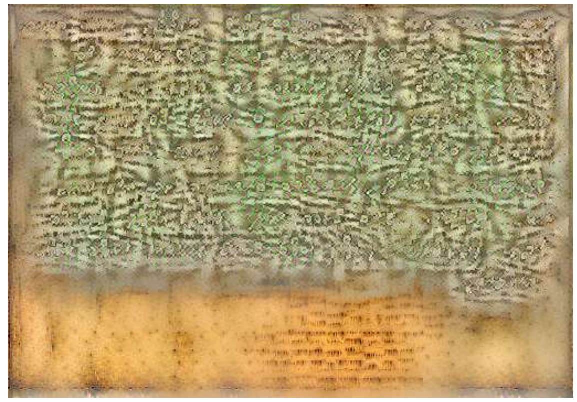
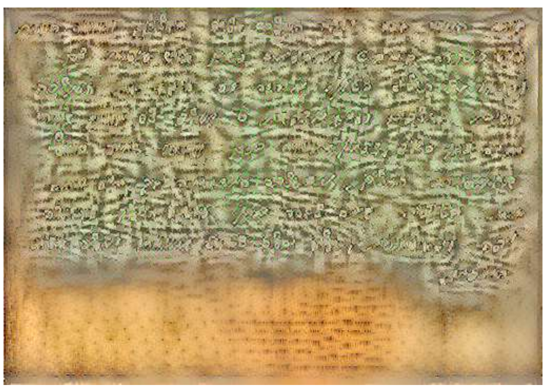
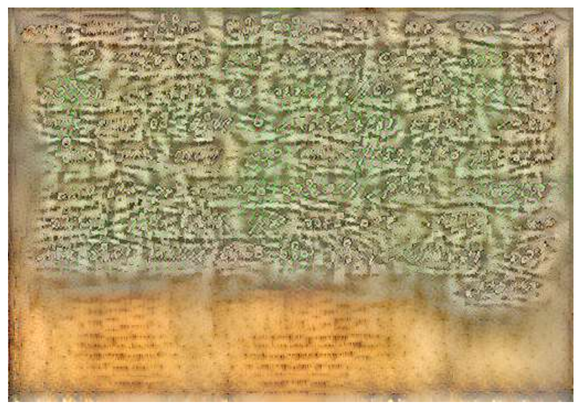
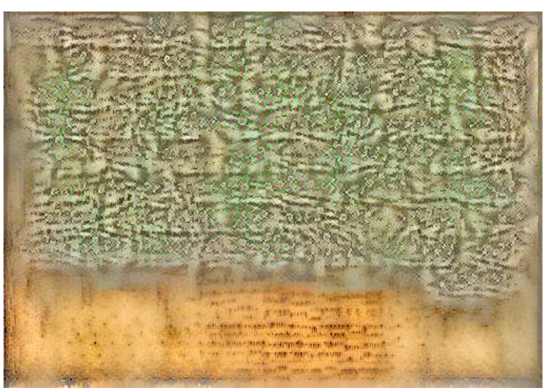
This experiment used a hebrew middle age document with text for style.

2.10.3. Results

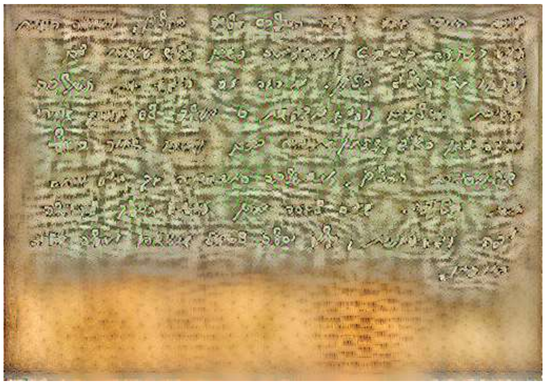
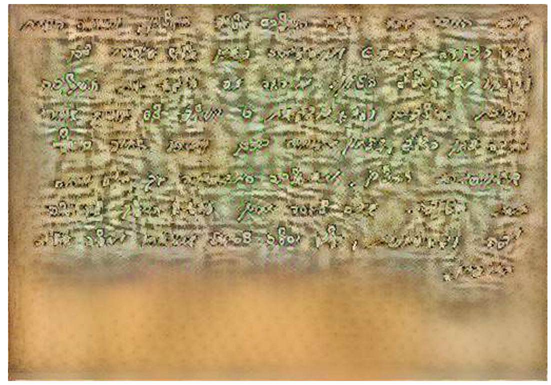
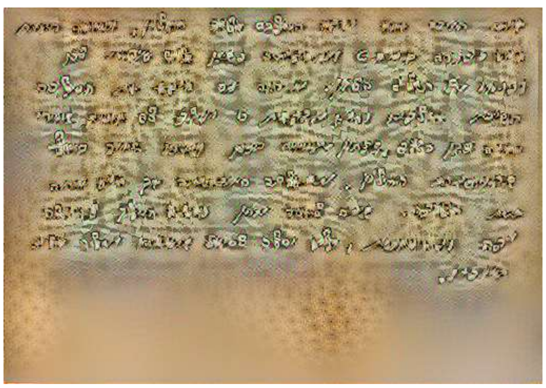
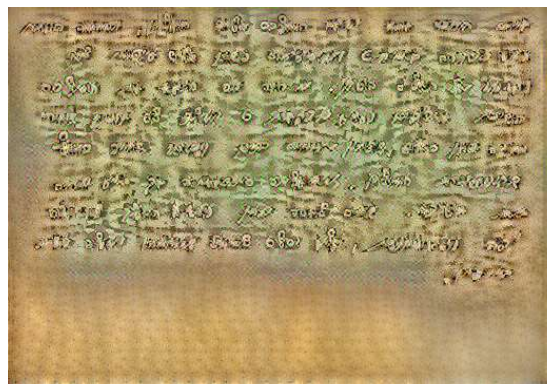
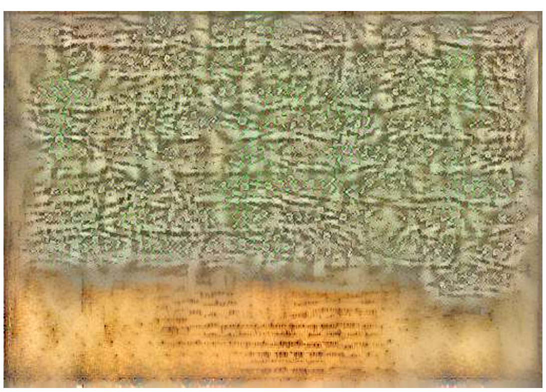
We will show samples of the results with each weight and content layer (style layers are the same).

Content image	Style image
	

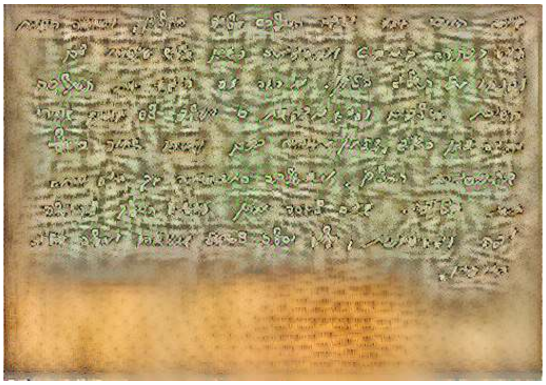
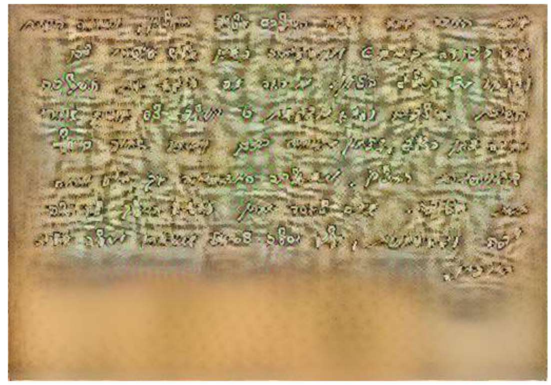
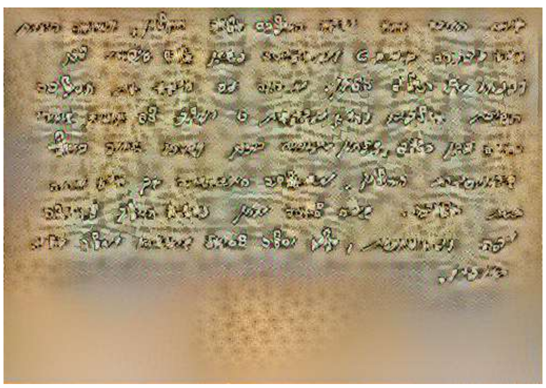
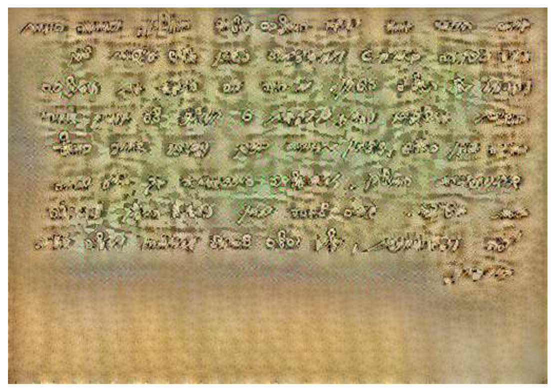
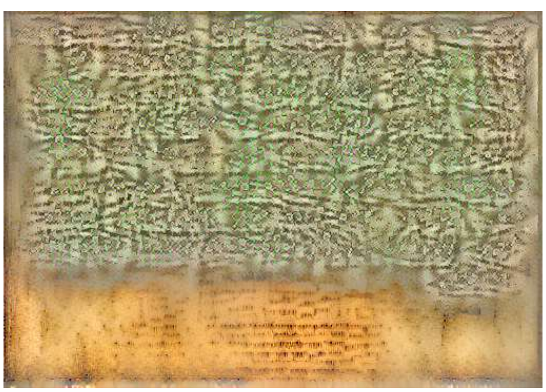
2.10.3.1. Weight 0

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

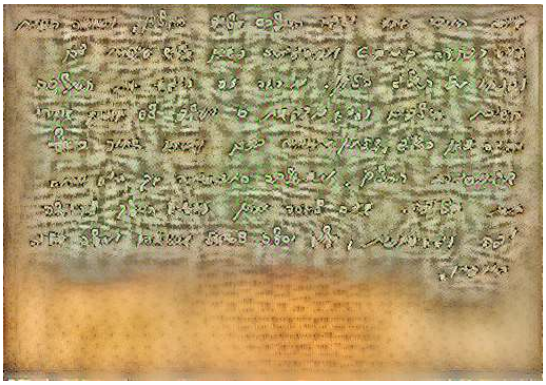
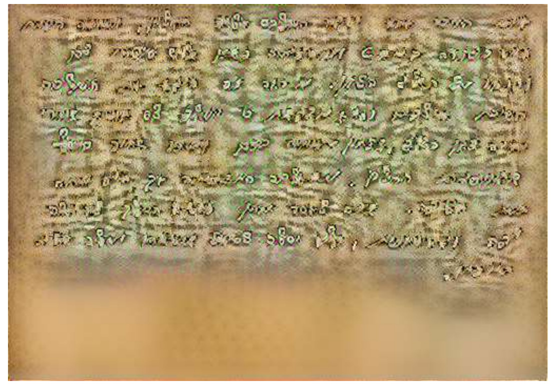
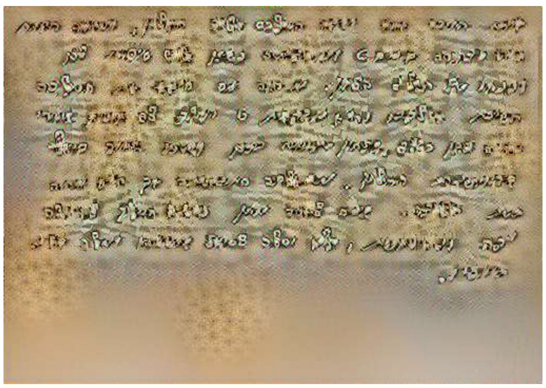
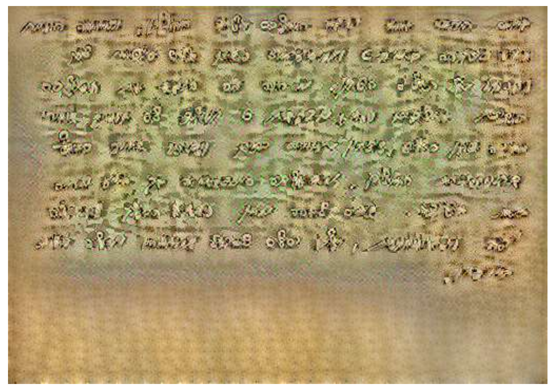
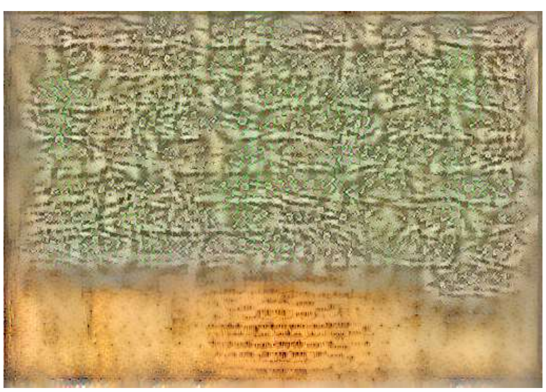
2.10.3.2. Weight 1

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

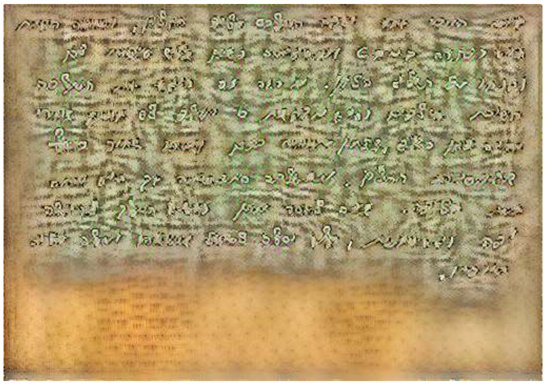
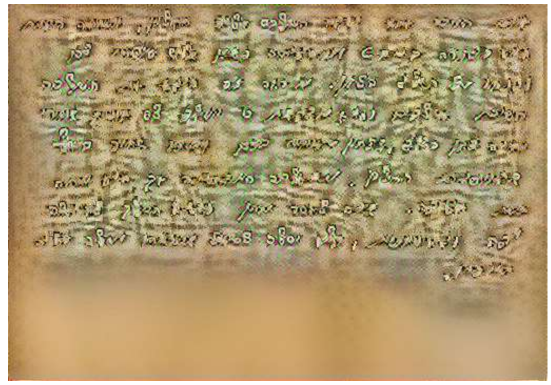
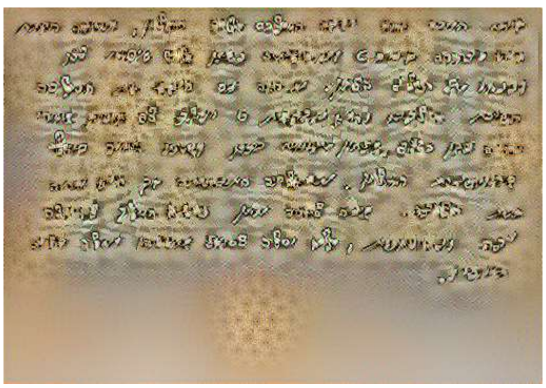
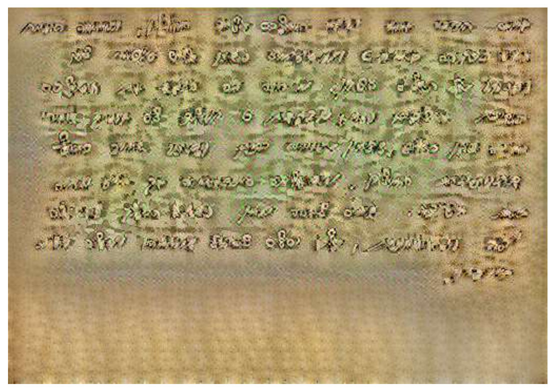
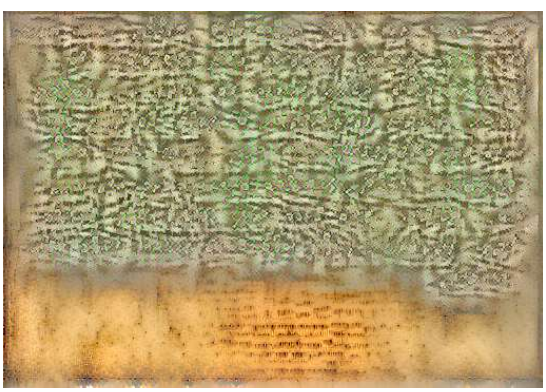
2.10.3.3. Weight 2

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

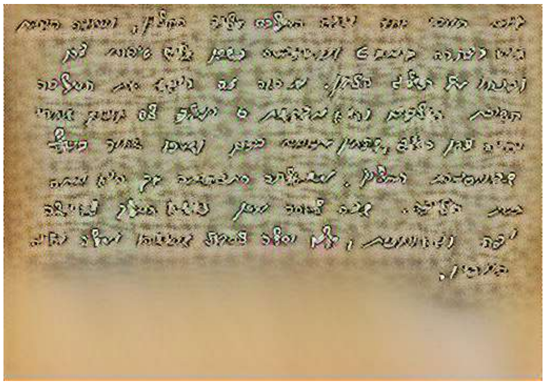
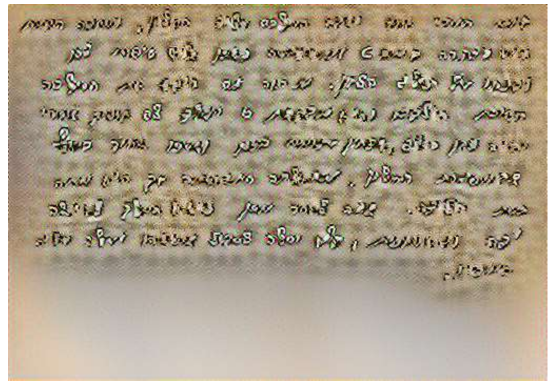
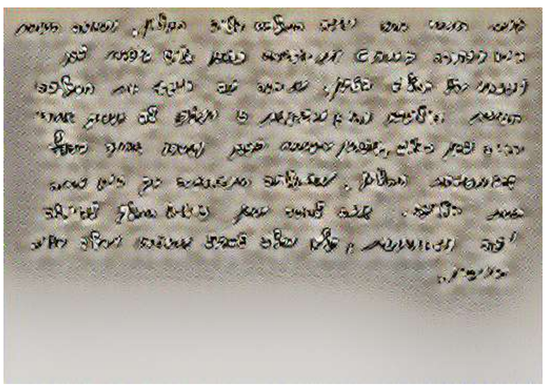
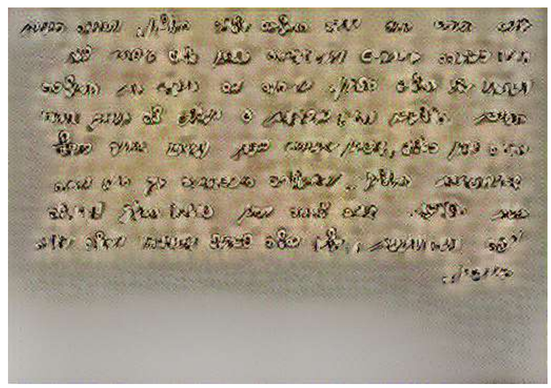
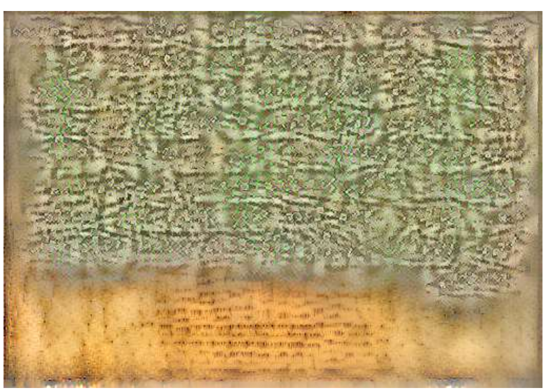
2.10.3.4. Weight 3

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

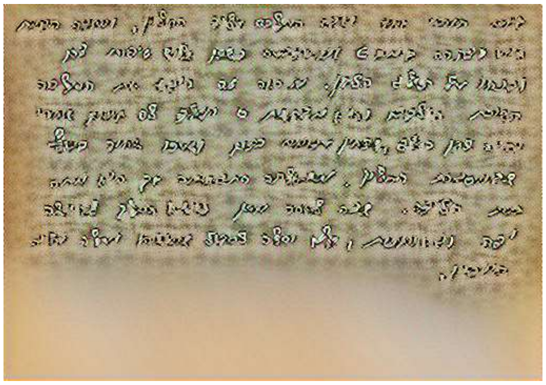
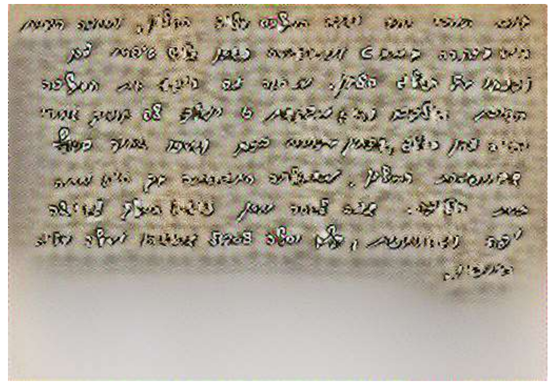
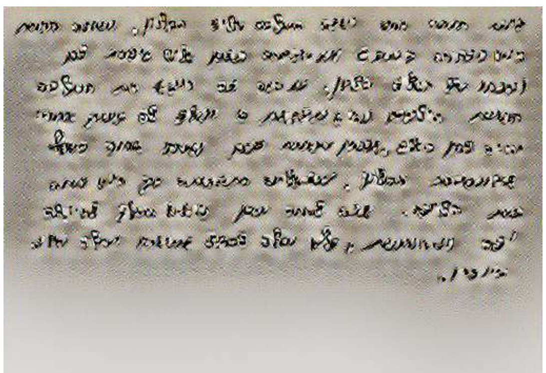
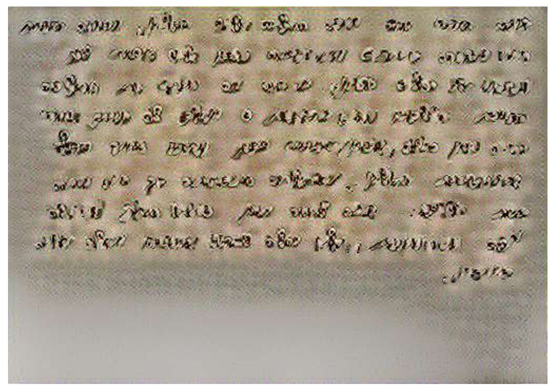
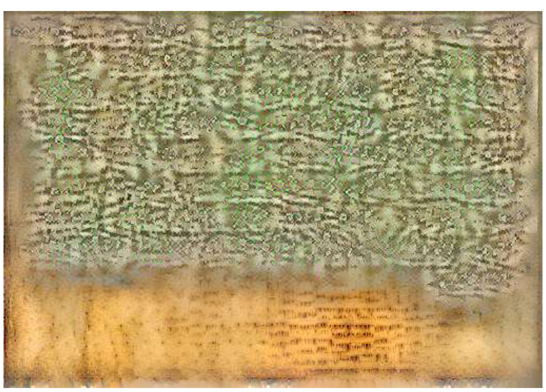
2.10.3.5. Weight 4

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

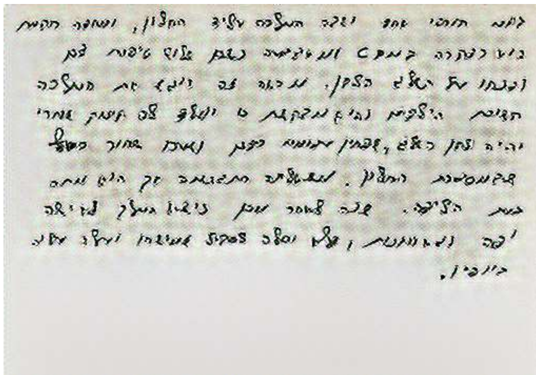
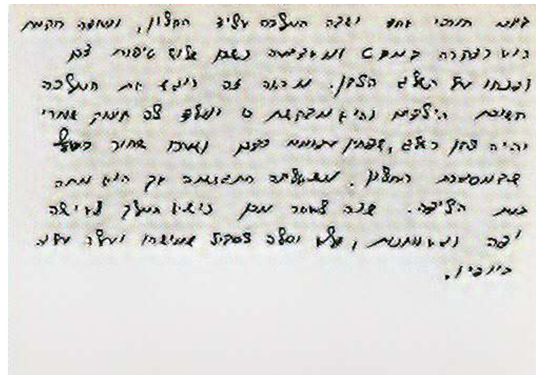
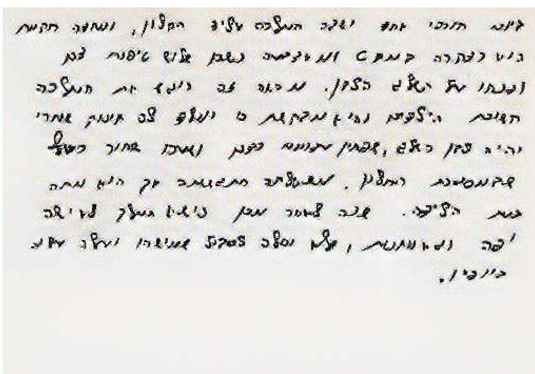
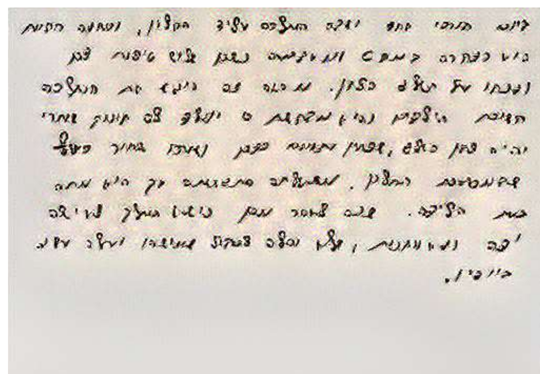
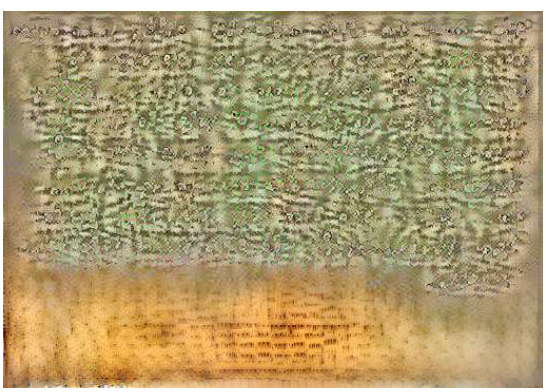
2.10.3.6. Weight 5

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

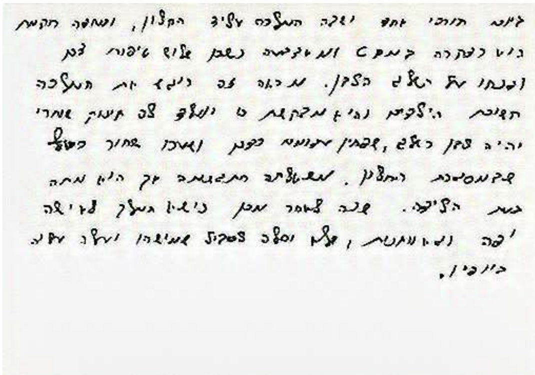
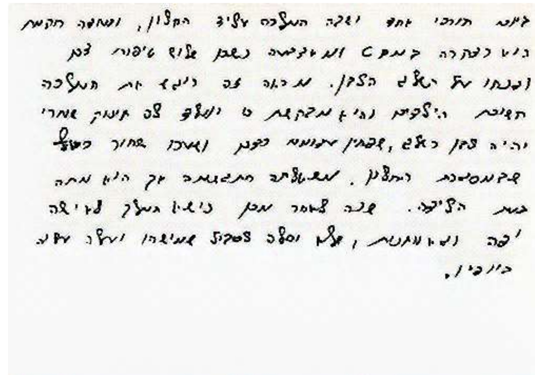
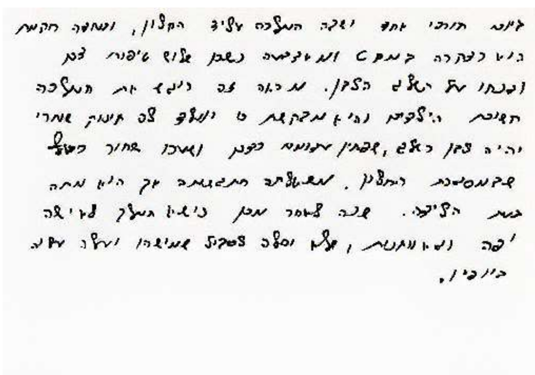
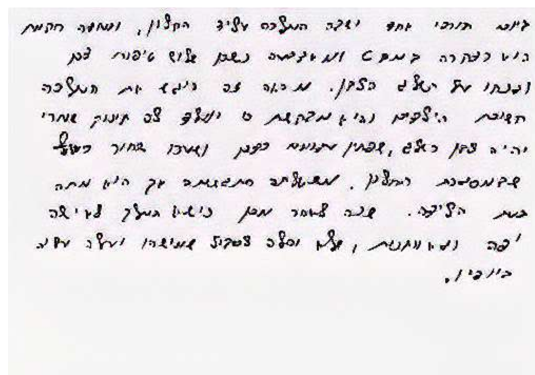
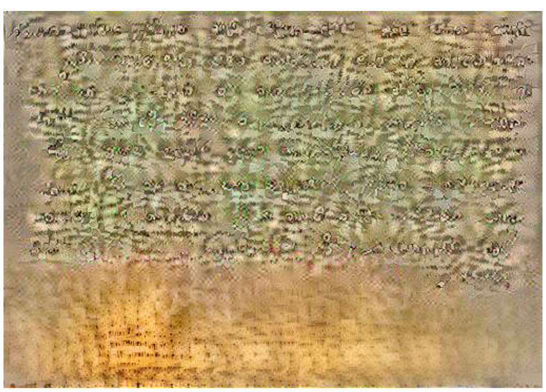
2.10.3.7. Weight 6

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.10.3.8. Weight 7

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.10.3.9. Weight 8

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.10.4. Discussion

As we can see, in this experiment the only weights that return good results are weights 5 and 6, and also the content layers have changed. The results have not been improved, style isn't being passed well, but we can see an improvement in content.

2.11. Experiment 8

Next we test if removing text from the style will give us a better result.

2.11.1. Content Input

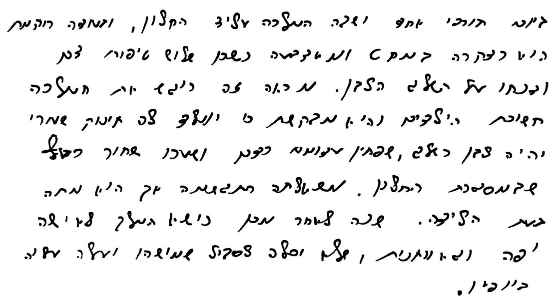
This experiment used a dilated binary modern hebrew handwritten document for content.

2.11.2. Style Input

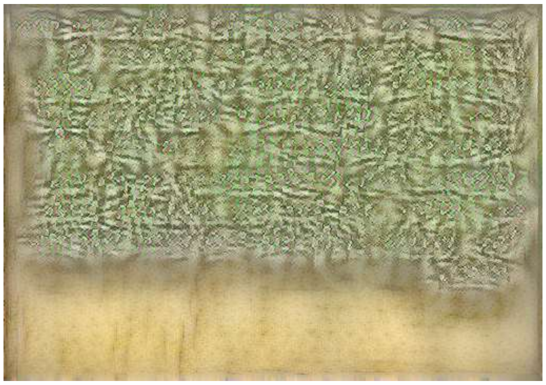
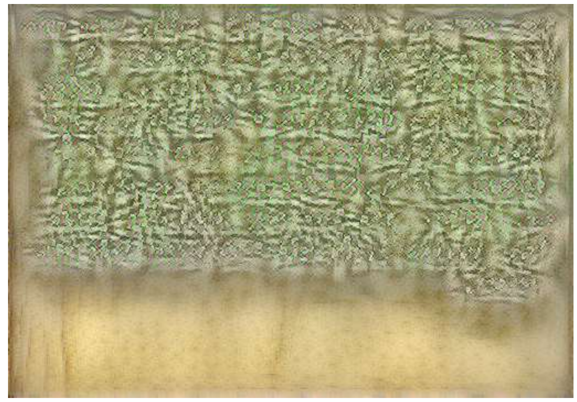
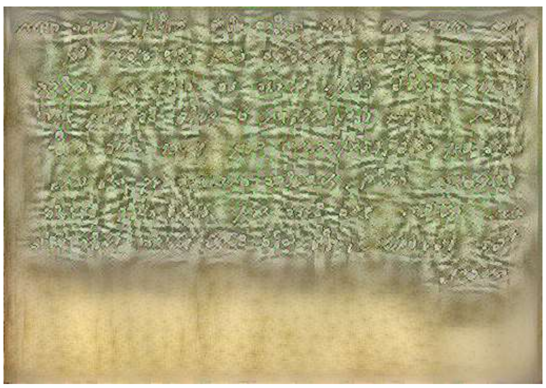
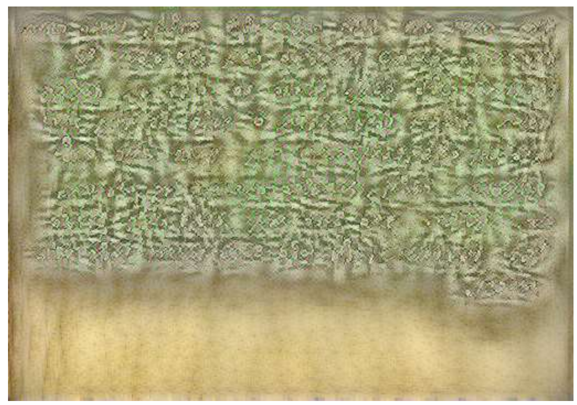
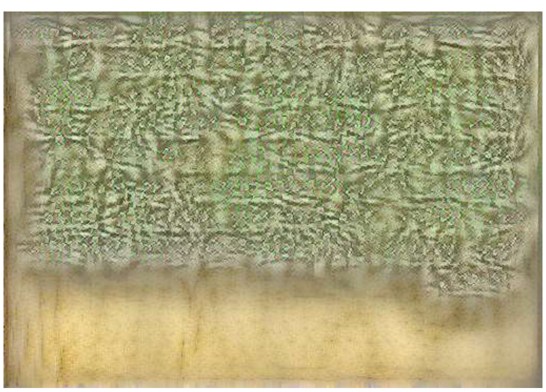
This experiment used a hebrew middle age document without text for style.

2.11.3. Results

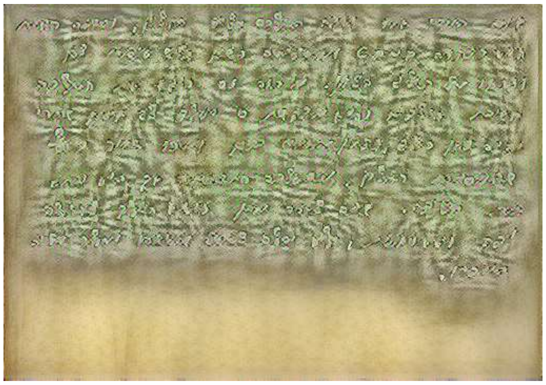
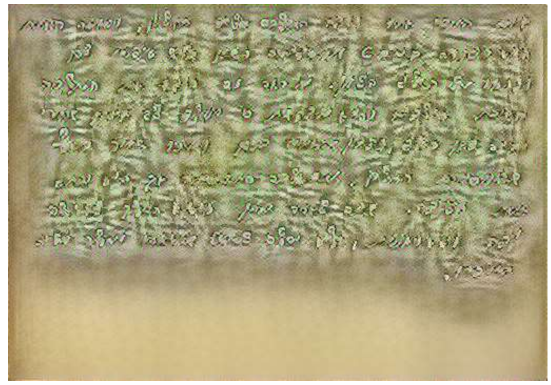
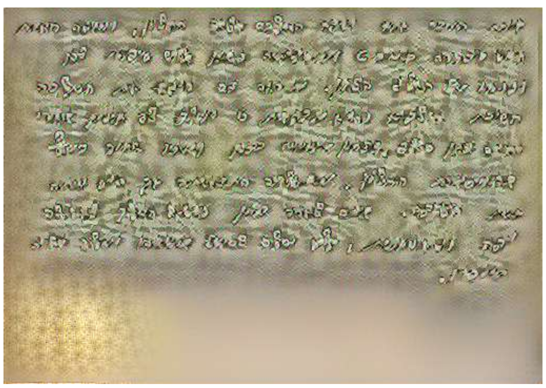
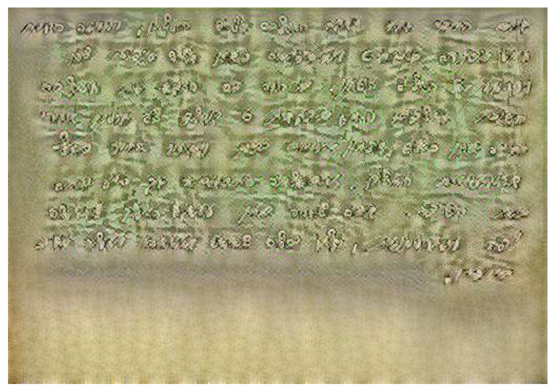
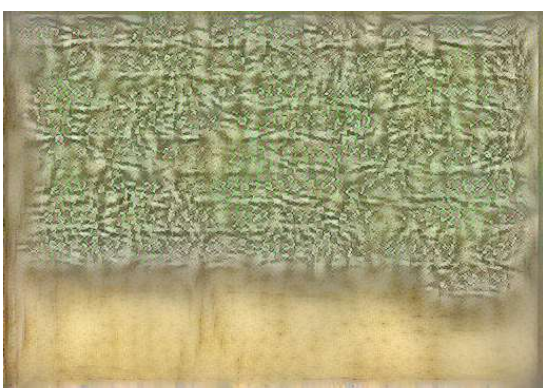
We will show samples of the results with each weight and content layer (style layers are the same).

Content image	Style image
	

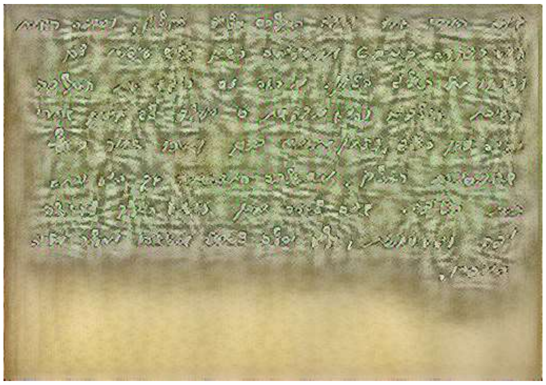
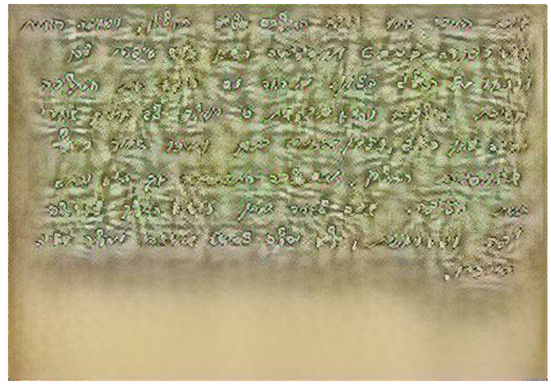
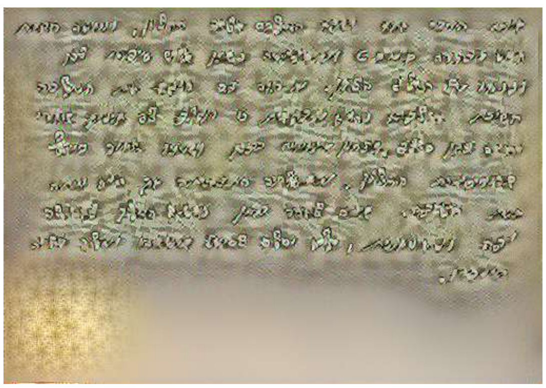
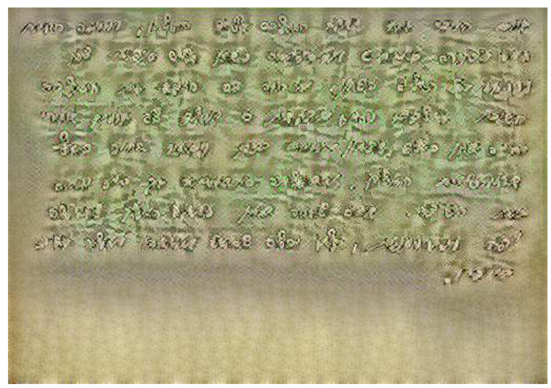
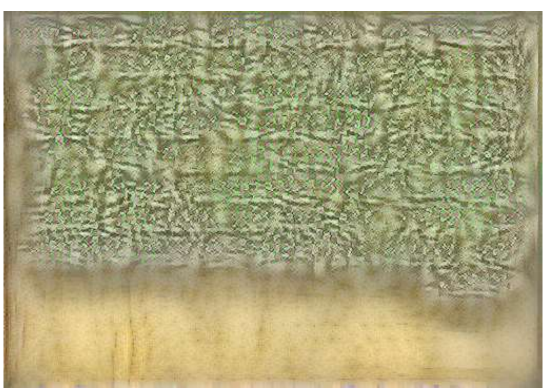
2.11.3.1. Weight 0

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

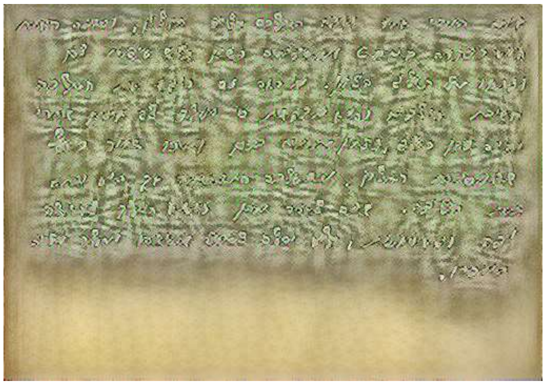
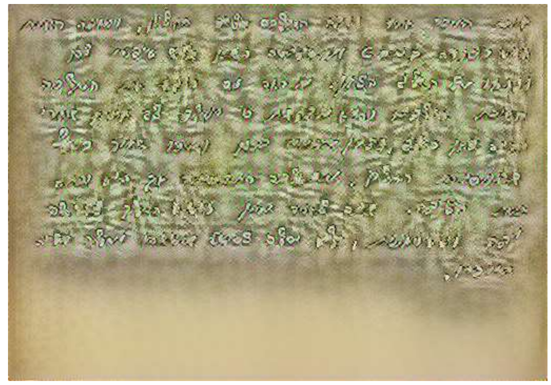
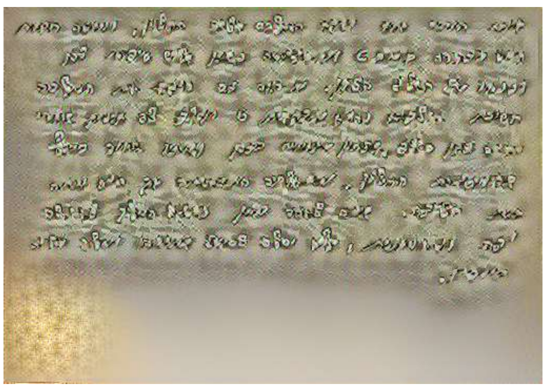
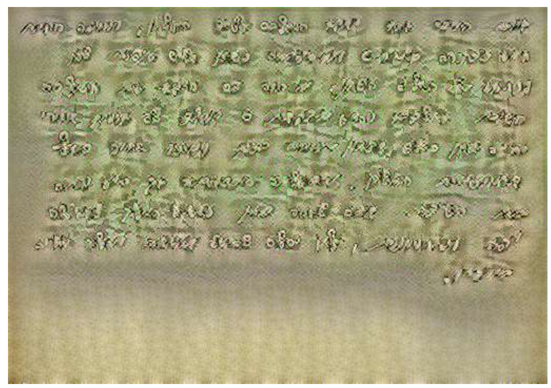
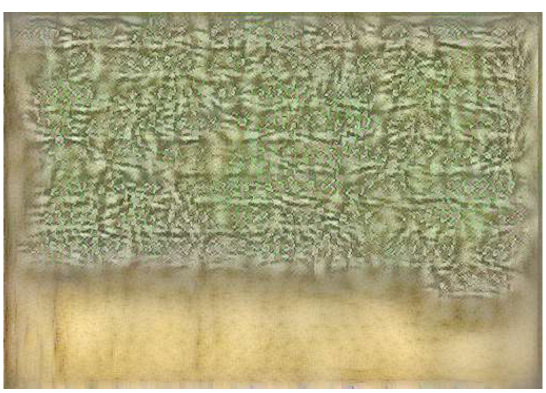
2.11.3.2. Weight 1

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

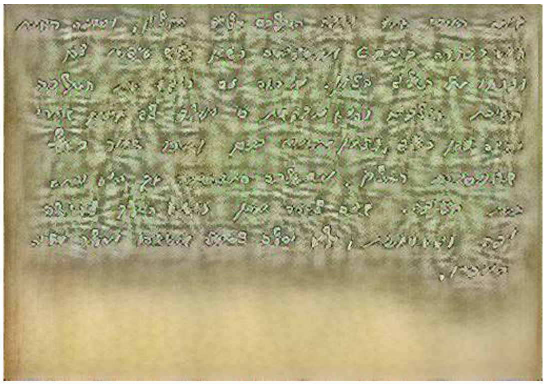
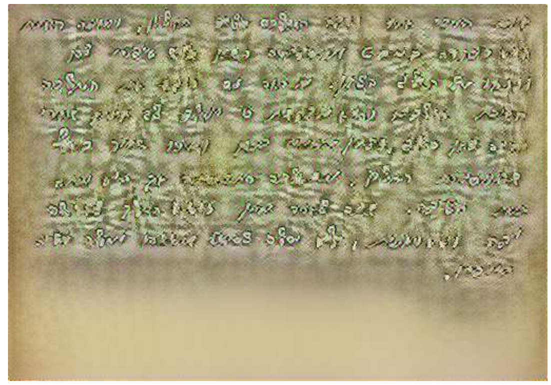
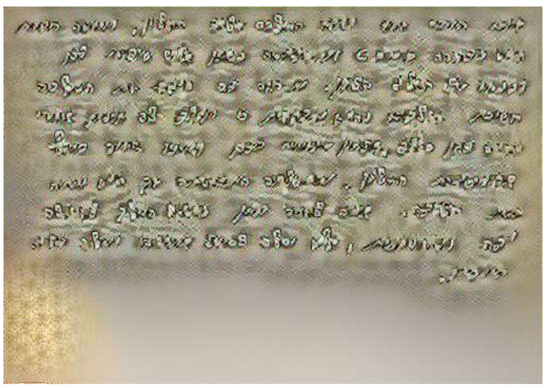
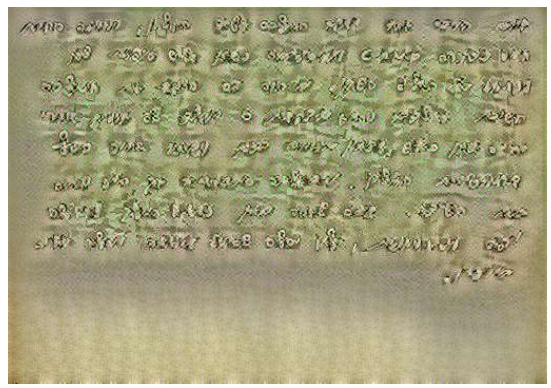
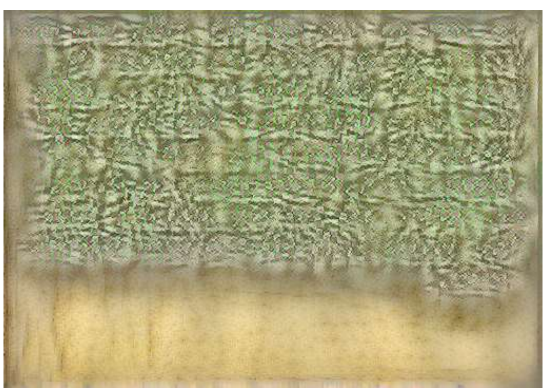
2.11.3.3. Weight 2

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

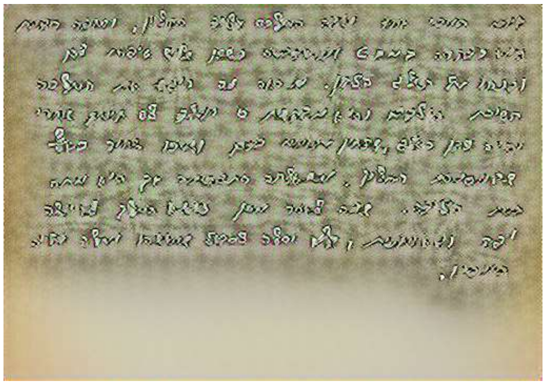
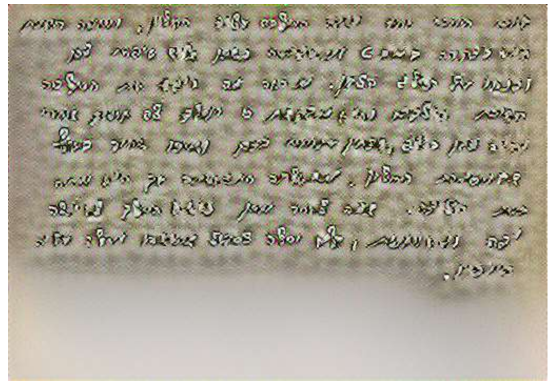
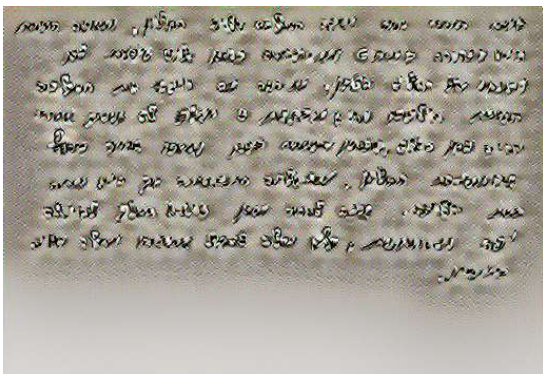
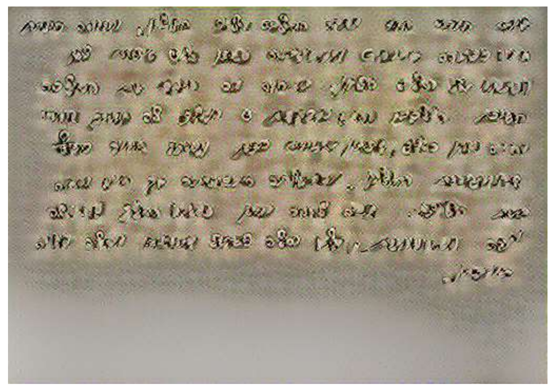
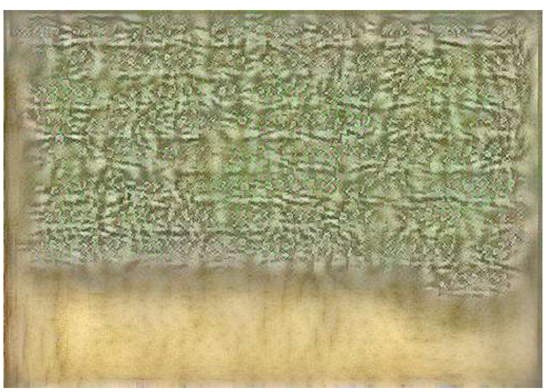
2.11.3.4. Weight 3

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

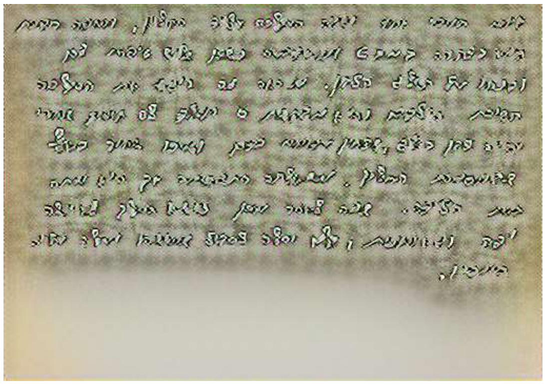
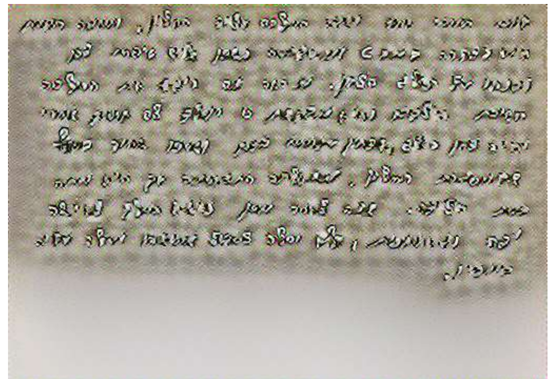
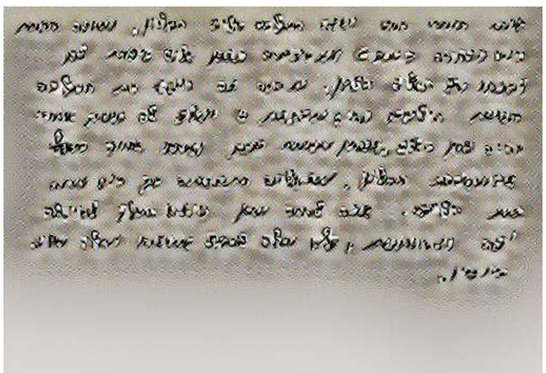
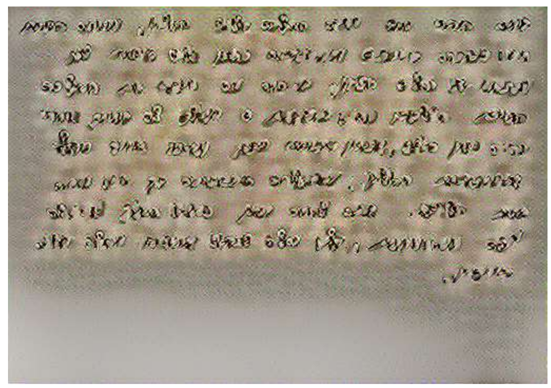
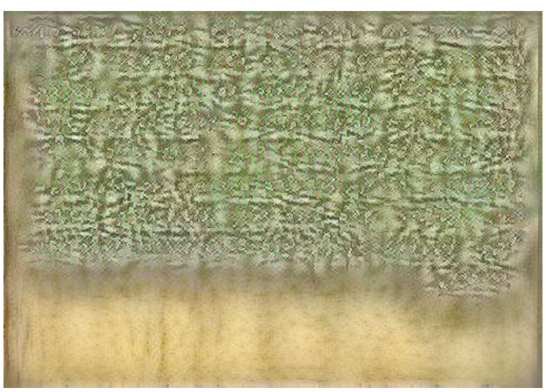
2.11.3.5. Weight 4

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

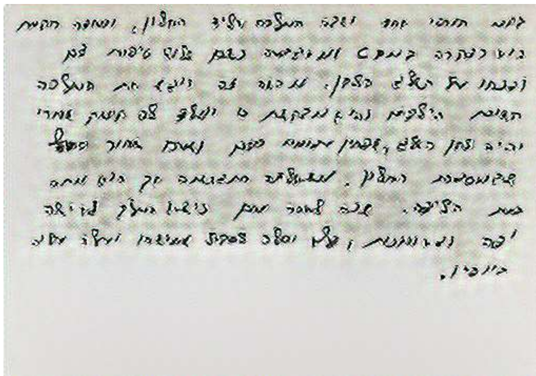
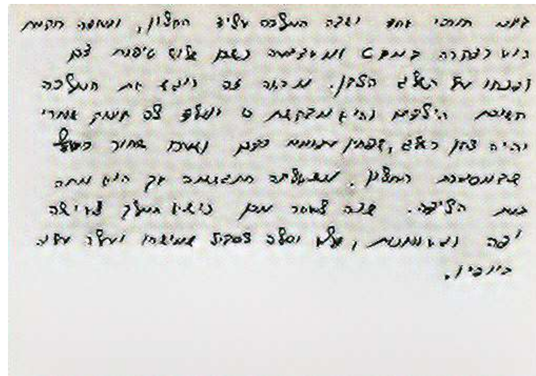
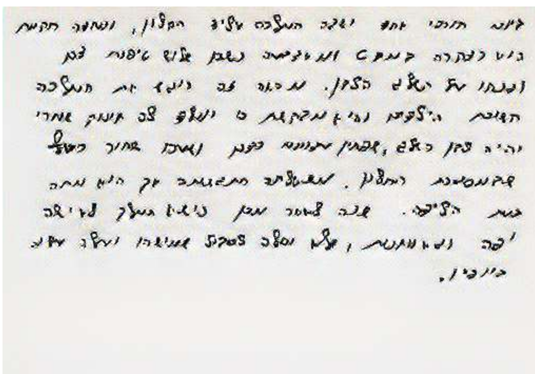
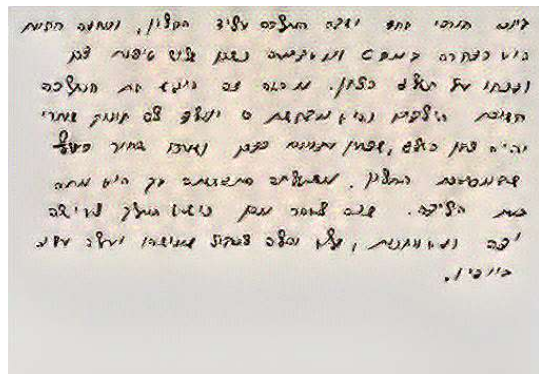
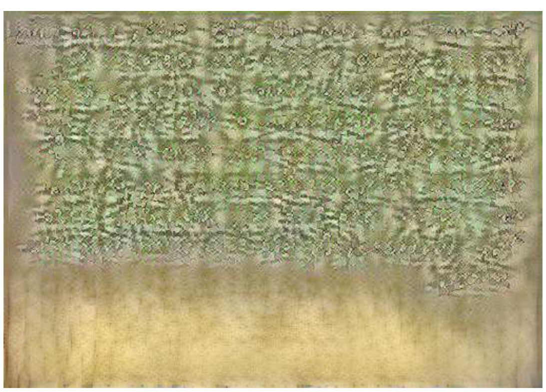
2.11.3.6. Weight 5

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

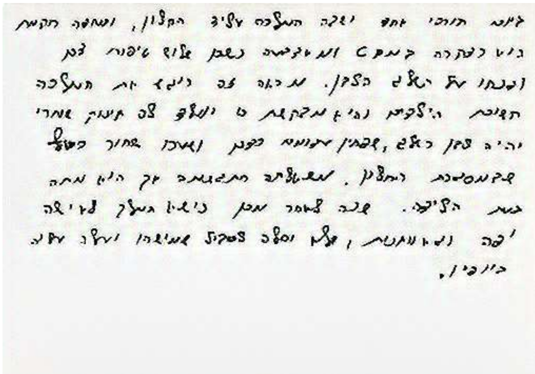
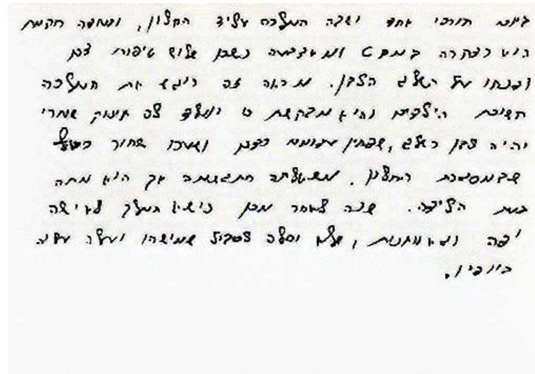
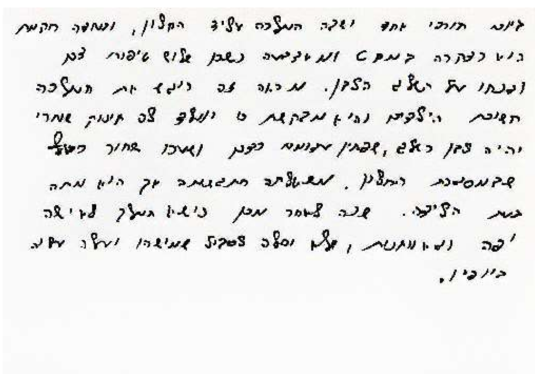
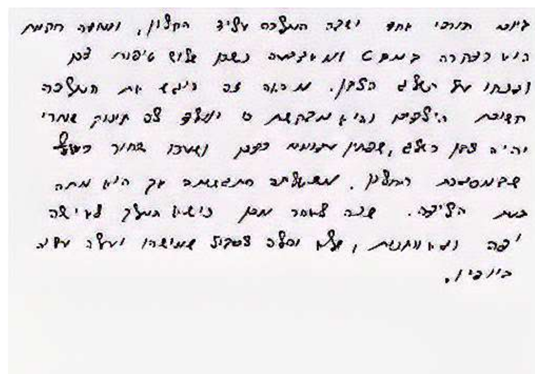
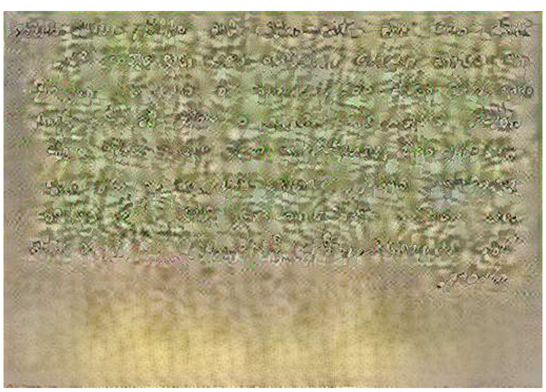
2.11.3.7. Weight 6

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.11.3.8. Weight 7

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.11.3.9. Weight 8

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.11.4. Discussion

As we can see, in this experiment the only weights that return good results are weights 5 and 6, and also the content layers have changed. The results have not been improved, style isn't being passed well, but we can see an improvement in content.

2.12. Experiment 9

Now, we test if adding to the content document a background created by averaging a pixels color from the style document will give us a better result.

2.12.1. Content Input

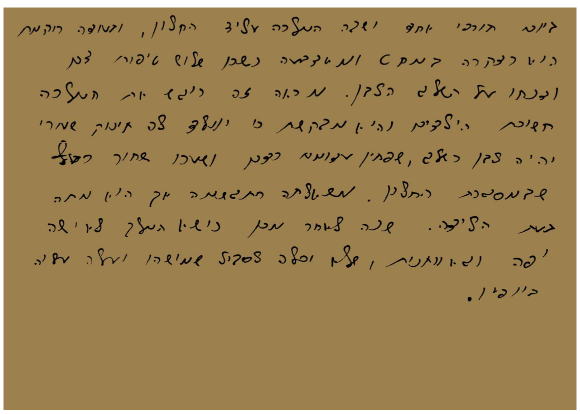
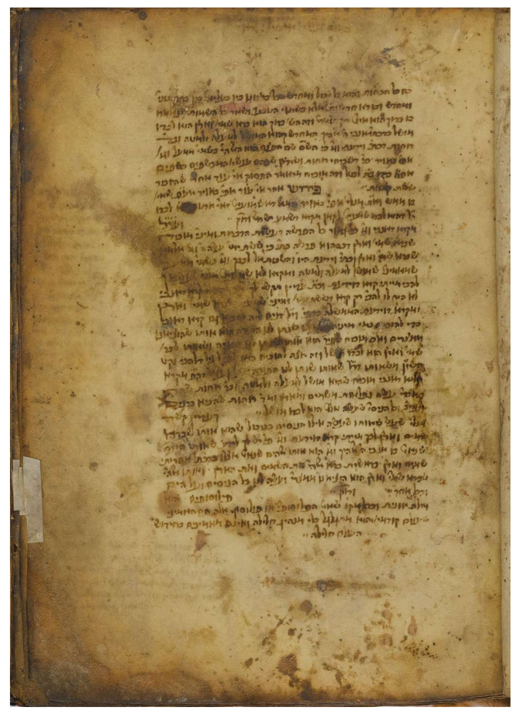
This experiment used a modern hebrew handwritten document for content, with a background created by averaging a pixels color from the style document.

2.12.2. Style Input

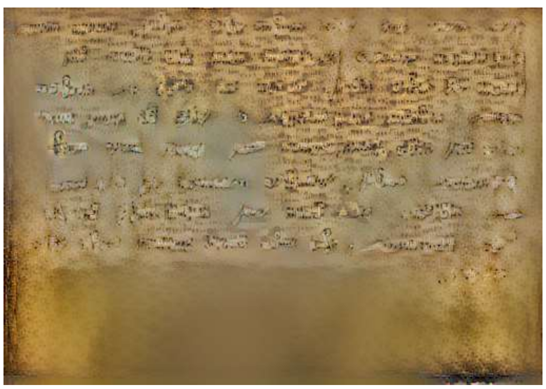
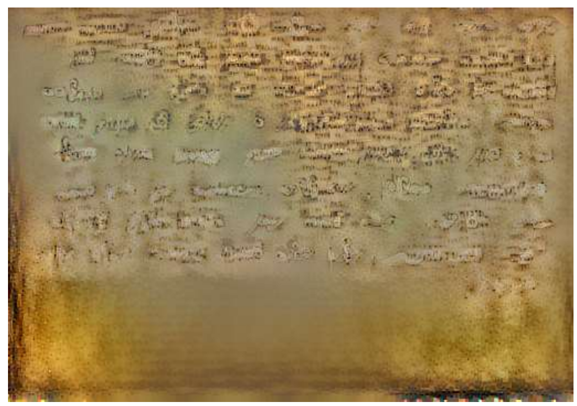
This experiment used a hebrew middle age document with text for style.

2.12.3. Results

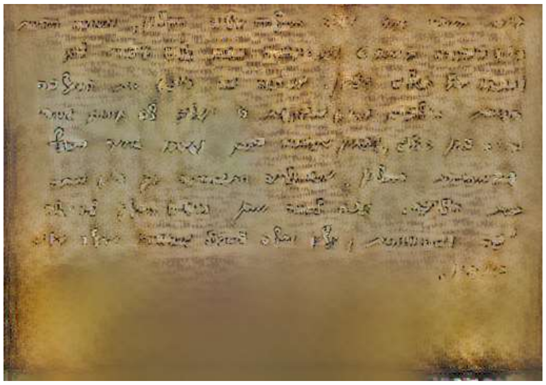
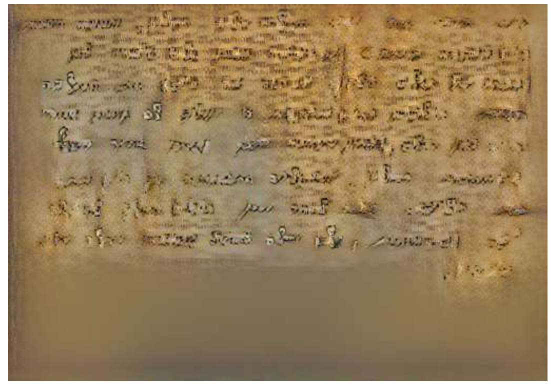
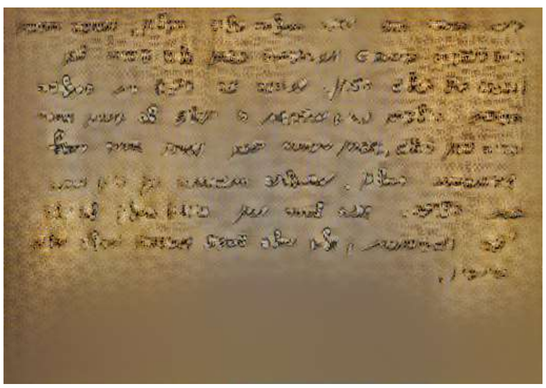
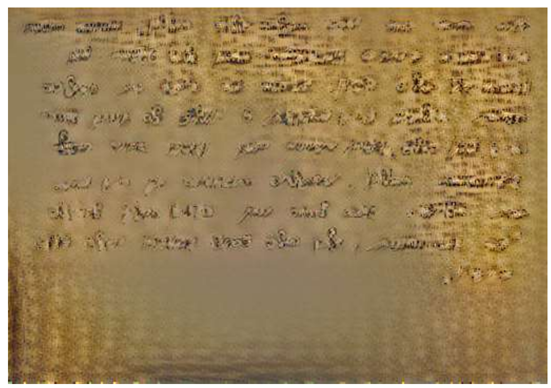
We will show samples of the results with each weight and content layer (style layers are the same).

Content image	Style image
	

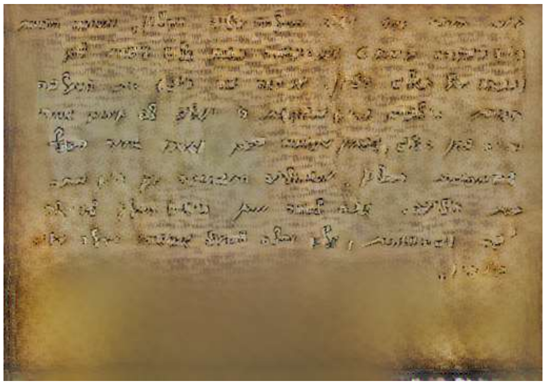
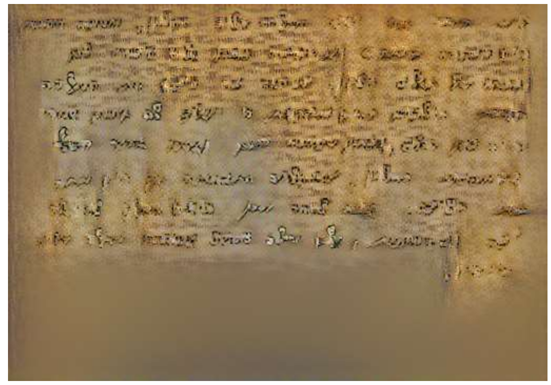
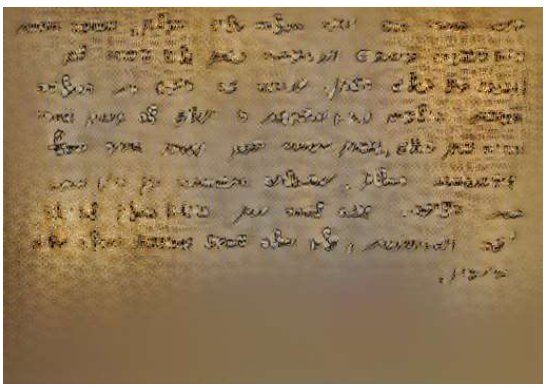
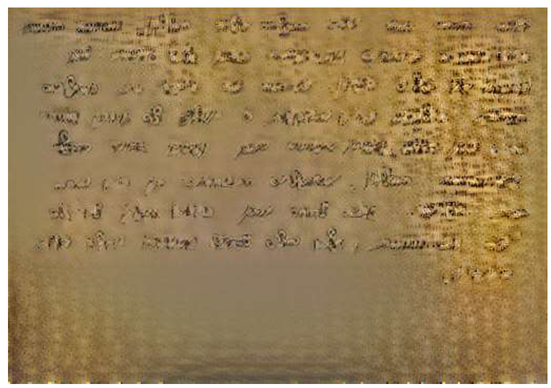
2.12.3.1. Weight 0

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

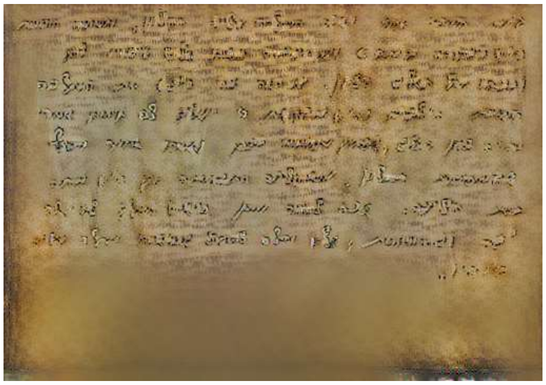
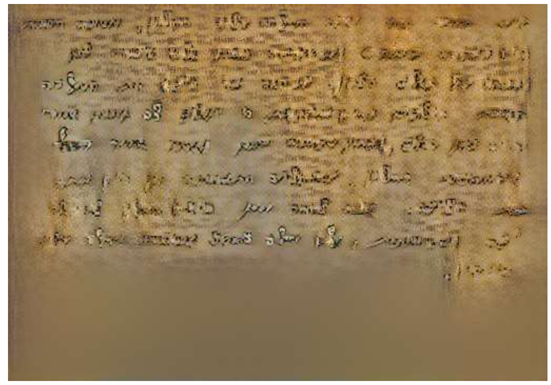
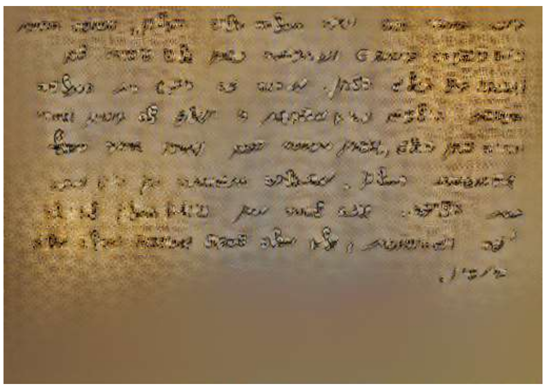
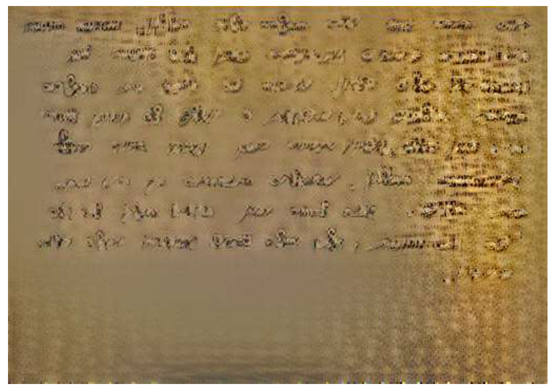
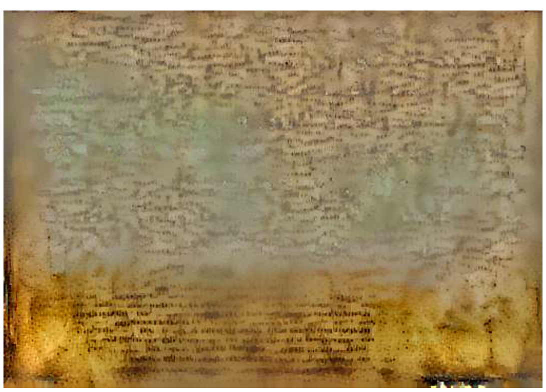
2.12.3.2. Weight 1

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

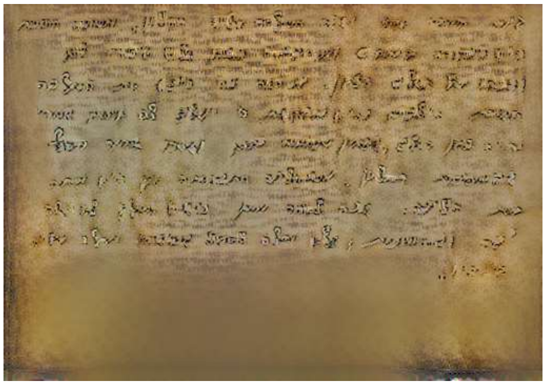
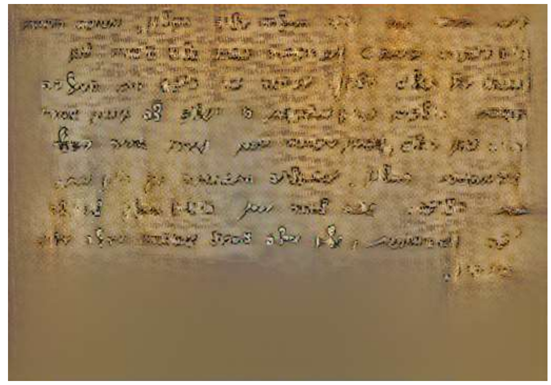
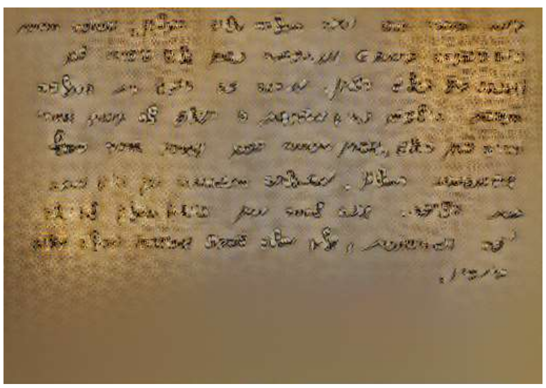
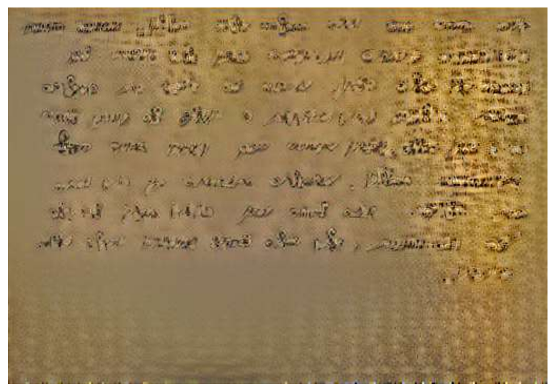
2.12.3.3. Weight 2

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

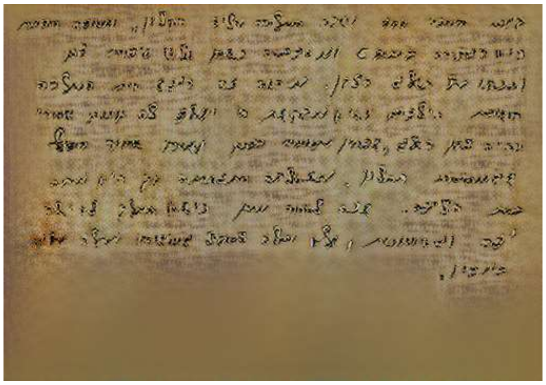
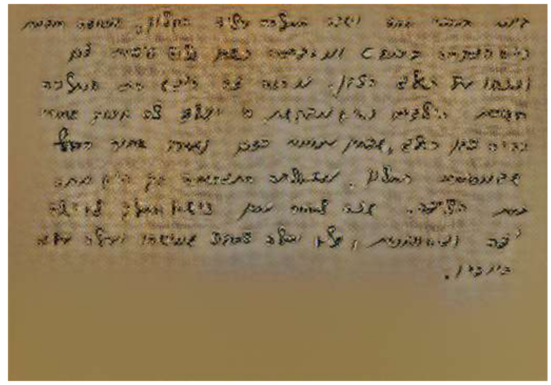
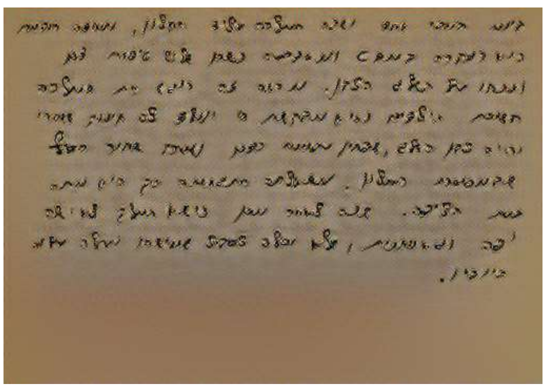
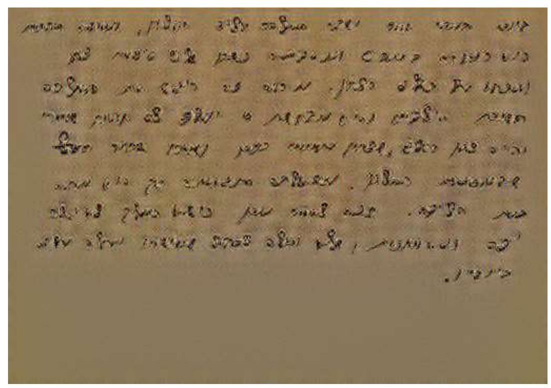
2.12.3.4. Weight 3

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

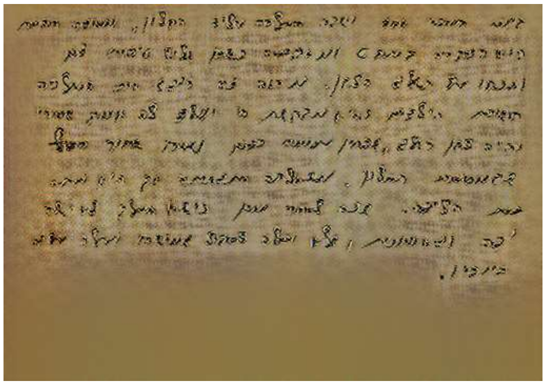
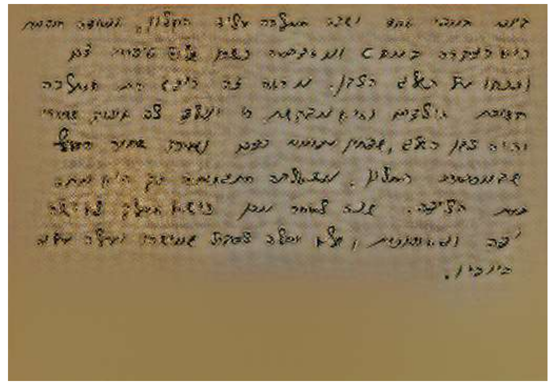
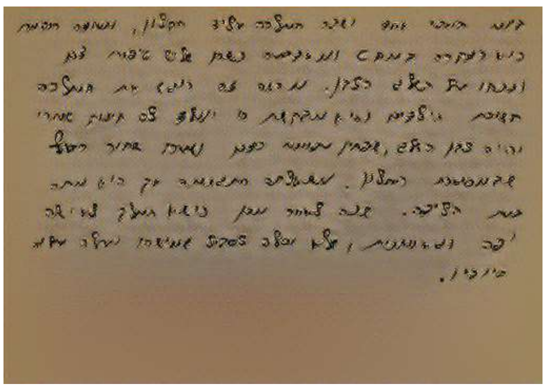
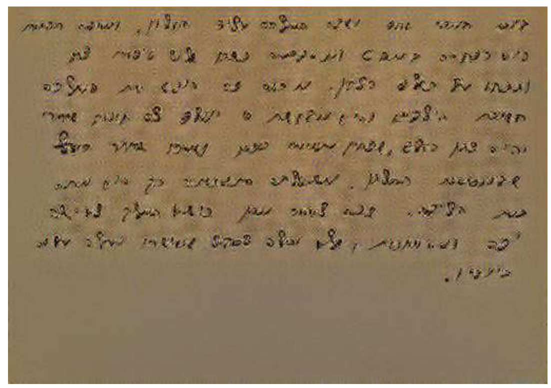
2.12.3.5. Weight 4

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

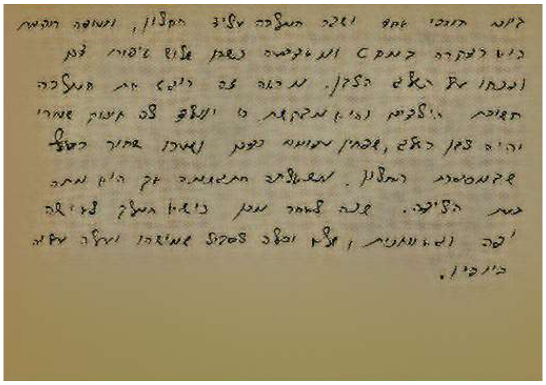
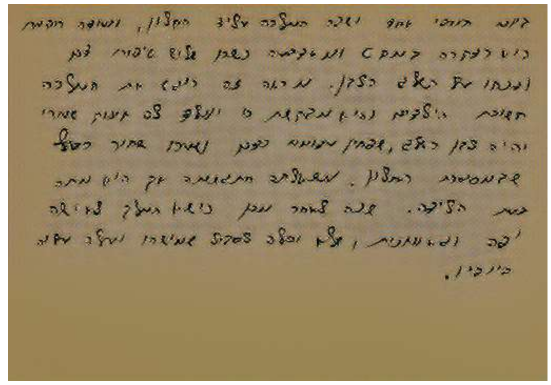
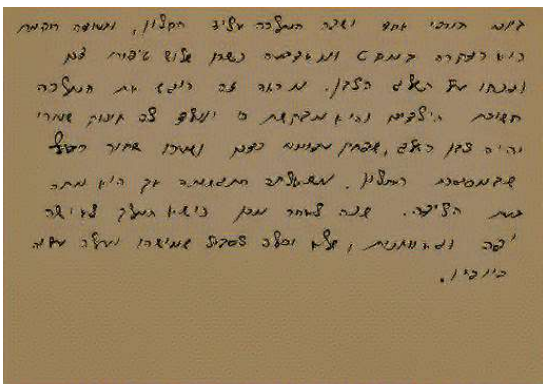
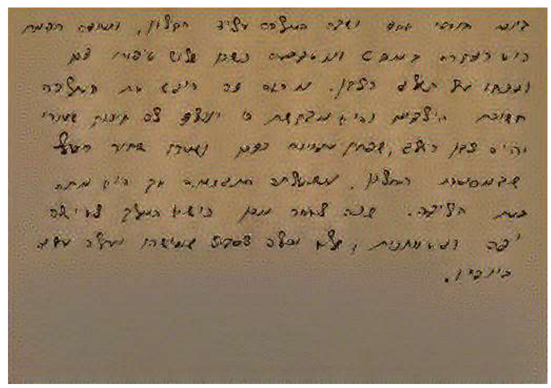
2.12.3.6. Weight 5

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

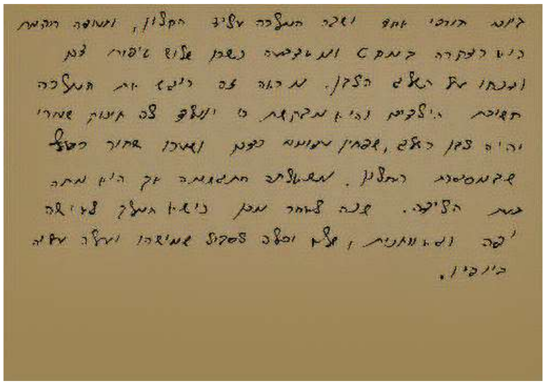
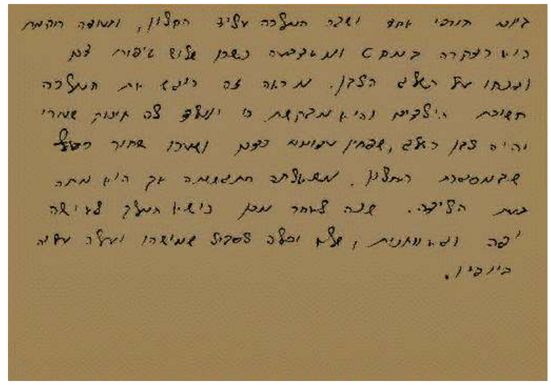
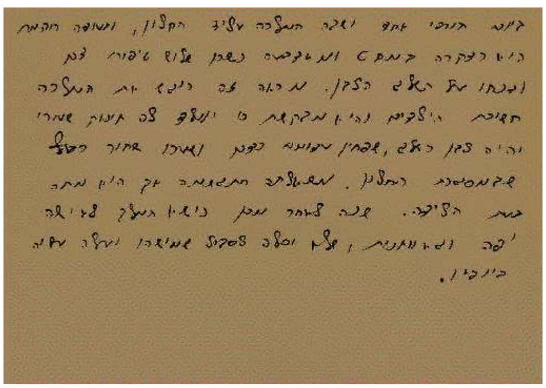
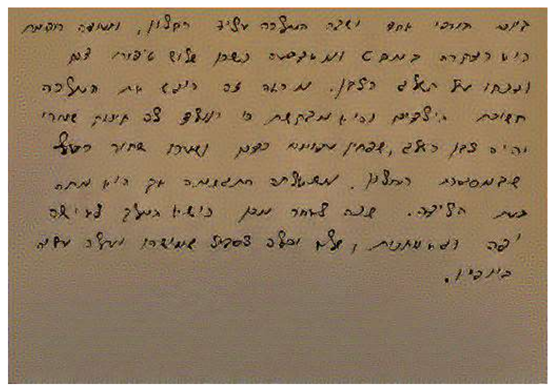
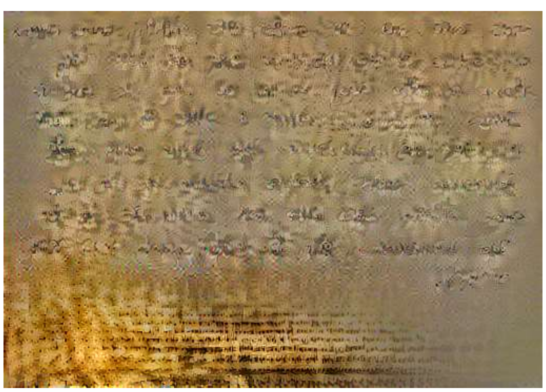
2.12.3.7. Weight 6

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.12.3.8. Weight 7

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.12.3.9. Weight 8

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.12.4. Discussion

As we can see, in this experiment there are a lot of blurry areas in the results. Also, in the last weights there are results where content is more clear but style isn't being passed well. So, results have not been improved.

2.13. Experiment 10

Now, we test if adding to the content document a background created by averaging a pixels color from the style document will give us a better result.

2.13.1. Content Input

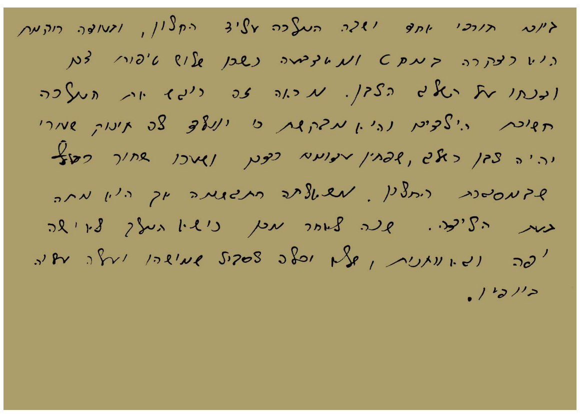
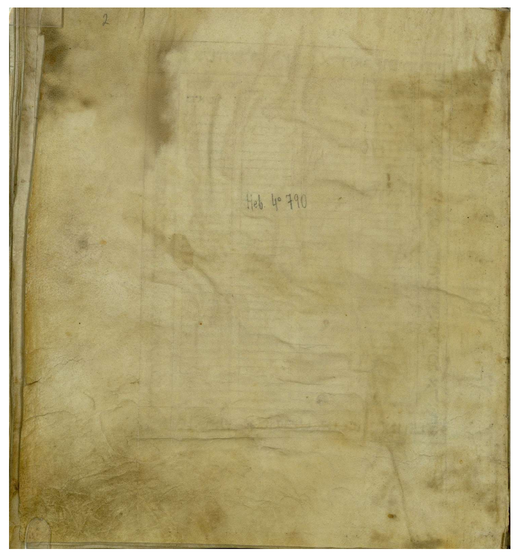
This experiment used a modern hebrew handwritten document for content, with a background created by averaging a pixels color from the style document.

2.13.2. Style Input

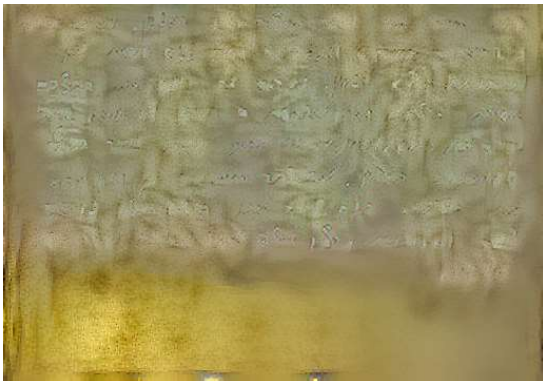
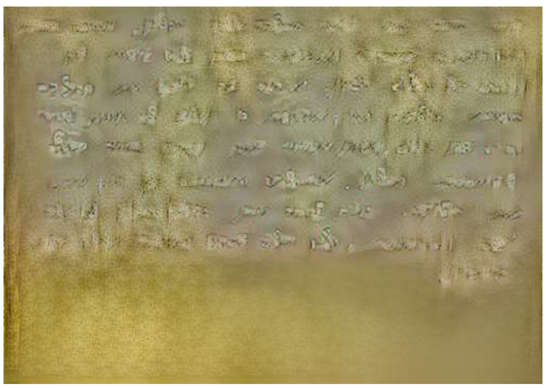
This experiment used a hebrew middle age document without text for style.

2.13.3. Results

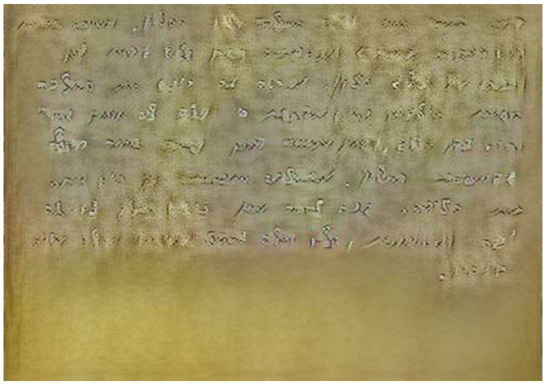
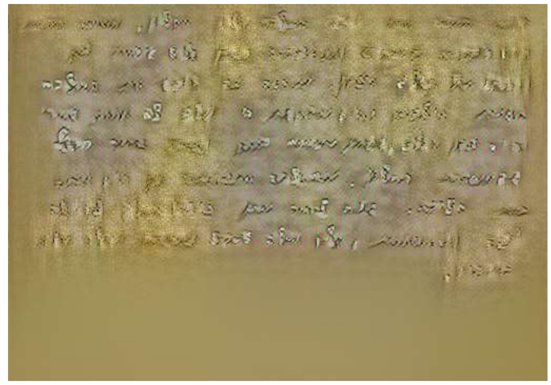
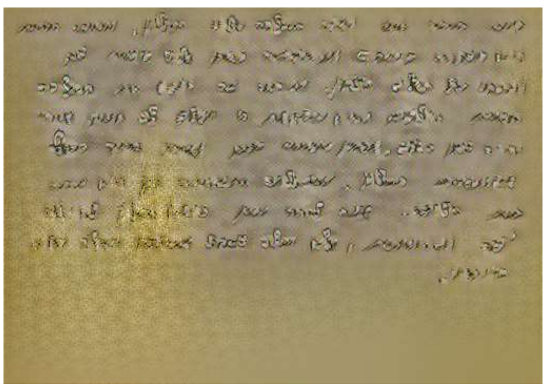
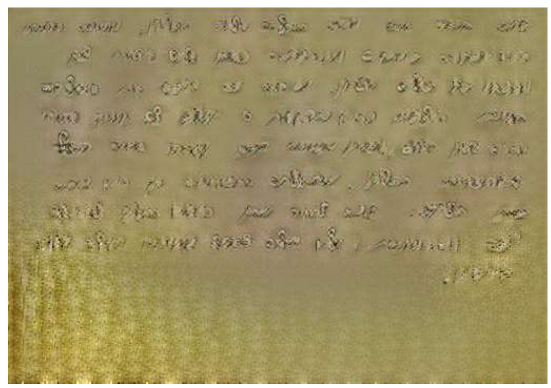
We will show samples of the results with each weight and content layer (style layers are the same).

Content image	Style image
	

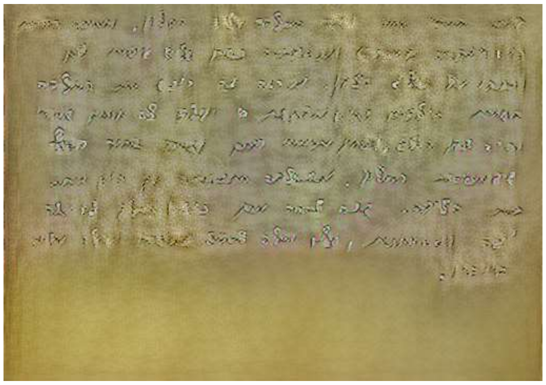
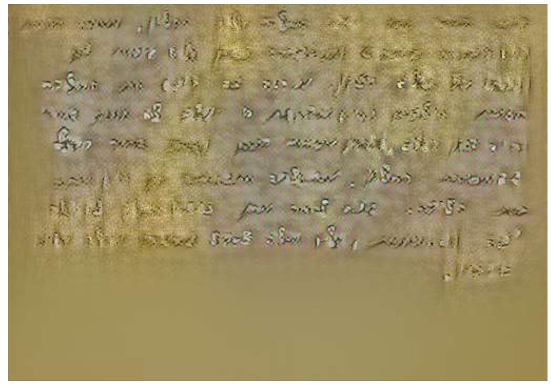
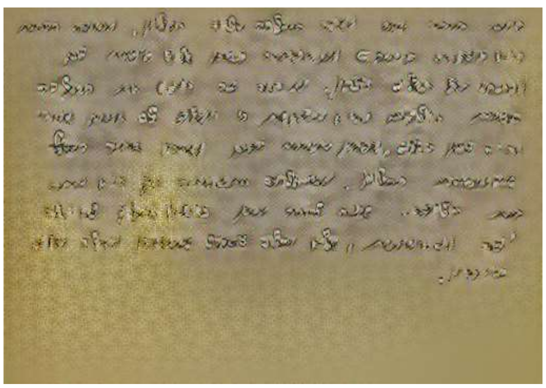
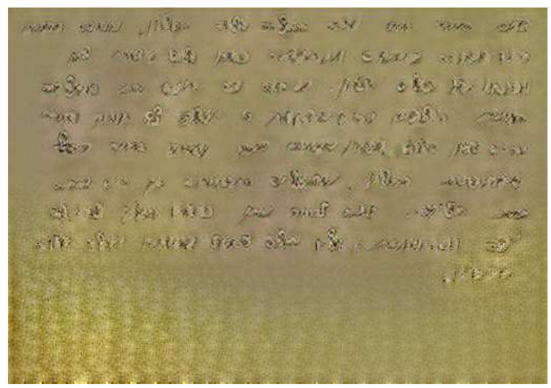
2.13.3.1. Weight 0

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

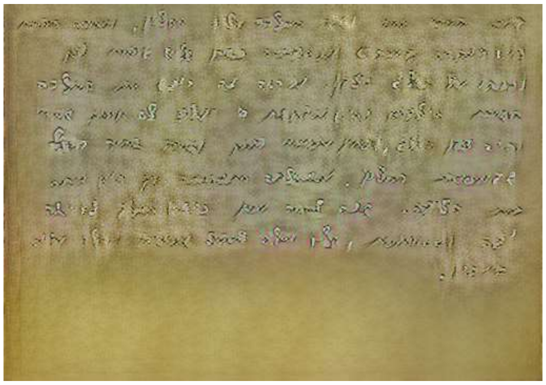
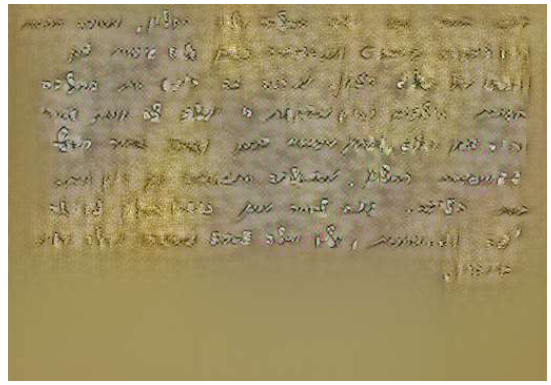
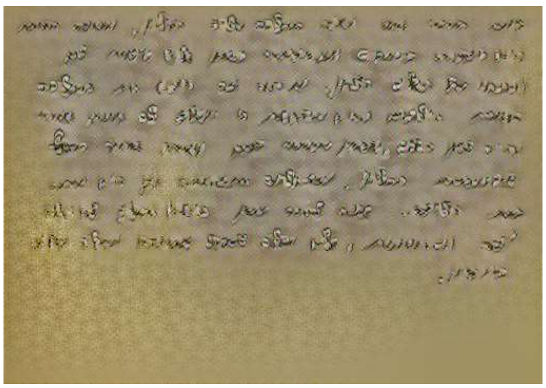
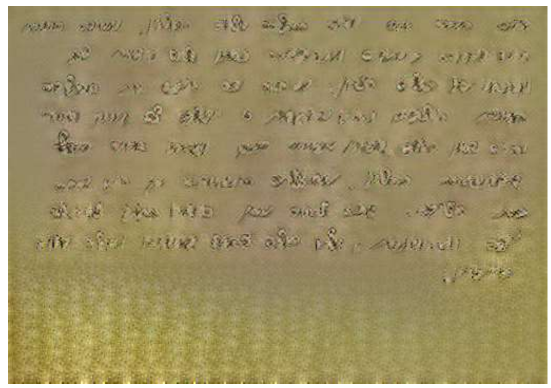
2.13.3.2. Weight 1

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

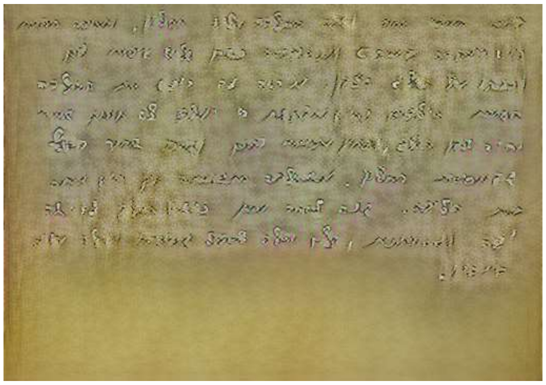
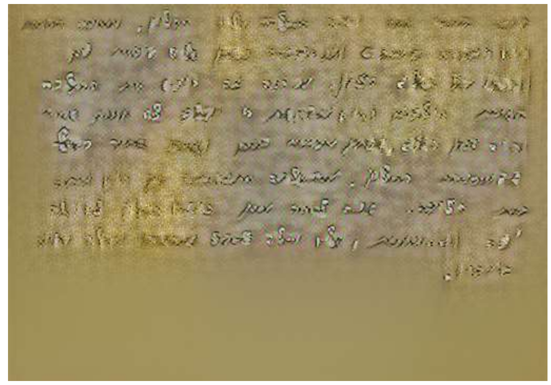
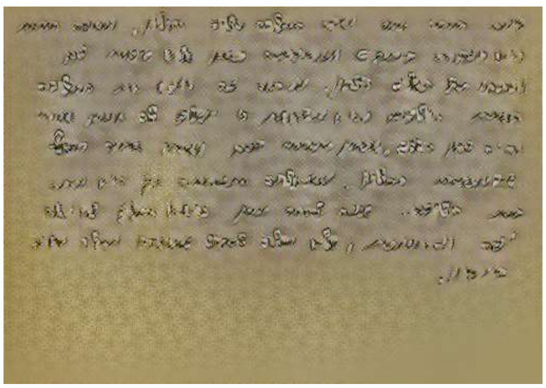
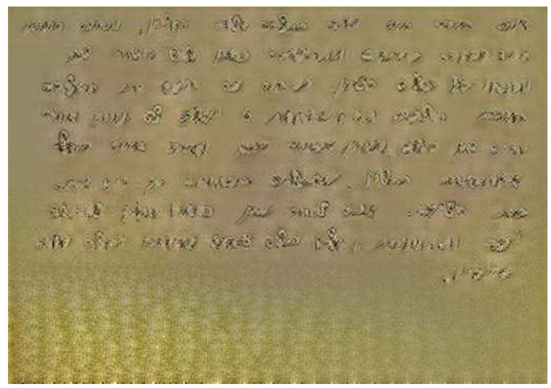
2.13.3.3. Weight 2

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

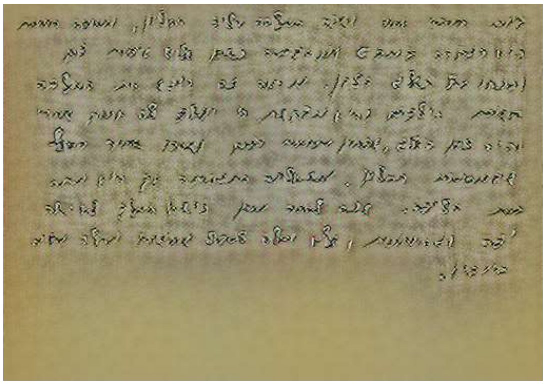
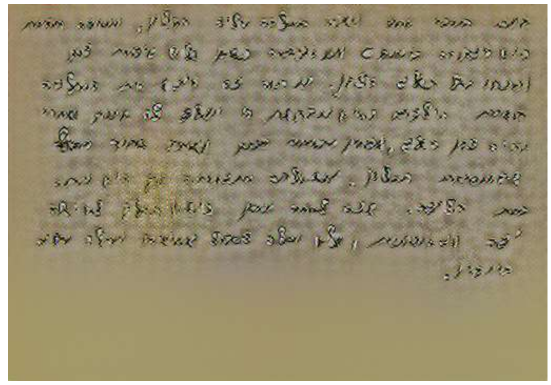
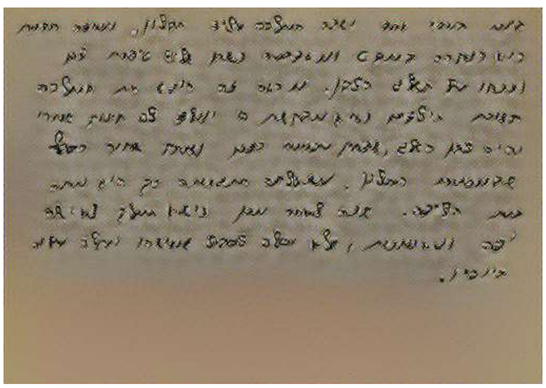
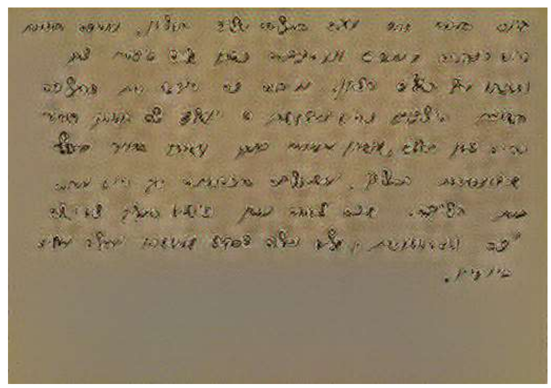
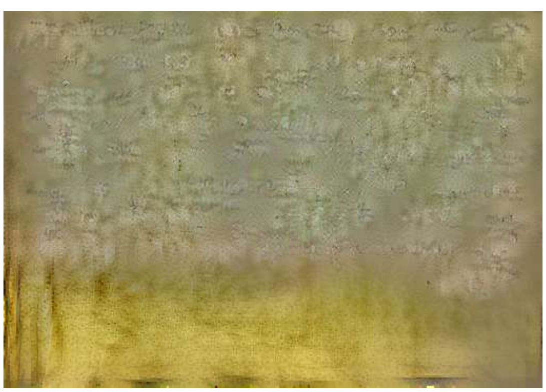
2.13.3.4. Weight 3

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

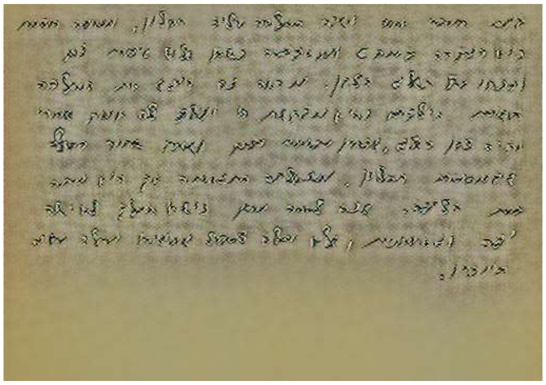
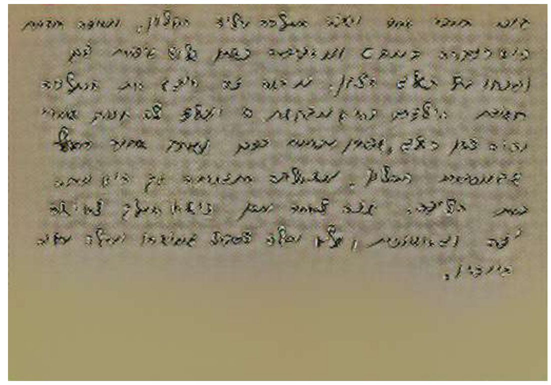
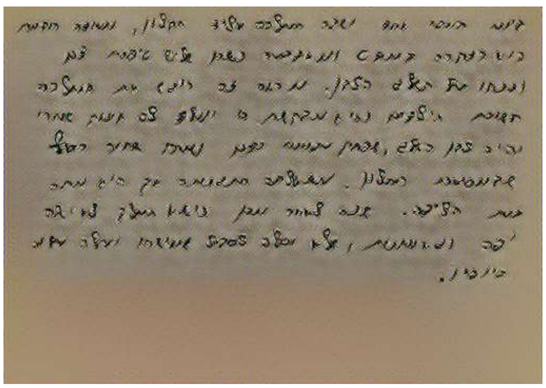
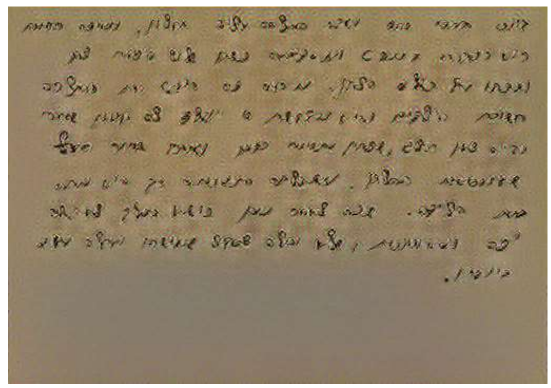
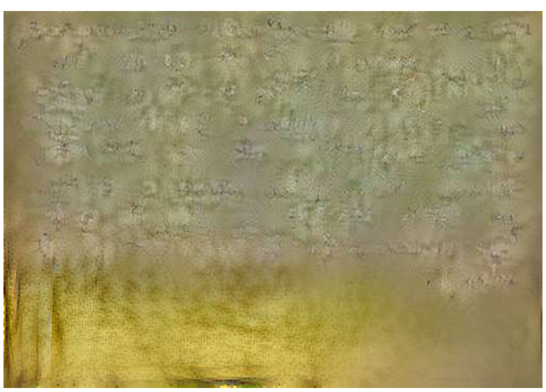
2.13.3.5. Weight 4

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

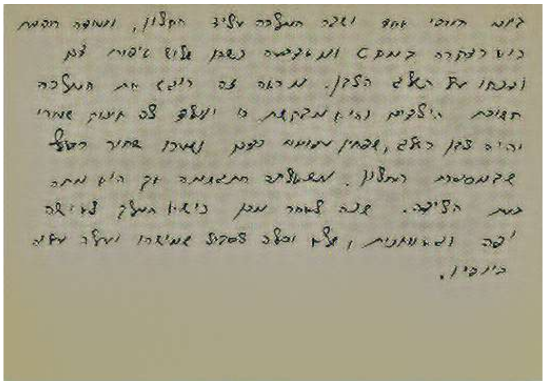
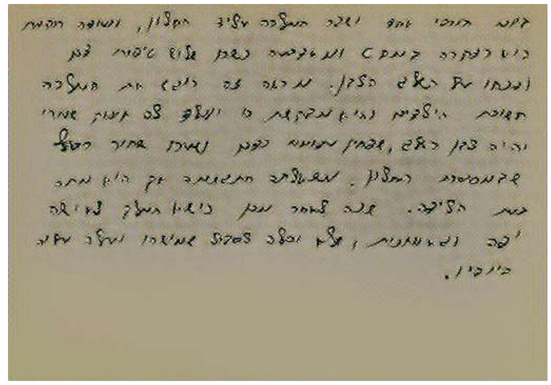
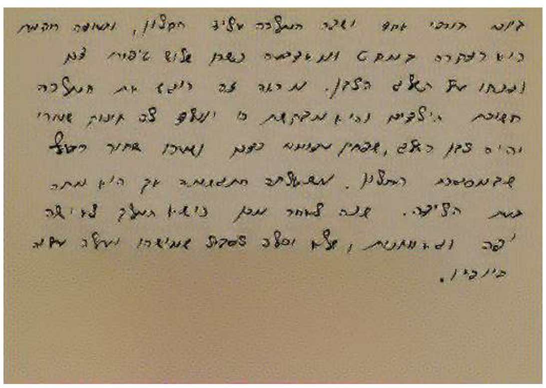
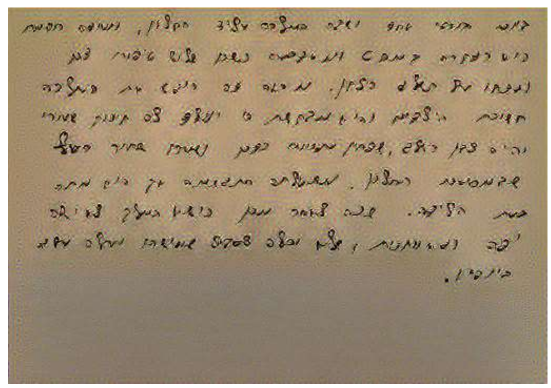
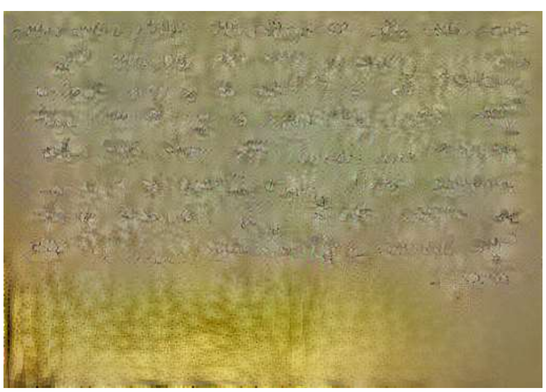
2.13.3.6. Weight 5

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

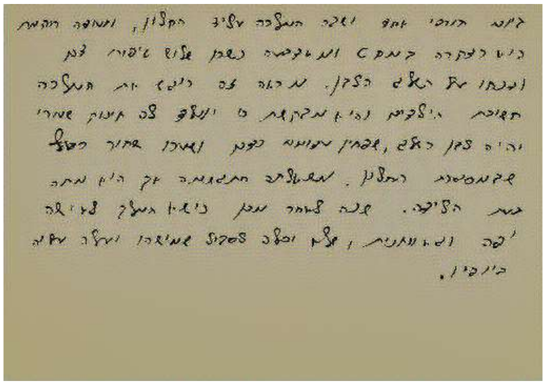
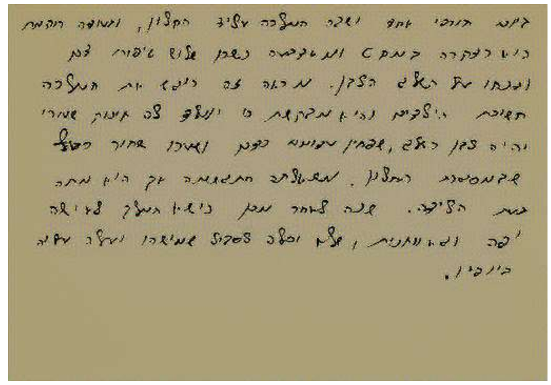
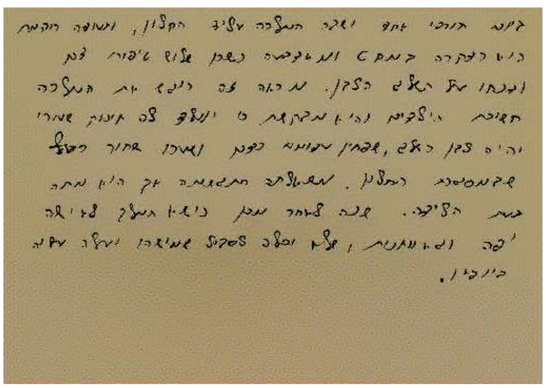
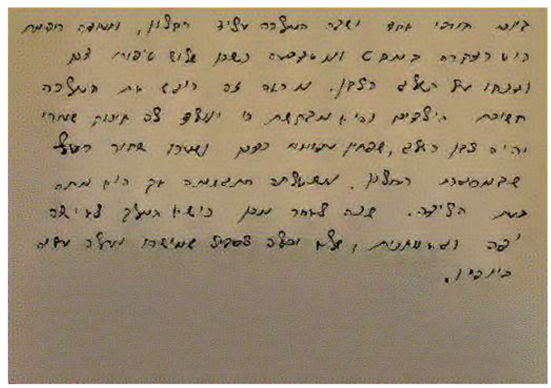
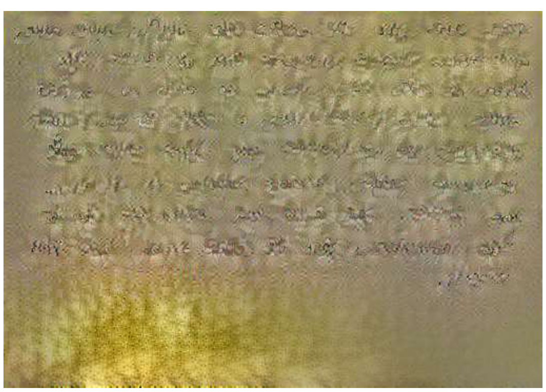
2.13.3.7. Weight 6

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.13.3.8. Weight 7

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.13.3.9. Weight 8

block3_conv2	block3_conv3
	
block3_conv4	block4_conv2
	
block5_conv2	
	

2.13.4. Discussion

As we can see, in this experiment there are a lot of blurry areas in the results. Also, in the last weights there are results where content is more clear but style isn't being passed well. So, results have not been improved.

Appendix B - Project Data

English Data

In our project there is one type of english data.

1. **Modern English**

   Images are documents and sentences, image format is png.

   Preprocess needed:

   1.1. Cropping

   Documents images need to be cropped to extract only handwriting text.

Hebrew Data

In our project there are three types of hebrew data.

1. **Modern Hebrew**

   Images are documents in .tif format.

   Preprocess needed:

   1.1. Changing Type

   Images need to be converted to .png format.

   1.2. Rotating

   Images need to be rotated 180deg to align text.

   1.3. Cropping

   Document images need to be cropped to extract only handwriting text.

   1.4. Trehoulding

   We need to remove yellow lines from the images.

2. **Cairo Geniza Historical Hebrew**

   Images are documents in .tif and .png format.

   No preprocess needed. This will be a hands on data preprocess by our needs.

3. **Middle Ages Historical Hebrew**

   Images are documents in .tif and .png format.

   No preprocess needed. This will be a hands on data preprocess by our needs.

Arabic Data

In our project there are two types of arabic data.

1. **Modern Arab**

   Images are documents and sentences in .tif format.

   Preprocess needed:

   1.1. Changing Type

   Images need to be converted to .png format.

2. **Historical Arab**

   Images are documents in .jpg format.

   No preprocess needed. This will be a hands on data preprocess by our needs.

Latin Data

In our project there is one type of latin data.

3. **Historical Latin**

   Images are documents in .jpg format.
   No preprocess needed. This will be a hands on data preprocess by our needs.