Neural Image Style Transfer

Neuronaler Stiltransfer für Bilder mit vortrainiertem Modell aus dem Magenta-Projekt Ein Report

AIAp: AI Applications

Nico Fehr, Kyra Maag

1
Einführung

1.1. Abstract

Problem	Generative künstliche Intelligenzen wie DALL-E, Midjourney, OpenAIs MuseNet oder Pictory, die Bilder, Audio und sogar Video erstellen, erweitern und verändern können, sind auf dem Vormarsch. Dabei stellt sich die Frage, ob es auch möglich ist, einen Stil, z.B. den eines Bildes, auf andere Bilder zu übertragen. In diesem Report stellen wir eine Implementierung in Form einer Webanwendung namens „Pablo“ eines Neural Style Transfer Modells aus der Magenta Library vor.
Ziel	Wir wollen zeigen, wie die Magenta Library genutzt werden kann und welche Möglichkeiten sich daraus für Kulturschaffende ergeben und wie generative AI dabei eingesetzt werden kann.
Methode	Es wurden verschiedene Fachtexte, Artikel und Websites recherchiert und eine Anwendung in Form einer Python-basierten Django-Webanwendung implementiert.
Wesentliche Ergebnisse	Unsere Quellen und die eigens entwickelte Webanwendung „Pablo“ bestätigen, dass die Technologie die Arbeit mit Bildern, insbesondere in der digitalen Fotografie, durchaus erleichtern und die Kreativität weiter fördern kann.
Empfehlung	In diesem Bericht kommen wir zu dem Schluss, dass aus künstlerischer Sicht eine interessante Entwicklung im Bereich der generativen KI stattfindet. Eine kommerzielle Anwendung wäre auch für Fotografen denkbar. Anwendungsprogramme wie Adobe Lightroom, die hauptsächlich mit digitalen Filtern arbeiten, könnten ein solches Modell nutzen, um komplexe Stile nahezu in Echtzeit auf andere Bilder anzuwenden.

1.2. Motivation

Wir haben uns für dieses Tool entschieden, weil wir uns für Kunst interessieren. Außerdem haben wir bereits mit TensorFlow gearbeitet und wissen, dass es einfach ist mit der Bibliothek und den Modellen zu arbeiten. Dazu tragen vor allem die gute Dokumentation und die intuitive Handhabung von TensorFlow bei. Da uns das Thema mit dem Style Transfer und der Generierung von Bildern, durch eine künstliche Intelligenz, fasziniert hat und momentan in den Medien sehr präsent ist, sehen wir auch eine gewisse Chance, dass diese Technologie in Zukunft noch weiter ausgebaut wird.

1.3. Hintergrund

TensorFlow und die Magenta-Bibliothek, die wir für unser Modell verwenden, werden von Google AI bereitgestellt und vom Google Brain Team entwickelt. [1] Magenta wurde als Open-Source-Projekt auf Github gestartet und stellt eine Sammlung von Tools und Modellen für Künstler, Musiker und Wissenschaftler zur Verfügung, um das Potenzial von KI in der Kreativbranche auszuschöpfen und zu erforschen. [2] Für unseren Bericht haben wir ein Modell aus der Magenta-Bibliothek ausgewählt: „Fast Transfer for Aribtrary Styles“ [3]. Mit diesem Modell ist es dem Google Brain Team gelungen, ein bestehendes Modell [4], das nur auf einen oder eine begrenzte Anzahl von Stilen anwendbar war, zu optimieren und eine Methode bereitzustellen, die auf beliebige Malstile angewendet werden kann. Diese Methode wird in der Arbeit „Exploring the structure of a real-time, arbitrary neural artistic stylization network“ [5] des Google Brain Teams beschrieben.

1.4. Community

TensorFlow als Basis der Magenta Library ist sicher kein Nischenprodukt mehr. Auf Reddit hat TensorFlow eine Community von gut 30’000 Mitgliedern und gehört damit zu den Top 5% der Website, gemessen an der Anzahl Mitglieder [6]. Auch Magenta strebt eine wachsende Community an und fordert seine Nutzer auf, sich einzubringen und auszutauschen. Dazu wird auch auf Twitter mit dem Hashtag „#MadeWithMagenta“ geworben. Künstlerinnen und Künstler sollen zeigen, was sie mit Magenta geschaffen haben. [7] Google organisierte ausserdem einige Talks [8] mit Magenta, in denen die Möglichkeiten der generativen Modelle und Tools, die Magenta bietet, diskutiert und analysiert wurden. Dabei wurden vor allem Tools vorgestellt, die in erster Linie Künstler bei der Musikkomposition unterstützen können.

2
Anwendung der Library

2.1. Funktionsweise

Das Modell [9] des Google Brain Teams funktioniert wie folgt:

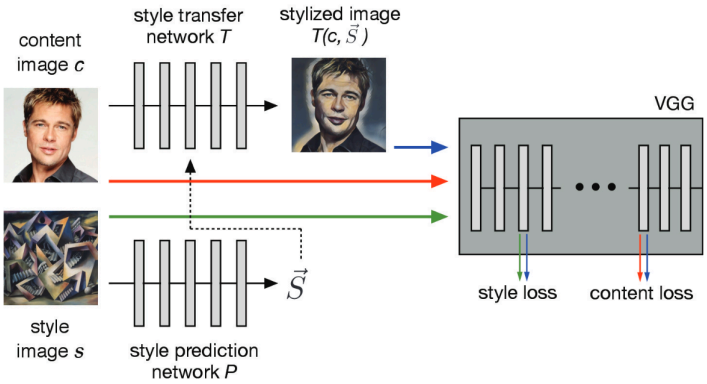

Abbildung 2.1: Diagram der Modell-Architektur, Quelle: Ghiasi et al. [5]

Im Paper von Ghiasi et al. [5] wird das Vorgehen aus Abbildung 2.1 folgendermassen beschrieben:

1. Das „Style prediction network P“ sagt einen Vektor $\vec{S}$ aus dem „Style image s“ vorher.
2. Der Vektor $\vec{S}$ besteht aus einem Set an Normalisierungskonstanten für das „Style transfer network T“
3. Das „Style transfer network T“ transformiert das Foto „Content image c“ in eine stilisierte Darstellung „Stylized image $T(c, \vec{S})$“
4. Die Inhalts- und Stilverluste (Losses) werden aus der Distanz im Repräsentationsraum des VGG-Bildklassifikationsnetzes [10] abgeleitet.

Das „Style transfer network T“ und das „Style transfer network P“ folgen weitgehend der Inception-v3 [11] [12] Architektur. Die genauen mathematischen Details sind im Paper von Ghiasi et al. [5] beschrieben.

2.2. Setup

Das Setup des Tools stellt keine grosse Herausforderung dar. Man benötigt für ein Tutorial [13] [14] lediglich eine Python-Umgebung mit den notwendigen Tools wie Numpy, Matplotlib, TensorFlow und TensorFlow Hub sowie ein Jupyter Notebook. Das Modell wird dann mittels TensorFlow Hub [9] auf die Umgebung heruntergeladen und kann direkt auf dem Notebook verwendet werden. Das Setup ist in der TensorFlow Dokumentation unter „Advanced“ beschrieben. [13] Eine genauere Beschreibung der Anwendung findet sich im Anhang unter Appendix A.

2.3. Umsetzung

Um zu zeigen, wie dieses Modell verwendet werden kann, haben wir eine Webanwendung namens „Pablo“ implementiert, die nach dem spanischen Künstler Pablo Picasso benannt ist. Mit Hilfe des auf Python basierenden Frameworks Django haben wir einen Webserver erstellt, auf dem die Benutzer ihre eigenen Bilder hochladen können. Auf der Seite kann dann aus einer Liste der 50 einflussreichsten Künstlerinnen und Künstlern ein Bild ausgewählt werden. Den Datensatz für die Kunstwerke haben wir von Kaggle aus dem Datensatz „Best Artworks of all Time“ exportiert. [15] Die Bildauswahl wird dann als Style auf das Bild des Benutzers angewendet. Das Modell zeigt eine erstaunliche Fähigkeit, Stile zu übernehmen. Es kann mit Bildern arbeiten, auf denen das Modell noch nie trainiert wurde. Trotz dieser Fähigkeit wollen wir die Auswahl der verfügbaren Kunstwerke, deren Stile übernommen werden können, einschränken. Bei der weitergehenden Nutzung des Modells ist man sehr eingeschränkt. Weitere Einstellungen am Modell sind

kaum möglich. Die Übernahme des Stils durch das Modell lässt keine weiteren Parameter zu. Es war relativ schwierig herauszufinden, wie stark der Stil auf das eigene Bild übertragen werden kann. In der Dokumentation der Bibliothek wird kein solcher Parameter erwähnt. Durch entsprechendes Skalieren der Bilder hat man jedoch die Möglichkeit, einen Stil stärker oder schwächer auf ein Bild zu übertragen. Umgesetzt wurde dies, wie im folgenden Codeausschnitt aus `utils.py` an der Funktion `load_img` zu sehen ist, mit dem Parameter `max_dim`.

```
...
def load_img(path_to_img, max_dim=512):
    img = tf.io.read_file(path_to_img)
    img = tf.image.decode_image(img, channels=3)
    img = tf.image.convert_image_dtype(img, tf.float32)

    shape = tf.cast(tf.shape(img)[:-1], tf.float32)
    long_dim = max(shape)
    scale = max_dim / long_dim

    new_shape = tf.cast(shape * scale, tf.int32)

    img = tf.image.resize(img, new_shape)
    img = img[tf.newaxis, :]
    return img
```

Die oben beschriebene Funktion wird verwendet, um aus den Bildern entsprechende „Tensoren“ [16] zu erstellen, die als Parameter für die Anwendung des Stils mit dem Magenta-Modell verwendet werden. Die Methode `hub_model` ist im folgenden Codeausschnitt aus `transfer.py` zu sehen:

```
import os
import tensorflow as tf
import tensorflow_hub as hub

from pablo_app.style_transfer.utils import load_img, tensor_to_image

os.environ["TFHUB_MODEL_LOAD_FORMAT"] = "COMPRESSED"
os.environ["TFHUB_DOWNLOAD_PROGRESS"] = "True"

hub_model_url = "https://tfhub.dev/google/magenta/arbitrary-image-stylization-v1-256/2"
hub_model = hub.load(hub_model_url)


def transfer_style(content_path, style_path):
    content_image = load_img(content_path, 1024)
    style_image = load_img(style_path, 290)

    stylized_image = hub_model(tf.constant(content_image),
                               tf.constant(style_image))[0]
    return tensor_to_image(stylized_image)
```

Entscheidend sind hier die beiden `max_dim`-Parameter `1024` für das Inhaltsbild und `290` für das Stilbild. Wir haben diese beiden Parameter heuristisch bestimmt. Die Übertragung des Stils auf das Inhaltsbild erschien mit diesen Parametern am natürlichsten.

2.4. Architektur

Die Architektur unserer Implementation des Modells als Webanwendung „Pablo“ sieht wie folgt aus:

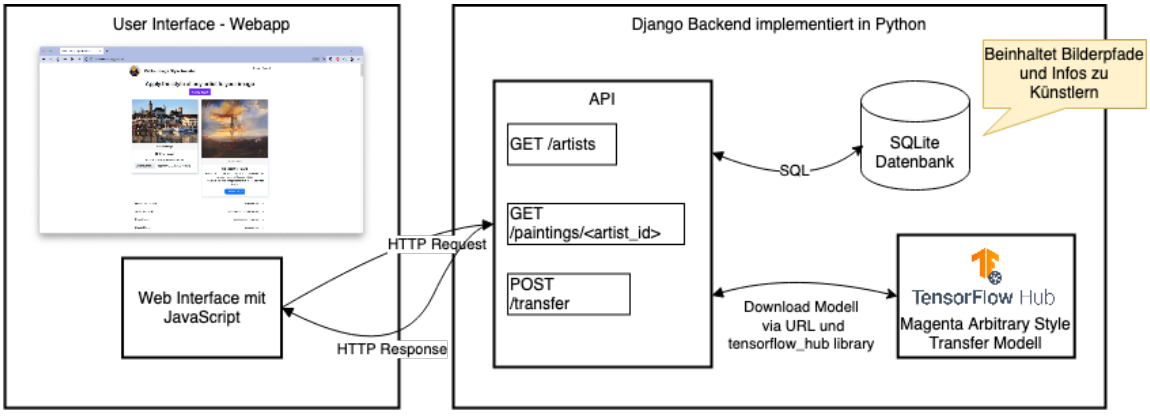

Abbildung 2.2: Diagram der Architektur unserer Implementation von „Pablo“, Quelle: Eigene Darstellung

3
Zusammenfassung

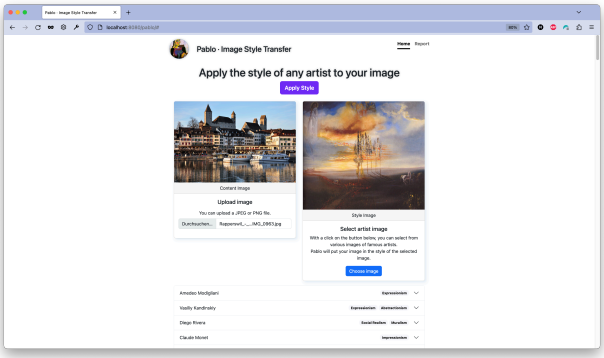

Abbildung 3.1: Screenshot der Webanwendung mit dem Inhaltsbild und Stilbild, Quelle: Eigene Darstellung

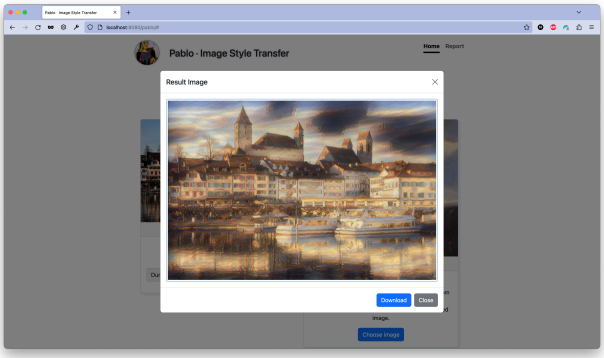

Abbildung 3.2: Screenshot des Resultats der Stilanwendung auf das Inhaltsbild, Quelle: Eigene Darstellung

Die effiziente Funktionsweise des „Style transfer network“ des Modells, das nahezu in Echtzeit (wenige Sekunden) arbeitet, macht es zu einer idealen Methode für kommerzielle und benutzerfreundliche Bildverarbeitungssoftware. Dies unterscheidet das Modell des Google Brain Teams von anderen bereits beschriebenen Methoden. [5]

Wir sehen mehrere Anwendungsbereiche für das Modell. Als Produktionshilfe für Künstler können „Neural Style Transfer“-Modelle die Arbeitsschritte effizienter gestalten. Künstler können unabhängig vom Medium ihren eigenen Stil in andere Werke einfließen lassen. Es ist vorstellbar, dass ein Modell nicht nur Stile von Bildern effizient erkennen und auf andere übertragen kann, sondern dasselbe auch mit Musik, Film oder Text machen kann.

Wir kommen zu dem Schluss, dass aus künstlerischer Sicht eine interessante Entwicklung im Bereich der generativen AI stattfindet. Anwendungen wie Adobe Lightroom, die hauptsächlich mit digitalen Filtern arbeiten, könnten ein solches Modell nutzen, um komplexe Stile nahezu in Echtzeit auf andere Bilder anzuwenden. Auch gängige Social-Media-Apps wie Snapchat, die spielerische Möglichkeiten der Bildbearbeitung bieten, könnten von solchen effizienten Modellen profitieren. Eine kommerzielle Anwendung ist daher aus unserer Sicht für Fotografen und im Alltag durchaus möglich. Zusammenfassend sehen wir einen starken kommerziellen Nutzen in einem solchen Modell. Es kann auch für eine Bachelorarbeit verwendet werden. Im Bereich der Stilerkennung und Bildklassifikation mit Hilfe von AI müsste noch weiter geforscht werden, um ein benutzerfreundliches Werkzeug zur Lösung entsprechend komplexer Probleme zu entwickeln. Es wäre interessant, weitere Methoden zu erforschen, um Stile noch effizienter zu erkennen und für die Videoverarbeitung zu nutzen.

A
Setup

In diesem Anhang beschreiben wir kurz unsere Konfiguration der Webanwendung „Pablo“ und zeigen einige Screenshots. Wir haben Python für die Entwicklung der Anwendung und JavaScript für das Frontend der Single-Page-Webanwendung verwendet. PyCharm wurde als IDE für das Backend mit Django eingesetzt.

Utils für unsere Anwendung `utils.py`

```python
import tensorflow as tf
import PIL.Image
import numpy as np


def tensor_to_image(tensor):
    tensor = tensor * 255
    tensor = np.array(tensor, dtype=np.uint8)
    if np.ndim(tensor) > 3:
        assert tensor.shape[0] == 1
        tensor = tensor[0]
    return PIL.Image.fromarray(tensor)


def original_image_path(style_image_path: str) -> str:
    if style_image_path is not None:
        split_image_path = style_image_path.split("/")
        image_name = "_".join(split_image_path[-1].split("_")[:-1])
        return "/".join(split_image_path[:3] + ["original"] + [image_name] + split_image_path[4:])
    return ""


def load_img(path_to_img, max_dim=512):
    img = tf.io.read_file(path_to_img)
    img = tf.image.decode_image(img, channels=3)
    img = tf.image.convert_image_dtype(img, tf.float32)

    shape = tf.cast(tf.shape(img)[:-1], tf.float32)
    long_dim = max(shape)
    scale = max_dim / long_dim

    new_shape = tf.cast(shape * scale, tf.int32)

    img = tf.image.resize(img, new_shape)
    img = img[tf.newaxis, :]
    return img
```

`transfer.py` führt den Stiltransfer durch. In dieser Datei wird deutlich, wie einfach die Anwendung mit dem TensorFlow Hub funktioniert. Das Hub-Modell wird als URL gespeichert und kann dann einfach mit dem `tensorflow_hub` heruntergeladen und verwendet werden.

```python
import os
import tensorflow as tf
import tensorflow_hub as hub

from pablo_app.style_transfer.utils import load_img, tensor_to_image

os.environ["TFHUB_MODEL_LOAD_FORMAT"] = "COMPRESSED"
os.environ["TFHUB_DOWNLOAD_PROGRESS"] = "True"

hub_model_url = "https://tfhub.dev/google/magenta/arbitrary-image-stylization-v1-256/2"
hub_model = hub.load(hub_model_url)


def transfer_style(content_path, style_path):
    content_image = load_img(content_path, 1024)
    style_image = load_img(style_path, 290)

    stylized_image = hub_model(tf.constant(content_image),
                               tf.constant(style_image))[0]
    return tensor_to_image(stylized_image)
```

A.1. Screenshots

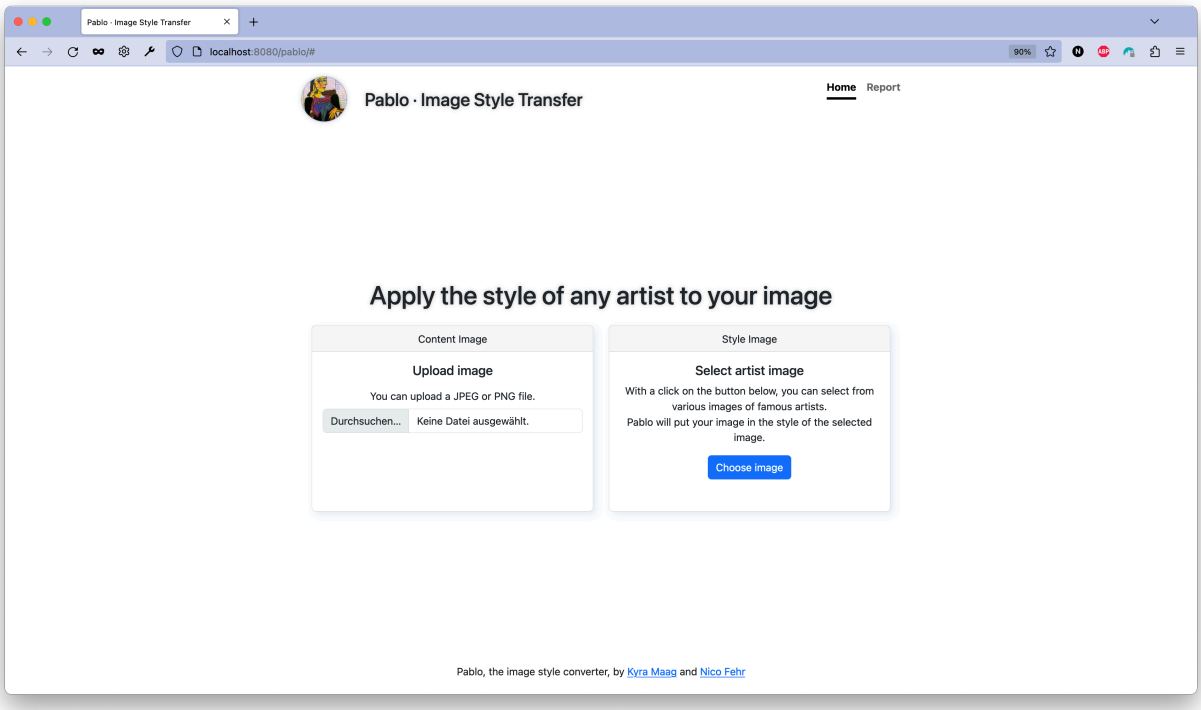

Abbildung A.1: Webanwendung „Pablo“ im Normalzustand, Quelle: Eigene Darstellung

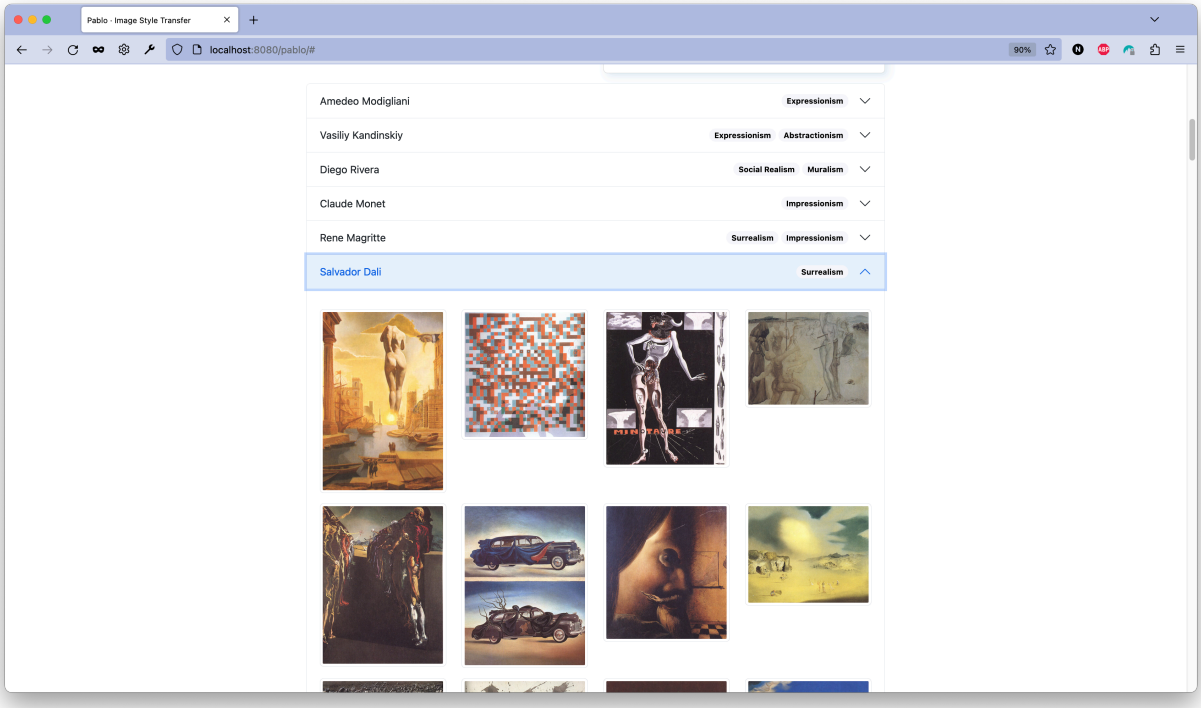

Abbildung A.2: Liste der Künstler und entsprechender Kunstwerke, Quelle: Eigene Darstellung

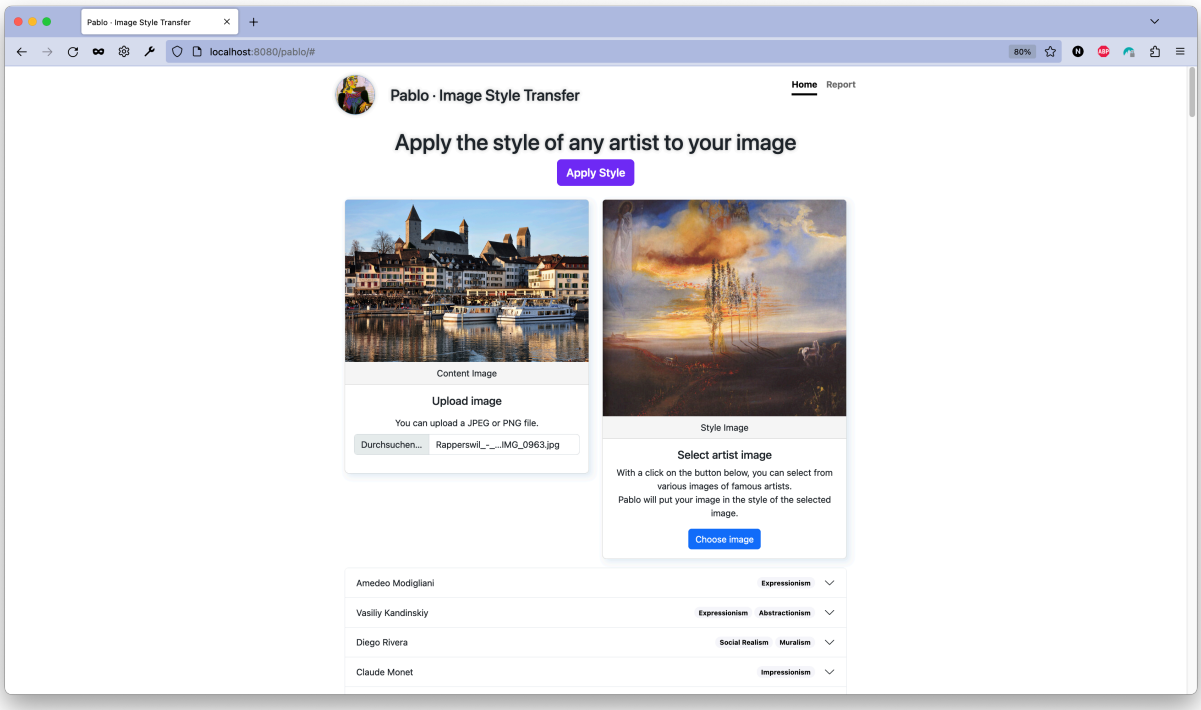

Abbildung A.3: Das ausgewählte Inhalts- und Stilbild in „Pablo“, Quelle: Eigene Darstellung

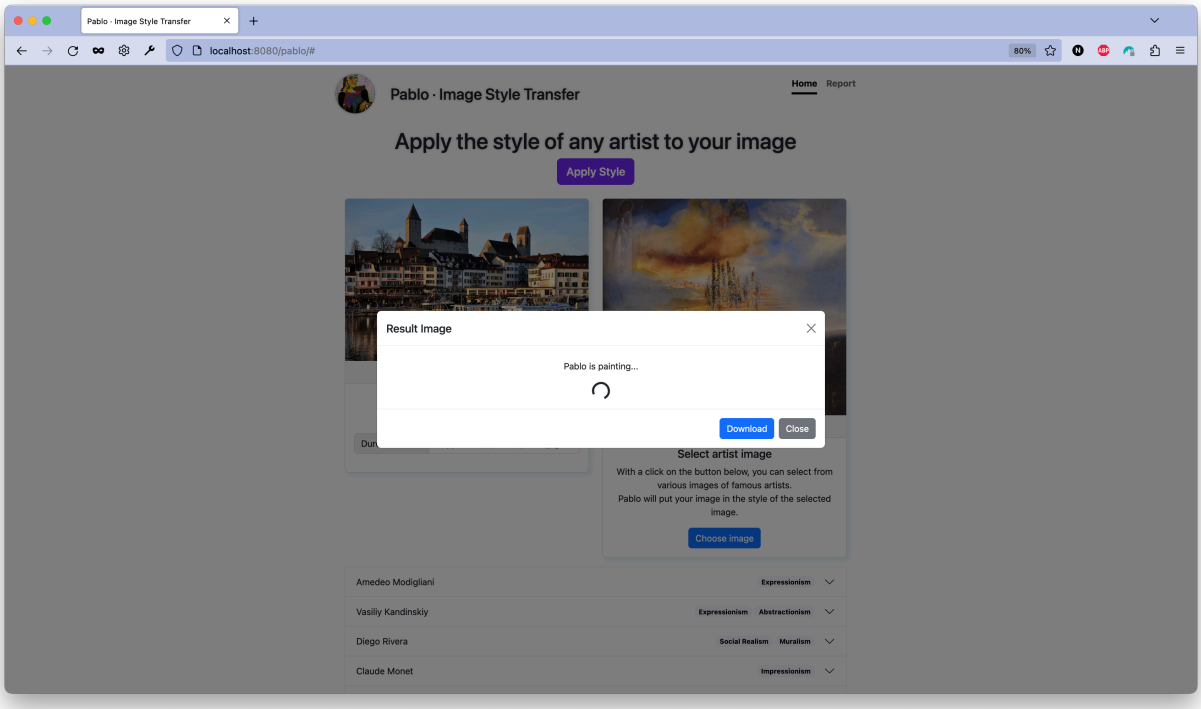

Abbildung A.4: Das aus den Modellberechnungen resultierende Bild wird geladen, Quelle: Eigene Darstellung

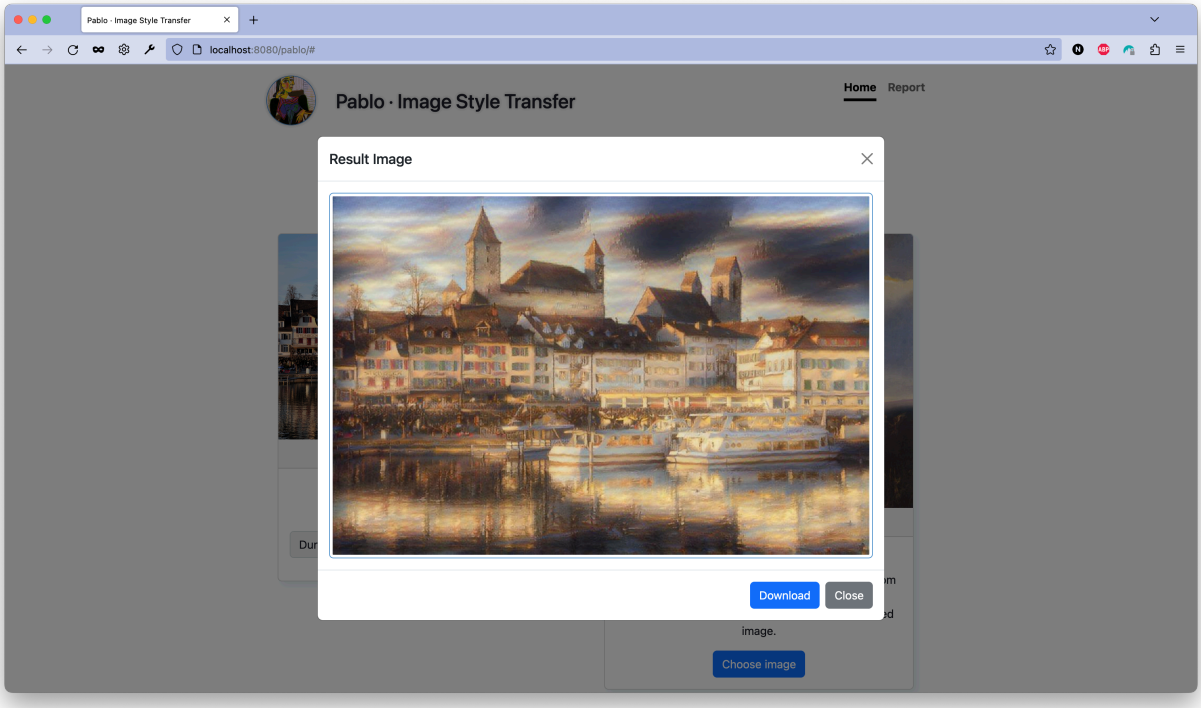

Abbildung A.5: Das Ergebnis wird in einem Popup-Fenster angezeigt, Quelle: Eigene Darstellung

Referenzen

[1] *Magenta: Music and Art Generation with Machine Intelligence*. URL: https://github.com/magenta/magenta (besucht am 22. 05. 2023).

[2] Abhishek Mishra. *Understanding Google Magenta: An Overview of Google's Open-Source Music and Art Project*. März 2023. URL: https://medium.com/@abhishekmishra13k/understanding-google-magenta-an-overview-of-googles-open-source-music-and-art-project-48ea9ee80024 (besucht am 22. 05. 2023).

[3] *Fast Style Transfer for Arbitrary Styles*. URL: https://github.com/magenta/magenta/tree/main/magenta/models/arbitrary_image_stylization (besucht am 22. 05. 2023).

[4] Leon A. Gatys, Alexander S. Ecker und Matthias Bethge. *A Neural Algorithm of Artistic Style*. 2015. arXiv: 1508.06576 [cs.CV].

[5] Golnaz Ghiasi u. a. *Exploring the structure of a real-time, arbitrary neural artistic stylization network*. 2017. arXiv: 1705.06830 [cs.CV].

[6] reddit - r/tensorflow. *A Powerful Machine Intelligence Library*. URL: https://www.reddit.com/r/tensorflow/ (besucht am 21. 05. 2023).

[7] TensorFlow Magenta. *Community*. URL: https://magenta.tensorflow.org/community (besucht am 22. 05. 2023).

[8] TensorFlow Magenta. *Talks*. URL: https://magenta.tensorflow.org/talks/ (besucht am 22. 05. 2023).

[9] TensorFlow Hub. *magenta/arbitrary-image-stylization-v1-256 - Fast arbitrary image style transfer.* URL: https://tfhub.dev/google/magenta/arbitrary-image-stylization-v1-256/2 (besucht am 27. 05. 2023).

[10] Karen Simonyan und Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV].

[11] Christian Szegedy u. a. *Rethinking the Inception Architecture for Computer Vision*. 2015. arXiv: 1512.00567 [cs.CV].

[12] *Inception-v3 Explained | Papers With Code*. URL: https://paperswithcode.com/method/inception-v3 (besucht am 27. 05. 2023).

[13] TensorFlow Tutorials. *Neural style transfer*. Dez. 2022. URL: https://www.tensorflow.org/tutorials/generative/style_transfer (besucht am 21. 05. 2023).

[14] The TensorFlow Hub Authors. *TF-Hub Fast Style Transfer for Arbitrary Styles - Colaboratory*. 2019. URL: https://colab.research.google.com/github/tensorflow/hub/blob/master/examples/colab/tf2_arbitrary_image_stylization.ipynb (besucht am 22. 05. 2023).

[15] Icaro on kaggle.com. *Best Artworks of All Time | Kaggle*. Collection of Paintings of the 50 Most Influential Artists of All Time. URL: https://www.kaggle.com/datasets/ikarus777/best-artworks-of-all-time (besucht am 22. 05. 2023).

[16] University of Cambridge. *What is a Tensor?* URL: https://www.doitpoms.ac.uk/tlplib/tensors/what_is_tensor.php (besucht am 27. 05. 2023).

Abbildungsverzeichnis

Neural Image Style Transfer

Neuronaler Stiltransfer für Bilder mit vortrainiertem Modell aus dem Magenta-Projekt Ein Report

von

Nico Fehr, Kyra Maag

Name Student	E-Mail
Nico Fehr	nico.fehr@ost.ch
Kyra Maag	kyra.maag@ost.ch

Dozent: M. Lehmann
Projektdauer: April, 2023 - Mai, 2023

Titelbild: Rapperswil (SG): Sicht vom Seedamm auf die Altstadt und den Hafen, im Hintergrund das Schloss und die Stadtpfarrkirche ©Roland Fischer CH-8050 Zürich bearbeitet durch das Magenta Modell mit Bildern von Pablo Picasso aus dem Kaggle Datenset „Best Artworks of All Time“

Style: EPFL Report Style, mit Modifikationen von Batuhan Faik Derinbay und Nico Fehr