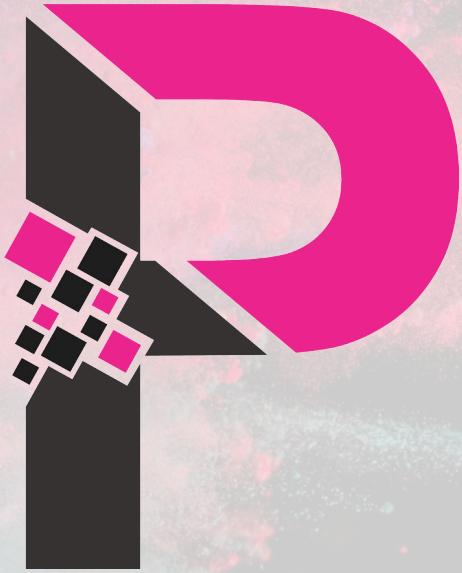
Pixorize

Fotokolorierung mittels Machine Learning

Stefan Etzelstorfer, Manuel Riedl, Stefan Haas



Projektbericht - Jugend Innovativ - Science

HTL Wels
Fischergasse 30
Österreich
Tel.Nr: 07242 65801
E-Mail: office@htl-wels.at

1 Formelle Daten

Projekt-Titel
Pixorize

Projekt-Nummer
J200166

Kategorie und Spezifikation
Science: Mathematik

Schule
HTL Wels
Fischergasse 30, Wels 4600
+43 7242 65801
office@htl-wels.at

Projektteam
Stefan Etzelstorfer, 5AHIT
Manuel Riedl, 5AHIT
Stefan Haas, 5AHIT

Projektkoordinator
Stefan Haas
+43 677 61201018
stefan.haas.privat@gmail.com

Projektbetreuer
DI Dr. Erich Gams
erich.gams@htl-wels.at

2 Abstract

The goal of Pixorize is the implementation of a new machine learning approach for the colorization of black and white images.
In contrast to other approaches, which handle image colorization completely automatically, Pixorize integrates the user by manually setting color-pixels on the black and white image. This allows more accurate results to be achieved. As a result, different filiations of the same image may be possible.

3 Einleitung

Die Diplomarbeit Pixorize beschäftigt sich mit einem neuen Ansatz, Schwarz-Weiß-Fotos zu kolorieren - sprich beispielsweise alte Familien-Fotos und Porträts von großen Persönlichkeiten der Geschichte, welche alle in Form von Schwarz-Weiß-Bildern vorliegen, in Farbe und somit in neuem Glanz erscheinen zu lassen.
Dabei bestimmt die Benutzerin/der Benutzer selbst, wie das Ergebnis – also das zu entstehende Farbfoto - aussehen soll. Dies wird durch gezieltes Setzen von Farb-Punkten ermöglicht, welche aus ein und demselben Schwarz-Weiß-Bild verschiedene Resultate hervorbringen. So ist es beispielsweise der Benutzerin/dem Benutzer offen und frei, wie er die Kleidung, der im Bild vorhandenen Personen färbt, welche Farbe die Landschaft im Hintergrund, etc. erhält.

Das Hauptaugenmerk von Pixorize: Der Gestaltungskraft der Benutzerin/des Benutzers freien Lauf zu lassen und demzufolge die Mannigfaltigkeit der entstehenden Resultat-Fotos zu gewährleisten.

Abbildung 1: Mit Pixorize koloriertes Porträt von Richard Feynman

Inhaltsverzeichnis

4 Projektdokumentation

4.1 Projektentstehung

4.1.1 Die Idee hinter Pixorize

Jeder besitzt alte Familien-Fotos, welche meist nur im Schwarz-Weiß-Format vorliegen. Oft kann man sich unter den Schwarz-Weiß-Fotos wenig vorstellen, es fehlt der für den Menschen wichtigste Teil: Farbe.
Schon des Öfteren kam der Gedanke, wie beispielsweise die Hochzeits-Fotos, Porträts oder sonstige Schwarz-Weiß-Fotos der Familie in Farbe aussehen würden. Leider stellte sich die Kolorierung von Eigenhand in Programmen wie Photoshop, aber auch analog, am eigentlichen Schwarz-Weiß-Foto, als eine große Herausforderung heraus. Auch die Alternative, sich die Schwarz-Weiß-Fotos von professionellen Anbietern färben zu lassen, stellt keine Lösung dar: Hierbei muss man mit hohen Kosten rechnen. Des Weiteren ist nicht gewährleistet, dass das Ergebnis den eigenen Vorstellungen schlussendlich entspricht.
All diese Gründe und vor allem auch das Interesse an künstlichen neuronalen Netzen brachte uns auf die Idee von Pixorize.[15]

4.1.2 Ziele von Pixorize

Innovativer Machine Learning Algorithmus
: Das Hauptaugenmerk von Pixorize liegt auf einem innovativen Machine Learning-Algorithmus, der die Aufgabe einer aufwändigen Hand-Kolorierung übernimmt. Im Fokus stehen dabei Schwarz-Weiß-Familien-Fotos und Schwarz-Weiß-Porträts.

Benutzerinteraktivität
: Mithilfe von der Benutzerin/dem Benutzer gesetzten Farb-Punkten, können beispielsweise Kleidungsstücke wie Pullover, Jacken, usw. individuell eingefärbt werden. Somit kann die Benutzerin/der Benutzer direkt mitbestimmen, welches Foto als Ergebnis entstehen soll.

Intuitive Benutzeroberfläche
: Des Weiteren wird ein großes Augenmerk auf eine intuitive, leicht bedienbare Grafische Benutzeroberfläche gelegt, welche der Benutzerin/dem Benutzer verschiedene Funktionen zur Bildverarbeitung zur Verfügung stellt.[12] Hier werden auch die erwähnten Farb-Punkte gesetzt.

4.1.3 Nicht-Ziele von Pixorize

Kolorierung von Bildern außerhalb der Domäne Schwarz-Weiß-Familien-Fotos
: Ein künstliches neuronales Netz benötigt enorme Mengen an Daten, um korrekte Resultate liefern zu können. Hinzu kommt ein enormer Zeitaufwand für das Anlernen des künstlichen neuronalen Netzes. Aufgrund mangelnder Ressourcen und Zeit, können daher nicht alle Arten von Schwarz-Weiß-Fotos (Schwarz-Weiß-Landschaftsfotos, -Tierfotos, etc.) gehandhabt werden. Jedoch kann dies bei Bedarf später noch ergänzt werden.

Webanwendung
: Aufgrund zeitlicher Begrenzung beschränkt sich Pixorize auf eine Desktopanwendung, da eine Webanwendung den Rahmen der Diplomarbeit sprengen würde.

4.1.4 Recherche und Quellen

Machine Learning stellt eine komplexe Thematik dar. Dementsprechend ist eine ausgiebige Recherche nötig. Vor allem das World-Wide-Web stellt hierbei viele Informationen zur Verfügung. Auf vielen Webseiten werden die meisten Begrifflichkeiten erklärt und einfache Beispiele angeführt. Wenn die benötigten Informationen spezifischer werden, muss in

mühseliger Selektierung von verschiedensten Webseiten eine Lösung zusammengefügt werden. Dies ist vor allem der Fall, wenn Probleme während der Entwicklung auftreten: Hier kann es durchaus schon vorkommen, dass die umgesetzte Lösung sich von mehr als zehn Quellen zusammensetzt.
Somit ist eine Auflistung aller Quellen ein Ding der Unmöglichkeit.
Im Folgenden werden die (wichtigsten) Hauptquellen, die zur Recherche dienten, aufgelistet und es wird erläutert, wie sie zur Umsetzung von Pixorize beitrugen:

Wikipedia
Vor allem bei der Recherche-Arbeit konnte viel Wissen in den Bereichen Machine Learning, Datenaufbereitung und GUI durch Wikipedia dazugewonnen werden. Bei Wikipedia bestehen zu den meisten Themen, die in Pixorize umgesetzt wurden, Unterseiten, welche die Begriffe und Themen einfach erklären - ideal zur Beseitigung von Verständnisproblemen.

Stackoverflow.com
Stack Overflow stellt eine der größten Internetforen für Programmierer dar. Hier kann jeder, der einen Account besitzt, eine Frage stellen, welche von professionellen Programmierern meist schon in wenigen Stunden beantwortet wird[18]. Gerade bei der Implementierung des Machine Learning-Algorithmus konnten viele Probleme gelöst werden. Oft entstanden hierbei Lösungen, die auch vielen anderen Benutzerinnen/Benutzern von Stack Overflow noch in Zukunft helfen werden.

Weitere nützliche Links zu den Themen:

- Machine Learning
  - `https://algorithmia.com/blog/convolutional-neural-nets-in-pytorch`
  - `https://pytorch.org/docs/stable/index.html`
  - `https://towardsdatascience.com/review-fcn-semantic-segmentation-eb8c9b50d2d1`
  - `https://distill.pub/2016/deconv-checkerboard/`
  - `https://github.com/udacity/deep-learning-v2-pytorch/tree/master/autoencoder/convolutional-autoencoder`
- Datenaufbereitung
  - `https://de.talend.com/resources/what-is-data-preparation/`
  - `https://www.empirical-methods.hslu.ch/forschungsprozess/quantitative-forschung/datenaufbereitung/`
  - `https://www.bigdata-insider.de/datenaufbereitung-ist-ein-unterschaetzter-prozess-a-803469/`
  - `http://www.gitta.info/ThematicCart/de/html/themStatDat_learningObject1.html`
  - `https://www.altair.com/what-is-data-preparation/`
- Grafische Benutzeroberfläche
  - `https://www.w3schools.com/js/`
  - `https://www.it-talents.de/blog/it-talents/javascript-guide-was-ist-javascript`
  - `https://www.electronjs.org/`
  - `https://t3n.de/news/electron-javascript-desktop-apps-1084656/`
  - `https://www.grapecity.com/blogs/how-to-build-electron-app-with-javascript-ui-controls`

4.2 Projektplanung

4.2.1 Scrum

Bei Pixorize wurde das Projektmanagement mittels Scrum umgesetzt. Dabei handelt es sich um ein sogenanntes Vorgehensmodell der agilen Softwareentwicklung, welches sich durch selbstorganisierende Teams und eine inkrementelle Vorgehensweise auszeichnet.[17][8] Durch Scrum können sich - im Gegensatz zum herkömmlichen Projektmanagement - auch die Anforderungen während des Projektes ändern. Denn gerade in der Softwareentwicklung ändern sich die Anforderungen ständig. Gründe dafür:

- Neue Ideen, welche während der Entwicklung entstehen
- Unerwartete Ereignisse, für die Alternativen gesucht werden müssen

Gerade deswegen eignet sich Scrum sehr gut für Pixorize.

4.2.2 Meilensteine

Des Weiteren wurden grobe Meilensteine gesetzt:

Tabelle 1: Meilensteine

Datum	Meilenstein
01.10.2019	Recherche abgeschlossen
01.10.2019	Daten normalisiert
01.10.2019	KI-Prototyp entwickelt
01.10.2019	GUI-Prototyp entwickelt
20.12.2019	Funktionsprototyp 1 entwickelt
04.03.2020	Funktionsprototyp 2 entwickelt

4.2.3 Aufgabenbereiche

Pixorize kann insgesamt in drei große Kernbereiche geteilt werden: Die Datenaufbereitung, das künstliche neuronale Netz und die Grafische Benutzeroberfläche. Demzufolge wurden diese auf die Projektmitglieder aufgeteilt, mit folgenden Arbeitspaketen:

Stefan Etzelstorfer
Implementierung der Datenaufbereitung (180 Stunden):

- Recherche zu Python, OpenCV, Pillow, NumPy, Image Processing, Computer Vision
- Entwicklung eines unintelligenten Datapreparers (Farb-Punkte werden rein zufällig in einem Bild platziert; Benutzereingaben werden simuliert)
- Entwicklung eines intelligenten Datapreparers (die Farb-Punkte, werden vermehrt auf Gesichtern und Kleidung gesetzt; natürliche und Benutzereingaben werden simuliert)

Stefan Haas
Implementierung des künstlichen neuronalen Netzes (180 Stunden):

- Recherche zu Machine Learning, Python, Pytorch
- Architektur des künstlichen neuronalen Netzes entwickeln
- Künstliches neuronales Netz trainieren
- Künstliches neuronales Netz testen, auswerten und debuggen

Manuel Riedl

Implementierung der Grafischen Benutzeroberfläche (180 Stunden):

- Recherche zu Electron, Python, JavaScript und allgemein zur Entwicklung einer Grafischen Benutzeroberfläche
- Architektur der KI entwickeln
- GUI-Prototyp entwickeln
- Software-Architektur entwickeln (Kommunikation zwischen der GUI und dem neuronalem Netz)
- Computer-Vision-Funktionen entwickeln

4.3 Projektumsetzung

4.3.1 Technologien-Stack

Für die Implementierung von Pixorize wurde auf eine große Menge an Technologien verwendet. In diesem Abschnitt werden diese erläutert.

Übergreifend verwendete Technologien Pixorize stellt eine Desktopanwendung dar. Diese enthält alle Komponenten: Die Datenaufbereitung, das künstliche neuronale Netz und die Grafische Benutzeroberfläche. Damit diese Komponenten miteinander kommunizieren können, war eine übergreifende Technologie nötig: Python. [16]

Python

- Bei Python handelt es sich um eine sogenannte Interpretersprache. Das Hauptaugenmerk dieser liegt auf einem gut lesbaren, knappen Programmierstil. Sie findet vor allem im Bereich des Machine Learnings Anwendung - ideal für Pixorize.
  Mithilfe von Python wurde die Datenaufbereitung implementiert, das Neuronale Netzwerk entwickelt und die Kommunikation zwischen der Grafischen Benutzeroberfläche und den anderen Komponenten bewerkstelligt.

Folgende Frameworks kamen in den jeweiligen Bereichen zum Einsatz:

Datenaufbereitung

- OpenCV [4]
  - Bei OpenCV handelt es sich um eines der Kern-Frameworks zur Fotomanipulation. Vom einfachen Laden der Bilder, bis hin zu komplexen Manipulationen von Fotos, OpenCV bewerkstelligt es.
- Pillow [6]
  - Auch Pillow stellt ein sehr mächtiges Framework dar, das auch im Bereich der Fotomanipulation zum Einsatz kommt. Dabei wurde es hauptsächlich bei einfachen Fotomanipulationen verwendet (Änderung der Größe eines Bildes, Speichern des Bildes).
- NumPy [3]
  - Digitale Fotos entsprechen in der Informatik Matrizen, welche alle Informationen zu einem Foto enthalten. Künstliche neuronale Netze arbeiten mit solchen Matrizen, so dass sie auch Fotos "verstehen"können - denn erst durch Zahlen und Werten kann ein künstliches neuronales Netz dazulernen. NumPy ist eine Bibliothek, welche einfaches Arbeiten mit Matrizen, Vektoren und deren Operationen ermöglicht.

Künstliches neuronales Netz

- PyTorch [7]
  - PyTorch ist eine Open-Source-Bibliothek, welche von Facebooks Abteilung für künstliche Intelligenz entwickelt wurde und stellt einen Bausatz für Machine Learning dar. Es ermöglicht das Erstellen, Trainieren, Persistieren und Laden von künstlichen neuronalen Netzen, aber auch vieles mehr.
- Matplotlib [2]
  - Matplotlib ist eine Python-Bibliothek, welche mathematische Darstellung jeglicher Art ermöglicht. In Verbindung mit Machine Learning wird diese Bibliothek sehr häufig verwendet Statistiken zum Überprüfen der Lernfähigkeit eines künstlichen neuronalen Netzes.

Grafische Benutzeroberfläche

- JavaScript und Electron [14]
  - Die Grafische Benutzeroberfläche wurde mittels JavaScript und dem dazugehörigen JavaScript-Framework Electron implementiert. Mithilfe von Electron können aus einfachem JavaScript-Code komplexe Desktopanwendungen entwickelt werden. Dabei vereint Electron die besten Konzepte von Desktop- und Webanwendungen zu einem Framework mit unzähligen Funktionen. Ein einzigartiger Vorteil von Electron ist die Verwendung betriebssysteminterner Funktionen wie der Aufruf des File-Systems oder die Benachrichtigung durch Pop-Up-Fenster. Mittels JavaScript und den dazugehörigen Bibliotheken wurden die alle Funktion, die der Bildbearbeitung dienten, umgesetzt.
- HTML und CSS [13] [9]
  - Mithilfe von HTML und CSS wurde die Strukturierung und das Design umgesetzt. Als Hauptgrundlage für die Gestaltung der GUI wurde das für CSS-Layout Photonkit verwendet. Photonkit wurde speziell für Desktopanwendungen entwickelt und ist im simplen Apple Design gehalten.

4.3.2 Implementierung der Datenaufbereitung

Einleitung Künstliche neuronale Netze benötigen fehlerfreie Datensätze, damit ein erfolgreiches Anlernen des Netzes möglich ist. Dies wird durch die sogenannte Datenaufbereitung gewährleistet.
Die Datenaufbereitung, oder auch Datapreparation im Englischen genannt, umfasst das Sammeln, Transformieren, Bereinigen und Bereitstellen von Rohdaten. Dabei sind die verwendeten Daten in ihrer ursprünglichen Form oft inkonsistent, fehlerhaft und nicht standardisiert. Dies ist vor allem der Fall, wenn die Daten aus mehreren beziehungsweise verschiedenen Quellen stammen. Bei Pixorize handelt es sich bei den aufzubereitenden Rohdaten um Farbfotos, welche eine Reihe von Transformationen und Schritte der Aufbereitung durchlaufen. Das Ziel der Datenaufbereitung ist die Erzeugung eines qualitativ hochwertigen Datensatzes - ein solcher stellt vor allem im Bereich des Maschinellen-Lernens eine zentrale Rolle dar. Denn die meisten Algorithmen des Maschinellen-Lernens erfordern eine sehr spezifische Formatierung der Daten, so dass die Datensätze im Allgemeinen eine gewisse Vorbereitung erfordern, bevor sie nützliche Erkenntnisse beziehungsweise Informationen liefern können. Oft haben Datensätze Werte, die fehlen, ungültig oder anderweitig für einen Algorithmus schwierig zu verarbeiten und zu interpretieren sind[10].
Dabei könnten folgende Probleme auftreten[10]:

- Wenn Daten fehlen, kann der Algorithmus sie nicht verwenden
- Wenn Daten ungültig oder fehlerhaft sind, erzeugt der Algorithmus weniger genaue oder sogar irreführende Ergebnisse

Somit ist eine gute Datenaufbereitung für Maschinelles-Lernen unentbehrlich, da erst durch eine solche, fehlerfreie Ergebnisse gewährleistet werden können.

Schritte der Datenaufbereitung Die Datenaufbereitung umfasst insgesamt fünf Schritte. Diese können je nach Branche oder Einsatzgebiet variieren. Die folgenden Schritte wurden in Pixorize angewendet[5]:

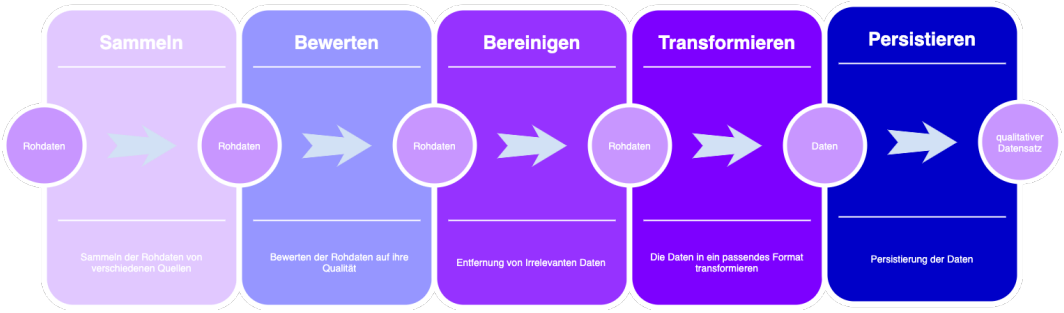


Abbildung 2: Schritte der Datenaufbereitung

1. Sammeln der Daten
   Der wichtigste Schritt der Datenaufbereitung stellt das Sammeln der Daten dar. Denn nur durch das Vorhandensein dieser können weitere Schritte zur Aufbereitung getätigt werden. Noch vor dem Sammeln der Daten ist es notwendig, sich Gedanken zu machen, welche Daten überhaupt benötigt werden. Denn sie entscheiden, welche Resultate mit dem daraus entstehenden Modell, geliefert werden sollen. Wenn feststeht, welche Daten benötigt werden, erfolgt das eigentliche Sammeln der Rohdaten.

   Wie wurde dies in Pixorize umgesetzt?
   Bei Pixorize handelt es sich bei den zu sammelnden Rohdaten um Porträt- oder Familien-Fotos. Dementsprechend wurde ein passender Datensatz gesucht. Aufgrund der speziellen Anwendung von Pixorize - nämlich die Kolorierung von Schwarz-Weiß-Fotos - stellte dies eine große Herausforderung dar. Folgende Anforderungen wurden an den Datensatz gestellt, welche die Suche erschwerten: Die Fotos sollen wenig Landschaft oder Tiere im Hintergrund enthalten, der Anteil an Frau und Mann sollte gleich sein. Erst durch einen lange andauernden Suchprozess konnte ein passender Datensatz im World-Wide-Web gefunden werden, der folgendermaßen erreichbar ist: `https://github.com/NVlabs/ffhq-dataset`

   Hier ein paar Ausschnitte aus dem Datensatz:

Abbildung 3: Beispielbild 1

Abbildung 4: Beispielbild 2

Abbildung 5: Beispielbild 3

2. Bewerten der Daten
   Als nächstes folgt ein ebenso wichtiger Schritt: Das Bewerten der Daten. In diesem geht es vor allem um das Verstehen und die Bewertung der Daten.
   Wichtig hierbei ist, sich genau Gedanken zu machen, was später mit den Rohdaten umzusetzen ist – also welche Ergebnisse damit erzielt werden sollen. Dementsprechend werden nötige Schritte und Maßnahmen überlegt, um die Daten für einen bestimmten Kontext brauchbar zu machen. Durchaus kann dieser als schwerster aller Schritte betrachtet werden, denn hierbei dürfen keine Fehler passieren. Werden dennoch Fehler gemacht, so kann es durchaus vorkommen, dass von vorne begonnen werden muss.

   Wie wurde dies in Pixorize umgesetzt?
   Bei Pixorize sollen als Ergebnis aus Schwarz-Weiß-Familien oder -Porträts kolorierte Farbfotos entstehen. Somit muss der für das Anlernen des künstlichen neuronalen Netzes benötigte Datensatz folgendermaßen aussehen (auf eine genaue Erläuterung wird verzichtet - dies würde den Rahmen dieser Ausarbeitung sprengen):

   - Farbfotos
   - Dazugehöriges Schwarz-Weiß-Foto mit zufällig gesetzten Farb-Punkten (diese Farb-Punkte stellen zufällig simulierte Benutzerinputs dar – genauer gesagt die erwähnte Benutzerinteraktivität)

   In den anschließenden Seiten wird für ein besseres Verständnis näher darauf eingegangen.

3. Bereinigen der Daten
   Nun, da man sich Gedanken zu den Maßnahmen gemacht hat, folgt der Schritt der Bereinigung dieser Daten. Nomen est omen, werden fehlerhafte Daten aber auch inkonsistente Daten hierbei bereinigt oder gelöscht.

   Wie wurde dies in Pixorize umgesetzt?
   Pixorize verwendet einen Datensatz, welcher aus Fotos besteht. Gründe, warum Fotos bereinigt beziehungsweise gelöscht werden müssen:

   - Beim Download von großen Datensätzen, die aus Fotos bestehen, kann es durchaus vorkommen, dass Teile dieser korrupt sind – diese Fotos können nicht verwendet, sondern müssen entfernt werden. Bei Pixorize erfolgte dies automatisiert.
   - Oft befinden sich in einem Datensatz auch gewisse Ausreißer: Also Fehler, die von den Entwicklern dieses Datensatzes gemacht wurden: So war es bei dem von Pixorize verwendeten Datensatz oft der Fall, dass Fotos verwackelt wurden, die Belichtung von Bildern nicht unseren Anforderungen entsprach. In mühseliger Arbeit mussten genau solche Fotos selektiert und anschließend aus dem Datensatz entfernt werden.

4. Transformieren der Daten
   Alle Daten sind nun von Fehlern und Inkonsistenz bereinigt. Es folgt nun eine Transformation der Daten. Das bedeutet, dass die Daten in ein gewisses Format gebracht werden.

   Wie wurde dies in Pixorize umgesetzt?
   Wie schon vorher erwähnt, muss der entstehende Datensatz folgendermaßen aussehen:

   - Farbfotos
   - Die dazugehörigen Schwarz-Weiß-Fotos mit zufällig gesetzten Farb-Punkten

   Beim Schritt des Transformierens wird der Datensatz genau in diese Form gebracht. Auch hier gilt wieder: Die genaue Implementierung dieses Schrittes würde den Rahmen dieser Ausarbeitung sprengen. Jedoch wird der grobe Ablauf hier erläutert:

(a) Laden der Fotos aus dem Datensatz (der gedownloadete Datensatz entspricht einem Ordner)

(b) Konvertierung der Farbfotos ins Schwarz-Weiß-Format

(c) Einfügen von zufälligen Farb-Punkten im Schwarz-Weiß-Foto (diese Farb-Punkte werden aus dem ursprünglichen Farbfotos zu einer gewissen Pixel-Koordinate ausgelesen und an der entsprechenden Position des Schwarz-Weiß-Fotos wieder eingefügt)

(d) Des Weiteren werden auch noch weitere Schritte, wie das mehrfache Ändern der Farbräume getätigt. Auf dies wird später noch eingegangen.

5. Persistieren der Daten
   Der letzte Schritt der Datenaufbereitung, das Persistieren, wird benötigt, um die transformierten und verbesserten Daten wieder zu speichern.

   Wie wurde dies in Pixorize umgesetzt?
   Nach all den vorherigen Schritten werden die Daten gespeichert und können für das Anlernen verwendet werden. Bei Pixorize wurden die Fotos in folgenden Verzeichnissen gespeichert:

   - Farbfotos: Hier befinden sich die Farbfotos
   - Schwarz-Weiß-Fotos: Hier befinden sich die zu den Farbfotos dazugehörigen, transformierten Schwarz-Weiß-Fotos mit den gesetzten Farb-Punkten. Diese Ordner entsprechen dem entstandenen fehlerfreien, konsistenten Datensatz.

Hier ein paar Ausschnitte aus dem entstandenen Datensatz:

Abbildung 6: Farbfoto (Ausgangsbild)

Abbildung 7: Schwarz-Weiß-Foto

Abbildung 8: Schwarz-Weiß-Foto mit Farb-Punkten

4.3.3 Implementierung des künstlichen neuronalen Netzes

Einleitung Künstliche neuronale Netze sind mathematische Konstrukte, welche von der Biologie inspiriert sind und versuchen, eine künstliche Intelligenz zu erschaffen. Man verknüpft eine große Menge von künstlichen Neuronen, man sagt dann, dass das künstliche neuronale Netz verschiedene Schichten besitzt. Die Neuronen der ersten Schicht sind mit allen Neuronen der nächsten Schicht verbunden. Diese Schicht ist wieder mit deren nächsten Schicht verbunden. Die Verknüpfungen von Neuronen kann jedoch sehr speziell und divers sein, deshalb unterscheidet man zwischen verschiedenen Netzwerkarchitekturen.
Die Entwicklung des künstlichen neuronalen Netzes war der empirischen Methode ähnlich, denn die Entwicklung durchlief folgende drei Phasen zyklisch:

- Entwicklung eines künstlichen neuronalen Netzes beziehungsweise Anpassungen bei einem bestehendem Neuronalen Netzwerk

- Trainieren des künstlichen neuronalen Netzes mithilfe des Datensatzes
- Evaluierung der Resultate

Sozusagen wurde eine These in Form eines künstlichen neuronalen Netzes aufgestellt, welche anschließend durch das Trainieren des Datensatzes geprüft wurde. Anschließend wurde das Ergebnis evaluiert und ein neuer Zyklus konnte eingeleitet werden.

Entwurf Beim Entwurf müssen viele Faktoren beachtet werden, denn ob eine Netzwerkarchitektur funktioniert, hängt von der Hardwarelimitierung, dem Datensatz und dem Anwendungsfall ab.
Hat man zum Beispiel wenig, beziehungsweise schlechte Hardware-Ressourcen, so sollte man nicht allzu viele Neuronen implementieren, ansonsten würde die Trainingszeit enorm hoch sein. Wenn nicht sogar das Training aufgrund starker Hardwarelimitierungen fehlschlägt.
Die Daten werden normalisiert, indem sie in mehrdimensionale Matrizen mit einem bestimmten Wertebereich konvertiert werden. Demnach muss das künstliche neuronale Netz die Dimension, als auch den Wertebereich, der Daten beachten.
Die Wahl der richtigen Architektur ist stark von der Anwendung abhängig. Möchte man, wie bei Pixorize, Bilder verarbeiten, so sollte man ein Convolutional Neural Network in Form eines Autoencoders verwenden. Schließlich wird ein Convolutional Neural Network benötigt, um Bilder zu verarbeiten. Außerdem wird eine Autoencoder-Architektur verwendet, da diese in der Lage ist, das Bild zu manipulieren, sprich im speziellen Fall bei Pixorize das Bild zu kolorieren.

Debugging Natürlich treten während der Programmierung des künstlichen neuronalen Netzes Fehler auf, welche für gewöhnlich durch Debugging behoben werden können. [11] Indem man ganz detailliert eine Zeile Code nach der anderen inspiziert, dabei Werte von Variablen einsieht, anschließend versucht den Algorithmus nachzuvollziehen und schlussendlich den Fehler beseitigt.

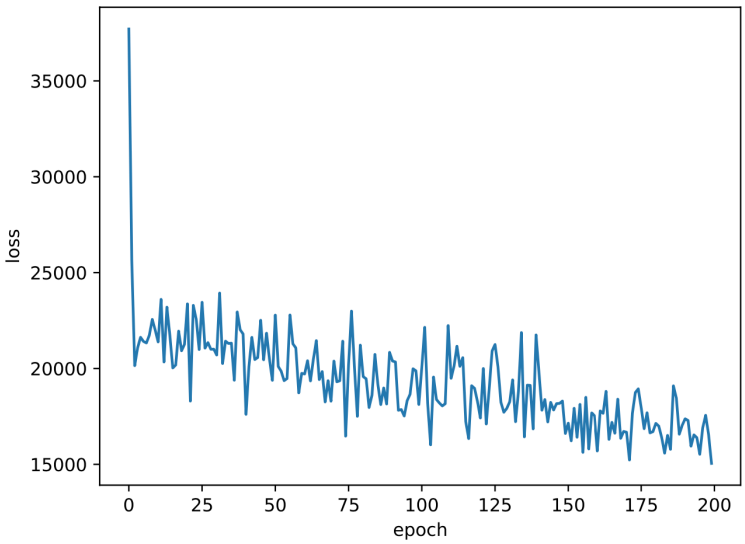


Abbildung 9: Durchschnittliche Fehlerrate pro 300 Iterationen

Jedoch begrenzt sich Debugging bei Machine Learning nicht ausschließlich auf das Inspizieren von Code. Denn man muss nach einem Training die Genauigkeit des künstlichen neuronalen Netzes überprüfen, indem man eine Statistik unter die Lupe nimmt. In dieser Statistik sollte die Fehlerrate pro Bilditeration dokumentiert sein, um so die Lernfähigkeit über einen Zeitraum zu überwachen. Hierbei ist zu beachten, dass die Fehlerrate abnimmt - idealerweise exponentiell. Wie in Abbildung 9 gut zu sehen ist, verhält sich die Fehlerrate

degressiv über den Gesamten Zeitraum des Trainings und somit ist klar erkennbar, dass das künstliche neuronale Netz in diesem Fall keine Fehler aufweist.

4.3.4 Implementierung der Grafischen Benutzeroberfläche

Einleitung Die Grafische Benutzeroberfläche, im Englischen Graphical User Interface (GUI), bezeichnet die grafische Schnittstelle zwischen Mensch und Computer. Pixorize soll für jedermann - auch ohne große Computerkenntnisse - verwendbar sein. Gerade deswegen spielt eine GUI solch eine große Rolle.

Foto laden Dieses Fenster erscheint beim Start von Pixorize. Dieses beinhaltet das Logo, eine kurze Auflistung der Funktionen von Pixorize und die Möglichkeit ein Foto zu laden.

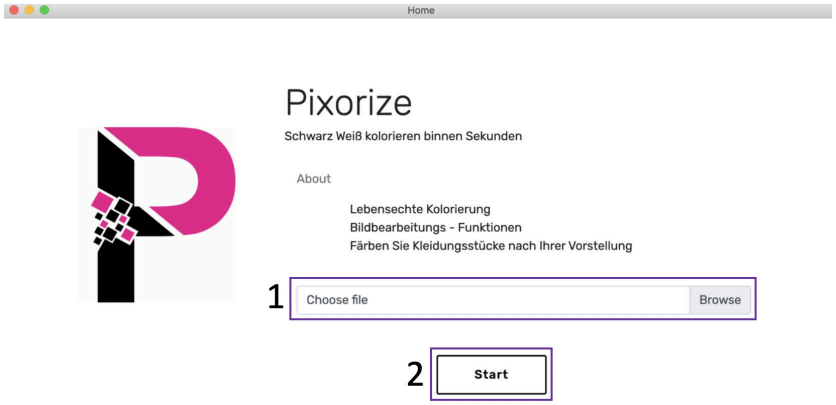


Abbildung 10: Laden des Fotos

1. Durch den File-Manger hat der Benutzer die Möglichkeit, sein Foto auszuwählen
2. Beim erfolgreichen Laden eines Fotos hat der User die Möglichkeit, das Bearbeitungs-Fenster aufzurufen

Foto bearbeiten Das Bearbeitungs-Fenster erscheint nach erfolgreichem Laden eines Bildes. Dieses Fenster ist der Kern der Grafischen Benutzeroberfläche. Es beinhaltet alle Funktionen, die für die Bearbeitung notwendig sind. Das Fenster ist in drei Sektionen gegliedert: Werkzeugleiste – Bearbeitungsfläche – Farbauswahl.

1. Der Benutzerin/Dem Benutzer wird durch die Zurück-Funktion die Möglichkeit geboten, fehlerhaft gesetzte Farb-Punkte wieder zu entfernen
2. Durch Aufrufen der Zoom-Funktionen, kann das Bild skaliert werden
3. Beim Auswählen der Paint-Funktion, kann man Farb-Punkte auf dem Bild setzen
4. Durch Auswählen der Move-Funktion, kann die Benutzerin/der Benutzer das Bild bewegen
5. Beim Klicken auf Pixorize, wird das Foto mithilfe des künstlichen neuronalen Netzes koloriert
6. Der Color-Picker ermöglicht es der Benutzerin/dem Benutzer, eine Farbe auszuwählen

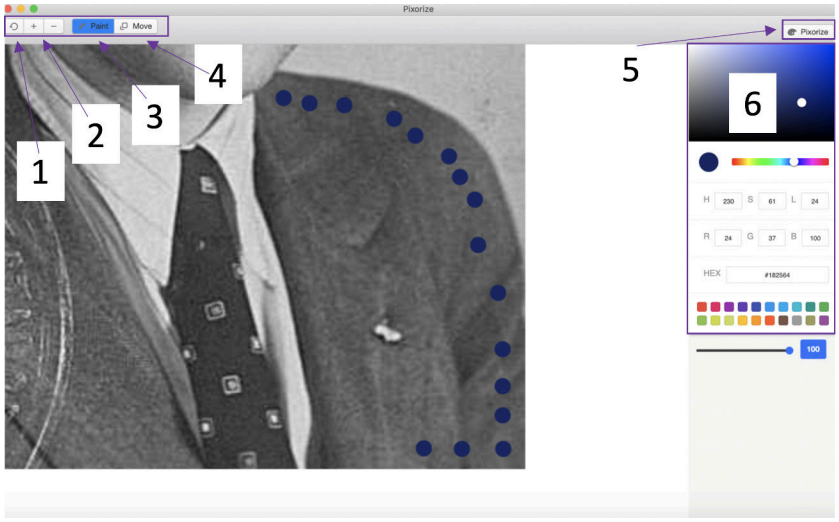


Abbildung 11: Bearbeitungsfenster

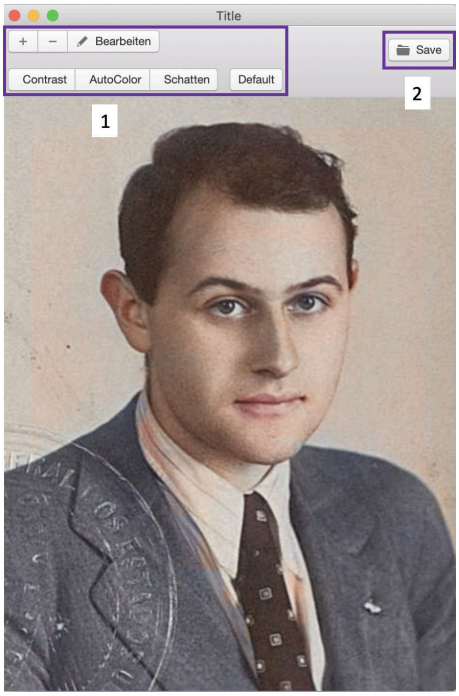


Abbildung 12: Präsentationsfenster

Foto anzeigen Nachdem das Foto mithilfe des künstlichen neuronalen Netzes koloriert wurde, erscheint das Präsentations-Fenster. Dieses Fenster beinhaltet noch zusätzliche Funktionen zur Nachbearbeitung des Fotos und bietet die Möglichkeit, das Foto zu speichern.

1. Die Benutzerin/Der Benutzer kann zwischen den Nachbearbeitungs-Algorithmen den Kontrast verbessern, die Farben und Schatten verbessern auswählen

2. Mittels der Save-Funktion kann die Benutzerin/der Benutzer sein Foto speichern und er erhält nach einem erfolgreichen Speichern eine Benachrichtigung

4.4 Aktueller Stand der Dinge

Trotz starker Hardwarelimitierung und anfänglichen Wissenslücken, war es uns möglich, eine enorm konkurrenzfähige Software für das Kolorieren von Schwarz-Weiß-Porträts zu erschaffen. Durch die Auswahl eines speziellen Farb-Modells, dem CieLAB-Farb-Modell, welches die Lichtintesität von der Farbe trennt, erhielten wir bei Vergleichen mit Konkurrenzprodukten, wie zum Beispiel ColouriseSG, weitaus bessere und foto-realistischere Resultate. [1] Da unser Algorithmus die ursprüngliche Lichtintensität nicht manipuliert, sondern nur die puren Farben erzeugt, ändert sich die natürliche Schattierung des originalen Bildes nicht.

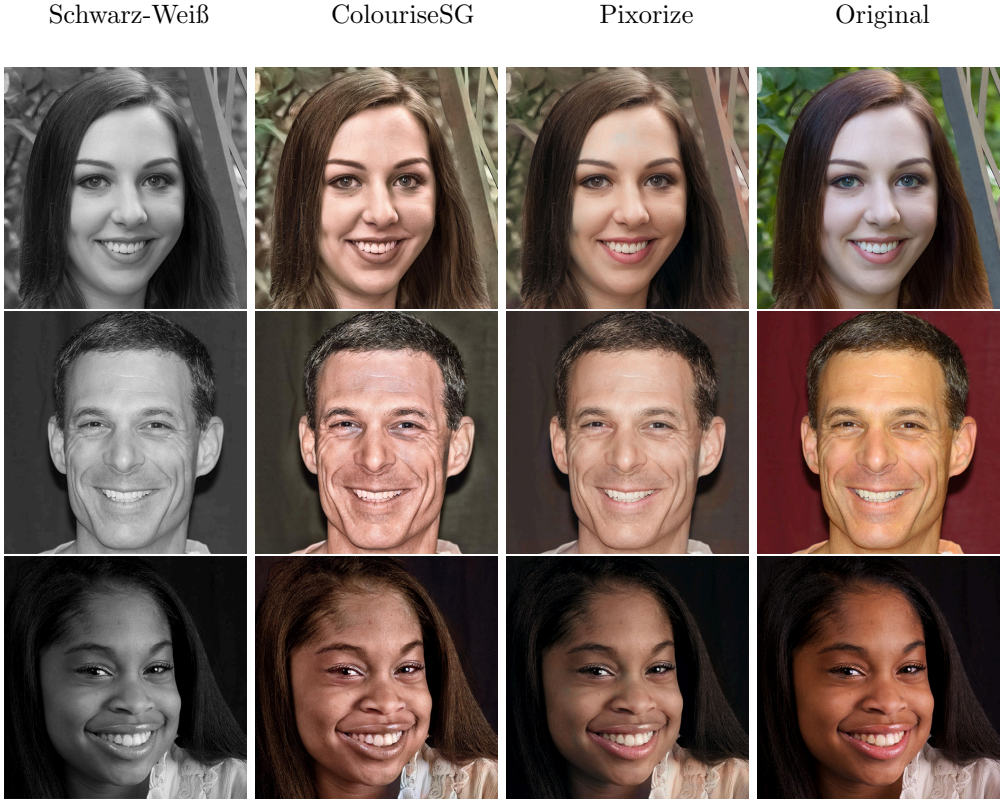


Abbildung 13: Vergleich zwischen der Konkurrenz (ColouriseSG) und Pixorize

Außerdem besitzt Pixorize zusätzlich zu der Funktion der vollautomatischen Kolorierung, auch die Möglichkeit, Farb-Punkte im Foto zu setzen, wodurch der Benutzer interaktiv und schnell Änderungen durchführen kann. Diese Interaktivität wird von keinem Konkurrenzprodukt angeboten und stellt neben der Überlegenheit in der Qualität eine einmalige innovative Funktion dar.
Jedoch ist interaktive Funktion zum aktuellen Stand der Dinge noch nicht fertig ausgereift. Setzt man zurzeit einen Farb-Punkt, wird eine lokale Befärbung durchgeführt. Das heißt, dass zurzeit noch keine ganzen Segmente mithilfe eines Farb-Punktes koloriert werden. Wenn man also ein großes Objekt einfärben möchte, müssen zurzeit mehrere Farb-Punkte in diesem Objekt gesetzt werden.

4.5 Ausblick

Die automatische Kolorierung funktioniert mithilfe des künstlichen neuronalen Netzes schon hervorragend und ist der Konkurrenz in der Domäne der Porträt-Fotos voraus, jedoch wollen wir die Innovation weiter vorantreiben. Der Algorithmus soll noch besser werden, indem wir unseren Datensatz vergrößern, beziehungsweise das künstliche neuronale Netz auf weitere Fotos trainieren, um so eine noch genauere und robustere Version zu entwickeln. Außerdem werden wir mit Mühe versuchen, die interaktive Befärbung zu verbessern und Fehler auszu-

merzen, um die bestmögliche User-Experience anzubieten.
Nun soll der Nutzen dieses Projektes der Gesellschaft nicht verwehrt bleiben und deshalb nehmen wir es uns als Ziel, Pixorize zum Download verfügbar zu machen. Somit soll es für Privatpersonen, als auch für professionelle Anbieter, möglich sein, Pixorize zu nutzen, um auf eine innovative Weise, Fotos schnell und komfortabel zu kolorieren. Außerdem soll der Code auf GitHub veröffentlicht werden, damit die wissenschaftliche Gemeinschaft eine detaillierte Einsicht hinter die Kulissen erhält. Zusätzlich zum Code soll aber auch eine ausführliche Beschreibung, einem Paper ähnlich, der Öffentlichkeit zur Verfügung stehen. Somit könnte das spezifische Wissen, welches bei Pixorize erlangt wurde, für andere Menschen zugänglich sein.

4.6 Bericht des Projektkoordinators

Die Teammitglieder waren schon vor der gemeinsamen Projektarbeit befreundet, was am Anfang des Projektes für Konflikte sorgte. Denn im Projekt wurde ein Projektkoordinator ernannt, welcher die Richtung vorgab und Aufgaben zuteilte und somit war es für die einzelnen Mitglieder erstmals schwer das Private vom Projekt zu trennen. Jedoch wurden diese Konflikte beseitigt und einer angenehmen Projektarbeit stand somit nichts mehr im Wege. Durch eine Ausbildung an einer HTL mit dem Schwerpunkt Informationstechnologie waren uns schon die Grundsteine für solch ein innovatives Projekt gelegt. Unterrichtsgegenstände wie Mathematik, Softwareentwicklung und Verteilte Systeme waren eine Grundvoraussetzung, ohne welche das Projekt niemals erfolgreich gewesen wäre. Zwar hatte nur der Projektkoordinator geringe Kenntnisse mit Machine Learning und Python, jedoch hatte jedes einzelne Mitglied unglaublich viel Motivation und Ehrgeiz, um das nötige Wissen zu erlangen. Durch Fleiß und Interesse konnten wir uns neues, komplexes Wissen aneignen und anschließend anwenden.

Manuel Riedl
: Ein Auge für modernes Design und ein Gespür für ein angenehme User-Experience war für Manuel Riedl ein Vorteil bei der Entwicklung des User-Interfaces

Stefan Etzelstorfer
: Eine logische und analytische Denkweise ermöglichte es Stefan Etzelstofer, komplexe Probleme zu analysieren und der Lösung einen Schritt näher zu kommen

Stefan Haas
: Führungskraft und Verantwortungsgefühl verleitete Stefan Haas zu der Rolle des Projektkoordinators, in welcher er das Team regelmäßig motivierte

4.6.1 Kommunikation

Innerhalb des Projektteams
: Natürlich wurde einerseits in der Schule kommuniziert, wo man Angesicht zu Angesicht Aufgaben koordinieren und auftretende Probleme kurz besprechen konnte, aber auch Treffen außerhalb der Schule wurden regelmäßig angesetzt. Meistens dienten solche Treffen dem Bestimmen der nächsten Arbeitspakete und der Hilfeleistung bei komplizierten Problemen. Zusätzlich wurde eine WhatsApp-Gruppe erstellt, welche der text-basierten Kommunikation diente.
Da wir ein starkes Augenmerk auf eine organisierte Kommunikation legten, wusste man immer über den Fortschritt seiner Kollegen Bescheid und konnte seine Hilfe bei Problemen anbieten.

Mit dem Projektbetreuer
: Am Ende jeder Woche wurde der Projektbetreuer per E-Mail benachrichtigt. Die E-Mail beinhaltete die erledigten Aufgaben und Hindernisse der letzten Woche, sowie einen detaillierten Plan für die Arbeitspakete, die in der darauffolgenden Woche bearbeitet werden.

Zusätzlich hat sich das Team mit dem Projektbetreuer nach Erreichen eines Meilensteins in der Schule getroffen. Dabei wurden die neuen Ergebnisse präsentiert und anschließend wurden die kommenden Schritte besprochen.

4.6.2 Arbeitsformen

Die einzelnen Funktionen und Komponenten wurden zu Beginn elementar aufgespaltet, somit war es uns möglich, diese Arbeitspakete an einzelne Personen zuzuweisen. Dadurch war es uns meistens möglich einzeln an Arbeitspaketen zu arbeiten, jedoch haben wir bei Problemen andere Mitglieder immer miteinbezogen, um somit gemeinsam die Fehlerursache herauszufinden. Dies geschah meistens nach dem Prinzip des Pair-Programmings, bei welchem man zu zweit vor einem Computer sitzt, eine Person programmiert, die andere analysiert. Diese Methode ist sehr effektiv, da man seine Denkweise immer der anderen Person erklären muss und dabei oftmals schon bemerkbar wird, ob etwas funktioniert, oder nicht.

4.6.3 Unerwartete Hindernisse

Eine der ersten Hindernisse war die Suche nach einem passenden Datensatzes, welches genügend Porträt-Fotos beinhaltet. Da wir eine innovative Bildkolorierung speziell für Familienfotos und Porträt-Fotos verfolgen und es in dieser Domäne so gut wie kein passender Datensatz existiert, mussten wir sehr viel Zeit investieren und das Internet durchforsten. Jedoch haben wir nach einer langen und sehr genauen Suche einen Datensatz gefunden, welcher unseren Anforderungen genüge tut.

Des Weiteren muss im Prozess des Kolorierens ein Bild mehrere Male in ein anderes Farb-Modell konvertiert werden. Dabei wurde die Bibliothek OpenCV verwendet, welche genau solche Konvertierungen ermöglicht, jedoch wurde bei Pixorize ein sehr spezielles Farb-Modell verwendet und die Umwandlung in dieses Farb-Modell war nicht ausreichend dokumentiert in der OpenCV-Dokumentation, wodurch die Umwandlung zu Beginn nicht möglich war. Jedoch haben wir einen eigenen mathematischen Algorithmus entwickelt, welcher eine Umwandlung doch noch ermöglichte.

Wie sich im Laufe der Zeit herausstelle, ist die zu Beginn gewählte Architektur des Neuronalen Netzes nicht in der Lage die benutzer-gesetzten Farb-Punkte auf ganze Flächen anzuwenden. Wird zum Beispiel ein Farb-Punkt auf den Himmel gesetzt, so wird nur ein kleiner rechteckiger Bereich koloriert.

4.6.4 Lösungshilfen

Da wir in der Schule keine Lehrkraft haben, welche ausgeprägte Kenntnisse im Bereich Machine Learning aufweist, waren wir zumeist auf uns gestellt. Jedoch griffen wir in Zweifelsfällen auf Internetforen, wie Stack Overflow, zu und andere Male mussten wir eben eine intensive Recherche betätigen.
Zusätzlich pflegte ich während des Projektes den Kontakt mit einem PhD-Studenten der JKU des Instituts für Machine Learning, welcher bei Fragen sehr hilfsbereit bereitstand. Diese Person durften wir auch einmal an der JKU besuchen, um Fragen vor Ort zu klären. Dabei wurde gemeinsam nach Ideen gesucht und Fehler wurden identifiziert und deren Ursprung hergeleitet. Durch die Expertise, welche uns von Seiten der JKU zuteilwurde, konnten wir Erfolge und Misserfolge bestätigen.

4.6.5 Konflikte

Anfänglich gab es mehrere Konflikte, wenn Ideen im Team präsentiert wurden. Denn manchmal wurden uniformiert Ideen aufgestellt, welche nicht umsetzbar waren und manchmal wollte man lieber seine eigenen Ideen integrieren. Somit war es nicht ungewöhnlich, dass Ideen nicht akzeptiert wurden und schließlich entstanden aus einhergehenden Unzufriedenheiten schlussendlich Konflikte.

Die Situation erforderte rasches Handeln und es musste dringend interveniert werden. Indem wir bei jeder Idee oder jedem Vorschlag, egal ob sinnvoll, oder sinnlos, eine konstruktive und zivilisierte Diskussion führten, kamen wir häufig zu einer gemeinsamen Einsicht. Dadurch fiel es uns zum einen viel einfacher Entscheidungen zu treffen und zum anderen wurden Diskrepanzen rasch eliminiert.

4.6.6 Soziale Lernprozesse

Da dieses Projekt ein ganzes Jahr in Anspruch nahm, mussten wir lernen, miteinander ein Ziel gemeinsam über einen langen Zeitraum zu erreichen. Dabei musste jeder, zu jederzeit motiviert sein und seinen Beitrag leisten, ansonsten wäre kein Erfolg in Aussicht gewesen. Man lernte also Zusammenhalt.
Außerdem mussten wir lernen, anderen Mitgliedern zu vertrauen. Wir mussten schließlich darauf verlassen, dass jeder ein komplexes Problem lösen kann und dass jeder bei Schwierigkeiten nicht aufgibt.
Eine weitere große Erkenntnis war, das Aussprechen von Kritik, als auch das Annehmen von Kritik, gekonnt sein muss. Denn in der heutigen Gesellschaft ist man leicht dazu verleitet Fehler bei anderen zu suchen und ist demnach schnell dazu verleitet in einem übertriebenen Maße Kritik auszuüben. Achtet man nicht darauf solch ein Verhalten zu unterlassen, so werden andere Personen diese übermäßige Kritik als persönliche Kritik auffassen, selbst wenn sich die Kritik ursprünglich nicht auf eine persönliche Ebene bezog. Aus diesem Grund ist es bei einer Projektarbeit, als auch im gänzlichen Leben, von Vorteil Kritik oftmals für sich zu behalten.

5 Resultate

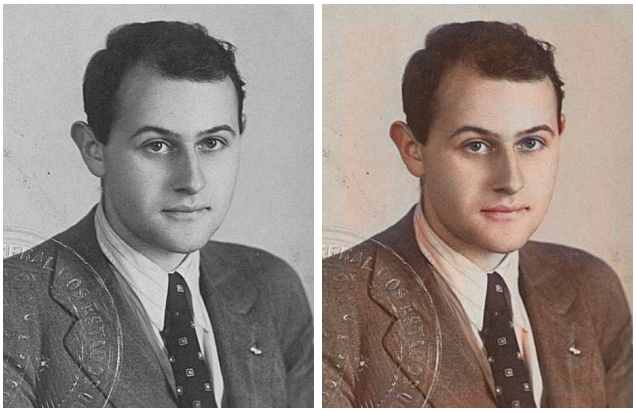


Abbildung 14: Vollautomatische Kolorierung

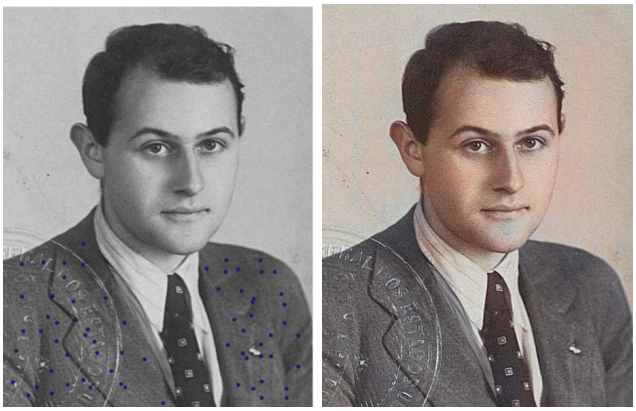


Abbildung 15: Benutzerinteraktive Kolorierung mit blauen Farb-Punkten

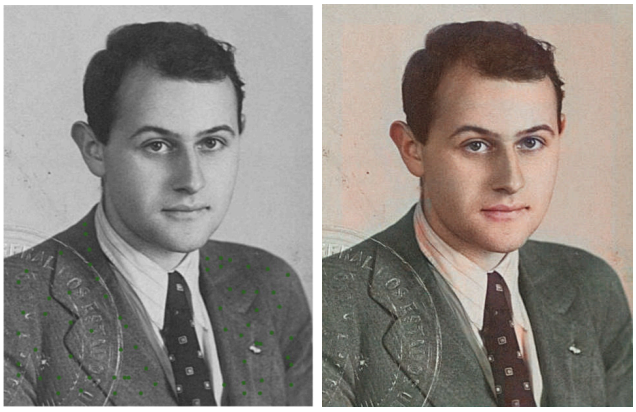


Abbildung 16: Benutzerinteraktive Kolorierung mit grünen Farb-Punkten

6 Literaturverzeichnis

[1] ColouriseSG. ColouriseSG. https://colourise.sg/. Eingesehen am 01.03.2020.

[2] matplotlib.org. Matplotlib. https://matplotlib.org/. Eingesehen am 26.02.2020.

[3] numpy.org. Numpy. https://numpy.org/. Eingesehen am 28.02.2020.

[4] OpenCV.org. Opencv. https://opencv.org/. Eingesehen am 28.02.2020.

[5] Shana Pearlman. Was ist Datenaufbereitung? https://de.talend.com/resources/what-is-data-preparation/. Eingesehen am 27.02.2020.

[6] Pillow.org. Pillow. https://pypi.org/project/Pillow/. Eingesehen am 28.02.2020.

[7] Pytorch.org. Pytorch. https://pytorch.org/. Eingesehen am 26.02.2020.

[8] Wikipedia. Agile Softwareentwicklung. https://de.wikipedia.org/wiki/Agile_Softwareentwicklung/. Eingesehen am 27.02.2020.

[9] Wikipedia. Cascading Style Sheets. https://de.wikipedia.org/wiki/Cascading_Style_Sheets. Eingesehen am 26.02.2020.

[10] Wikipedia. Data preparation. https://en.wikipedia.org/wiki/Data_preparation/. Eingesehen am 27.02.2020.

[11] Wikipedia. Debugging. https://en.wikipedia.org/wiki/Debugging. Eingesehen am 01.03.2020.

[12] Wikipedia. Grafische Benutzer Oberfläche. https://en.wikipedia.org/wiki/Graphical_user_interface. Eingesehen am 01.03.2020.

[13] Wikipedia. Hypertext Markup Language. https://de.wikipedia.org/wiki/Hypertext_Markup_Language. Eingesehen am 26.02.2020.

[14] Wikipedia. JavaScript. https://de.wikipedia.org/wiki/JavaScript. Eingesehen am 26.02.2020.

[15] Wikipedia. Künstliches neuronales Netz. https://en.wikipedia.org/wiki/Artificial_neural_network. Eingesehen am 01.03.2020.

[16] Wikipedia. Python. https://de.wikipedia.org/wiki/Python_(Programmiersprache). Eingesehen am 26.02.2020.

[17] Wikipedia. Scrum. https://de.wikipedia.org/wiki/Scrum/. Eingesehen am 27.02.2020.

[18] Wikipedia. Stackoverflow. https://de.wikipedia.org/wiki/Stack_Overflow_(Website)/. Eingesehen am 27.02.2020.

7 Bildverzeichnis

8 Tabellenverzeichnis