

Софийски университет "Св. Климент Охридски"
ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА
Специалност "Софтуерно инженерство"

Image Colorization

Оцветяване на черно-бели изображения



Изготвил: Даниел Халачев
Факултетен номер: 62547
Преподавател: проф. Иван Койчев
Дата: 26 януари 2024 г.

Декларация за липса на plagiatство

- Плагиатство е да използваш, идеи, мнение или работа на друг, като претендираш, че са твои. Това е форма на преписване.
- Тази курсова работа е моя, като всички изречения, илюстрации и програми от други хора са изрично цитирани.
- Тази курсова работа или нейна версия не са представени в друг университет или друга учебна институция.
- Разбирам, че, ако се установи plagiatство в работата ми, ще получа оценка “Слаб”.

Благодарности

Благодаря на колегата си Павел Атанасов, който ми предостави сървъра си, за да тренирам невронната мрежа за целите на този проект.

Даниел Иванов Халачев

Съдържание

1 Въведение	3
1.1 Мотивация	3
1.2 Дефиниция на проблема	3
2 Подходи и методи за оцветяване на черно-бели изображения	4
2.1 Критерии за избор на решение	4
2.2 Пространство на цветовете	4
2.2.1 Пространства, базирани на основни цветове	4
2.2.2 Пространства, базирани на характеристиките на цвета	4
2.2.3 Хибридни пространства	5
2.3 Невронни мрежи, подходящи за решение на проблема	5
2.3.1 Техники преди невронните мрежи	5
2.3.2 Convolutional Neural Network	5
2.3.3 Generative Adversarial Network	7
2.3.4 Transformer Neural Network	8
2.3.5 Други техники	8
2.4 Функция на загубата	9
2.4.1 L1 Loss (Mean absolute error)	9
2.4.2 L2 loss (Mean Squared Error)	9
2.4.3 Smooth L1 Loss	10
2.4.4 Perceptual Loss	10
2.4.5 Total Variance Loss	10
2.4.6 Adversarial Loss	10
2.5 Избор на база данни	10
2.6 Избор на метрика за качеството	11
2.6.1 Субективна оценка на качеството	11
2.6.2 Обективна оценка на качеството	11
3 Характеристики на избраното решение	12
3.1 CIELAB пространство от цветове	12
3.2 U-Net генератор	13
3.3 Patch Discriminator	14
3.4 Функция на загубата	15
3.5 Избор на база данни	15
3.6 Оценка на качеството	15
4 Реализация	15
5 Заключение	17

1 Въведение

1.1 Мотивация

Историята е основна хуманитарна наука от ключово значение за разбиране на света и събитията, довели до съвременното му състояние. Един от основните типове исторически извори са **историческите фотографии** на места, лица и събития, които, за съжаление, са предимно черно-бели. Но човешкото зрение е цветно и това не е случайно - чрез цветовете хората възприемат и взаимодействват със света по-добре.

Затова щеше да бъде полезно, ако историческите фотографии можеха да "оживеят" в истинските си цветове. За щастие това е напълно възможно днес с технологиите на дълбоките невронни мрежи.

1.2 Дефиниция на проблема

Оцветяване на изображения е техника, която използва алгоритми за дълбоко обучение, за да придае нов цвят на изображения, различен от оригиналния. Така дефиниран, проблемът е много по-общ от оцветяването на черно-бели изображения. В действителност са възможни други разнообразни сценарии, сред които оцветяване на:

- изображения от инфрачервени камери в естествените им цветове
- изображения от камери за нощно виждане в дневните им цветове
- необичайни предмети или фрактури в рентгенови изображения
- зони и терени в сателитни изображения
- и много други.



Фигура 1: Оцветяването на изображения с ИИ може да се използва в много области на човешкия труд

Този проект ще се придържа към темата за оцветяване на черно-бели изображения в цветни.

2 Подходи и методи за оцветяване на черно-бели изображения

Проблемът за оцветяване на черно-бели изображения отдавна е предмет на изследвания, които предлагат разнообразни техники и похвати за решението му.

2.1 Критерии за избор на решение

Решението на проблема не се състои в една единствена стъпка и изпълнението ѝ. Напротив, един мотивиран избор трябва да вземе предвид следните особености (Huang и др.):

- избор на пространство на цветовете (начин на представяне на цветовете)
- избор на архитектура за невронна мрежа
- избор на функция на загубата
- избор на база данни
- избор на техника за оценка на качеството

В следващите секции се коментират всички тези особености.

2.2 Пространство на цветовете

Естествените цветове в природата се представят обективно и възпроизведимо в цифров вид чрез различни пространства на цветовете (Gravesen), сред които най-известни са RGB, HSL(HSV), YCbCr, CMYK. Най-общо пространствата на цветовете попадат в една от **3 категории**:

2.2.1 Пространства, базирани на основни цветове

Тези пространства от цветове представляват цветовете чрез няколко основни. Най-популярният цветови модел - **RGB**, попада в тази категория, защото представя всички цветове и нюанси чрез три основни цвята - **червено, зелено и синьо**.

2.2.2 Пространства, базирани на характеристиките на цвета

Човешкото око се състои от два типа клетки - пръчици и колбички, които отговарят съответно за възприемането на светлината и цветовете, като пръчиците са значително повече и са отговорни за по-голямата част от зрителната информация.

Пространствата, базирани на характеристиките на цвета възникват като отговор на модели като RGB, които изобщо не вземат предвид светлината и яркостта като фактори за възприемане на цветовете. Затова те представляват цветовете чрез техни основни характеристики - например нюанс, насыщеност, яркост. Очевидно модели като **HSL** (Hue-Saturation-Lightness) попадат в тази категория. Сред тези пространства обаче има едно, което заслужава особено внимание - **CIELAB**.

CIELAB представя цветовете чрез три техни характеристики - осветеност (lightness, L), нюанси в спектъра на червеното и зеленото (означени с A), и нюанси в спектъра на жълтото и синьото (означени с B). Защо това пространство е важно, ще разберем в секцията Характеристики на избраното решение.

2.2.3 Хибридни пространства

Тези пространства използват за представянето на цветовете както характеристики на цвета, така и основни цветове. Примери за такива пространства са **RGB+YCbCr**, **RGB+HSV** и други. Те се отличават с по-голяма сложност от първите два типа и обикновено не се ползват за такива проблеми.

2.3 Невронни мрежи, подходящи за решение на проблема

Невронна мрежа е структура (мрежа) от взаимосвързани възли (неврони), подобна на структурата на човешкия мозък (Hopfield). Всяка невронна мрежа се състои от слоеве. За оцветяването на изображения се използват **дълбоки невронни мрежи**, т.е. невронни мрежи с повече от 3 слоя, като всеки слой се състои от множество взаимосвързани неврони.

2.3.1 Техники преди невронните мрежи

Невронните мрежи съществуват като концепция още от 60-те години, но изискват големи бази данни, ефективни алгоритми и достъпен хардуер за изпълнението им, с каквите разполагаме едва от скоро.

Именно поради тази причина решаването на проблема с невронни мрежи добива популярност едва след публикуването на изследването **Deep Colorization** (Cheng и др.). Преди това са използвани различни подходи, сред които:

- третирането на проблема като глобална оптимизационна задача (Levin и др.)
- използване на алгоритъма на Дейкстра за най-краткия път (Kawulok и Smolka) (Yatziv и Sapiro).

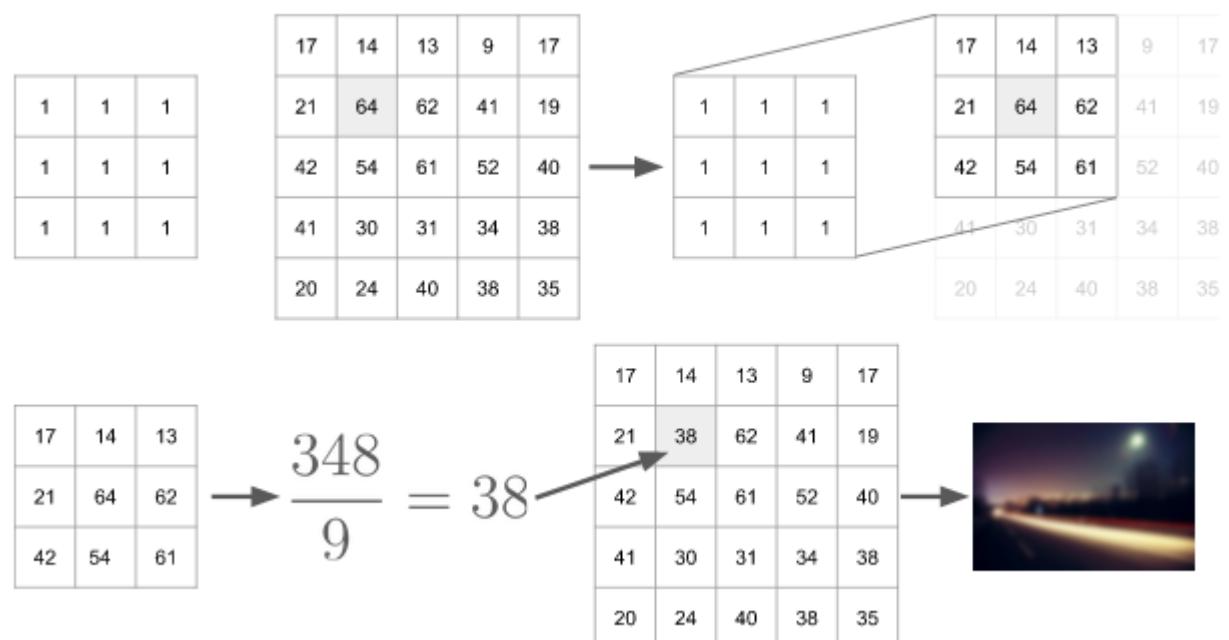
Всички тези алгоритми изискват сериозна обратна връзка от инженера и често дават неточни резултати (Zeger и Grgic). Именно поради тази причина невронните мрежи бързо се налагат, когато състоянието на областта най-накрая позволява това.

2.3.2 Convolutional Neural Network

Общо описание **Конволюционните невронни мрежи** са невронни мрежи, които се обучават сами в откриването и извлечането на характеристики от изображения (Venkatesan и Li). Това тяхно качество ги прави много използвани за разнообразни обработки на изображения. В конкретния случай за оцветяване на изображения, те се учат да асоциират обекти в изображенията с техните истински цветове. За целта, алгоритъмът първо разделя тренировъчните изображения на различни клъстери, след което обучава цветови модел за всяка група клъстери. Когато се подаде черно-бяло изображение, моделът първо търси най-близките клъстери според съдържанието и асоциира части от изображението с правилните

цветове (Huang и др.). За да се постигне това, тези мрежи биват обучавани с големи бази данни, за да дадат точни резултати.

Конволюция Основният способ за извличане на характеристиките е **конволюция**, откъдето идва и името на невронната мрежа. Конволюцията представлява прилагане на филтър над някой канал на изображението, в резултат на което се получава ново изображение, наречено **карта на характеристиките**, или **feature map**. Причината - новото изображение често подчертава някаква характеристика на оригиналното - линия, кръг, крива, цвят, обект или др.



Фигура 2: Прилагане на филтър



Фигура 3: Филтри и ефектите, които те прилагат над едно изображение. (Futurology)

Други видове слоеве на мрежата Освен конволюционни слоеве, този вид мрежа прилага още няколко вида слоеве с различна цел:

- **входен** (entrance) слой: приема черно-бялото изображение
- **конволюционен** (convolution) слой: произвежда карта на характеристиките
- **събирателен** (pooling) слой: намалява резолюцията на изображението (или канал от него), което позволява по-лесни изчисления и откриване на нови характеристики.
- **активационен** слой: конволюцията е линейна операция, която невинаги връща подходящи резултати. Затова се прилага и активационен слой, който използва нелинейна функция. Тя позволява активирането (тоест преминаването през) неврони, които не биха били достигнати в противен случай. Така се подобрява способността на модела да учи “нови неща”.
- **напълно свързани** (fully-connected) слоеве: приемат всички характеристики и класифицират (или оцветяват) изображението
- **изходен** (exit) слой: връща класификация, или в случая за оцветяване - оцветеното изображение.

Всяка дълбока конволюционна мрежа съдържа **един** входен и един изходен слой и **множество** дълбоки слоеве от останалите типове.

2.3.3 Generative Adversarial Network

Общо описание Генеративните състезателни мрежи са техника, която използва две невронни мрежи - генератор и дискриминатор (Wang и др.).

Генераторът, с помощта на вектор от произволни числа, създава нови изображения.

Дискриминаторът отговаря за категоризирането на подадено му изображение като истинско (взето от база данни) или фалшиво (създадено от генератора).

Двете невронни мрежи **се състезават** една с друга, подобно на играчите в minimax алгоритъма. Всяка мрежа използва обратна връзка от другата, за да подобри стратегията си и да “спечели играта”. Математически изразено, генеративните състезателни мрежи играят игра по функцията

$$\min_{\mu_G} \max_{\mu_D} L(\mu_G, \mu_D) := \mathbf{E}_{x \sim \mu_{\text{ref}}, y \sim \mu_D(x)} [\ln y] + \mathbf{E}_{x \sim \mu_G, y \sim \mu_D(x)} [\ln (1 - y)]$$

Видове GAN мрежи За генератора и дискриминатора могат да бъдат използвани различни типове невронни мрежи, което води до разнообразни комбинации, сред които най-известни са:

- **conditional GAN** - освен вектор от произволни числа, като вход на генератора се подава и допълнителна информация - клас или описание.
- **CycleGAN** - клас GAN мрежи, специализирани в транслирането на изображения чрез трениране по двойки (Zhu и др.).
- **StyleGAN3** - позволява по-голям контрол върху стила на изображенията, създадени от генератора чрез не един, а два източника на шум (Karras и др.).

Проблемът на обучението За разлика от други невронни мрежи, например конволюционните, генеративните състезателни мрежи се отличават с трудна имплементация и трудно обучение, поради наличието не на една, а на две мрежи, които **трябва да се тренират синхронизирано**.

Ако генераторът се обучава по-бързо от дискриминатора, дискриминаторът разпознава твърде малко снимки на генератора, при което генераторът не се обучава на нови стратегии, защото вече използваните се приемат.

Обратно, ако дискриминаторът се обучава по-бързо от генератора, той разпознава почти всички изображения на генератора. В резултат, дори напълно валидни стратегии за оцветяване биват “разпознати” от дискриминатора, и съответно отхвърлени като неуспешни от генератора.

2.3.4 Transformer Neural Network

Този популярен и успешен вид невронни мрежи използва условен **саморегресионен трансформатор**, който създава първоначално изображение с лоша резолюция и и неточно подбрани цветове, което с помощта на две невронни мрежи се усъвършенства.

2.3.5 Други техники

За оцветяването на изображения се използват и множество други невронни мрежи, които са разновидност на горе-изброените. Сред тях споменаване заслужават **CapsNet**, **PixelCNN**, **Vectorized CNN** и други.

2.4 Функция на загубата

Невронните мрежи се обучават и нагаждат към проблема сами, с помощта на малко на брой хиперпараметри. Но за целта е необходима **обективна метрика**, която да измерва временния успех по време на обучението, чрез който невронната мрежа може да измерва и оценява валидността на стратегията си. Тази метрика се нарича **функция на загубата**.

Обикновено една функция на загубата сравнява истинско цветно изображение със същото черно-бяло, но оцветено впоследствие от мрежата.



Фигура 4: Употреба на функцията на загубата

Следват няколко секции, посветени на най-популярните функции на загубата.

2.4.1 L1 Loss (Mean absolute error)

Функцията връща по модул сумата на разликите между матриците на резултата и истинското цветно изображение. В началото най-популярната метрика, днес тя е критикувана, защото не е достатъчно точна - изчислява само глобалната информация, но не и локалните дефекти, а освен това води до загуба на наситеност. Въпреки това, тя има своите предимства - все още е добра метрика за консистентност и намалява "замъгляването".

$$L_1(\hat{Y}, Y) = \sum_{i=1}^n |\hat{Y}_i - Y_i|$$

2.4.2 L2 loss (Mean Squared Error)

Функцията връща квадрата на разликите в матриците на резултата и истинското цветно изображение. Точно както хи-квадрат теста в статистиката, квадратът подобрява метриката, защото е по-лесен за изчисление и клони по-бързо към оптималното състояние. И тази метрика обаче има неточности, защото разглежда "глобалното" състояние - например често бърка глобалният фон на снимката.

$$L_2(\hat{Y}, Y) = \sum_{i=1}^n (\hat{Y}_i - Y_i)^2$$

2.4.3 Smooth L1 Loss

Функцията комбинира предимствата на L1 и L2.

$$L_\delta(\hat{Y}, Y) = \begin{cases} \frac{1}{2} (\hat{Y} - Y)^2, & |\hat{Y}_i - Y_i| \leq \delta \\ \delta |\hat{Y}_i - Y_i| - \frac{1}{2}\delta, & \text{в противен случай} \end{cases}$$

2.4.4 Perceptual Loss

Метриката включва оценка на загубата в съдържанието (feature loss) и на стила (style loss). Изчислението използва матрицата на Грам. Когато тази оценка се използва за оцветяване на снимки, обикновено се използва само оценката на съдържанието. Тази функция позволява точна оценка за семантичните разлики в резултата и истинското изображение.

$$\begin{aligned} L_f^{\phi,j}(\hat{Y} - Y) &= \frac{1}{C_j H_j W_j} \left\| \phi(\hat{Y}) - \phi(Y) \right\|_2^2 \\ G_j^\phi(X)_{c,c'} &= \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \phi_j(X)_{h,w,c} \phi_j(X)_{h,w,c'} \\ L_s^{\phi,j}(\hat{Y} - Y) &= \frac{1}{C_j H_j W_j} \left\| G_j^\phi(\hat{Y}) - G_j^\phi(Y) \right\|_F^2 \end{aligned}$$

2.4.5 Total Variance Loss

Функцията връща сумата от абсолютните разлики между съседни пиксели. Тази метрика намалява “шума” (мутацията на цветовете), позволявайки плавни преходи в цвета, каквото са типични за истинските фотографии.

$$L_{tv} = \sqrt{(y_{i+1,j} - y_{i,j})^2 + (y_{i,j+1} - y_{i,j})^2}$$

2.4.6 Adversarial Loss

Функция на загубата, специална за GAN мрежи. Базирана на ентропията между изображенията на генератора и дискриминатора.

2.5 Избор на база данни

Изборът на база данни е важен и зависи от областта, от която са изображенията. Проблемът за оцветяване на исторически изображения е по-специфичен, защото не съществуват (достатъчно) цветни изображения, върху които да се обучи моделът за обекти, които в съвременните цветни снимки не се срещат - например дървени тъкачни станове, специфични военни униформи, танкове и самолети (без защитни покрития или маскировка) и т.н. Защастие, такива случаи са рядко срещани и бази данни с изображения от днешното всекидневие могат да свършат достатъчно добра работа за оцветяването на пейзажи и лица, които са преобладаващи фигури дори в исторически изображения. Затова и бази данни, формирани с цел класификация, са подходящи и за оцветяване на черно-бели изображения.

В таблицата по-долу могат да разгледани най-машабните, разнообразни и популярни бази данни, които са съвместими с проблема:

Най-авторитетните бази данни за изображения на естествени сцени		
База данни	Съдържание	Размер
ImageNet	Изображения на 1000 класа обекти от всички области на човешкия живот	14 000 000
Pascal VOC	Изображения на 20 класа обекти от всички области на човешкия живот.	11 500
SUN	Изображения на 900+ сценария и 3900+ класа обекти от всички области на човешкия живот	131 000
Places205	Изображения на 205 сценария (основно места).	25 000 000
Places365	Изображения на 434 сценария (основно места).	10 000 000

2.6 Избор на метрика за качеството

След като бъде разработен и обучен един модел, трябва да бъде проверено **качеството** му, т.е. колко добре се справя с поставената му задача. За целта има два подхода.

2.6.1 Субективна оценка на качеството

Субективната оценка на качеството се прилага от човек, оценител. Пример за субективна оценка е **Mean Average Loss (MOS)**, при която оценителят дава оценка (например от 1 до 5) на партида от тестови изображения. Крайната оценка на модела се формира от средноаритметичното на всички поставени оценки.

Очевидно, този вид оценка страда от **субективност** - възможно е различни оценители да оценят едно и също изображение по различен начин. Освен това това оценяване изисква значителен **човешки труд и време**.

2.6.2 Обективна оценка на качеството

Обективната оценка на качеството разчита на функция, чрез която машина съпоставя на партида от тестови изображения число. Сред популярните метрики за обективна оценка на качеството са:

- **MSE (Mean Squared Error)**: сумата от квадратите на разликите между всеки два пиксела.

$$\text{MSE} = \frac{1}{mn} \sum_{i=0}^{n-1} \|T(i, j) - F(i, j)\|^2$$

- **PSNR (Peak Signal to Noise Ratio)**: измерва “изкривяването”, т.е. отклонението между две изображения.

$$\text{PSNR} = 10 \times \log_{10} \frac{\text{MAX}_I^2}{\text{MSE}}$$

- **SSIM (Structural Similarity Metric)**: оценява разликата в структурата на две изоб-

ражения въз основа на яркостта и контраста. Крайната стойност е в интервала [0, 1].

$$l(x, y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1}$$

$$l(x, y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2}$$

$$s(x, y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3}$$

$$c_1 = (k_1 L)^2, L := 255, k_1 := 0.01$$

$$c_2 = (k_2 L^2, L := 255, k_2 := 0.03$$

$$c_3 = \frac{c_2}{2}$$

$$\text{SSIM} = [l(x, y)]^\alpha \times [c(x, y)]^\beta \times [s(x, y)]^\gamma = \frac{(2\mu_x\mu_y + c_1)(\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}, \alpha = \beta = \gamma := 1$$

Обективната метрика на качеството не страда от недостатъците на субективната оценка, но поставената числена оценка е **неспособна** да опише или долови възприятията на човешкото око.

3 Характеристики на избраното решение

Обект на тази курсова работа е оцветяване на черно-бели изображения:

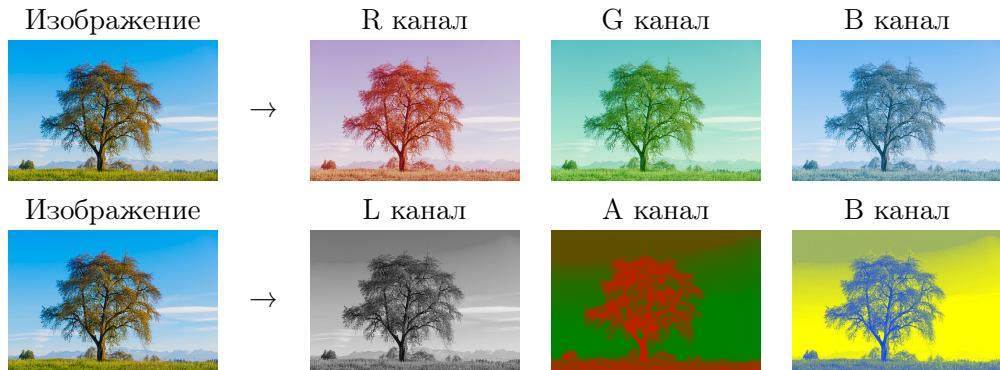
- чрез **CIELAB** пространство на цветовете
- чрез GAN невронна мрежа, състояща се от **U-Net генератор и Patch дискриминатор**, която ще позволи по-добри резултати с по-малка тренировъчна база данни.
- чрез **Adversarial Loss** функция на загубата за GAN мрежата и L1 функция на загубата за генератора.
- чрез подмножество на **ImageNet** базата данни от 50 000 изображения, от които на произволен принцип се избират **15 000** изображения за обучение и 1000 изображения за валидиране.
- Чрез субективна оценка - представяне на резултати от тренирането и валидирането на модела и посочване на успешни и неуспешни примери.

В секциите по-долу се коментират и обосновават по-подробно избрани техники.

3.1 CIELAB пространство от цветове

Внимателна съпоставка между RGB и CIELAB - най-често използваните пространства за този проблем, разкрива клучово **предимство на CIELAB** за черно-бели изображения пред RGB.

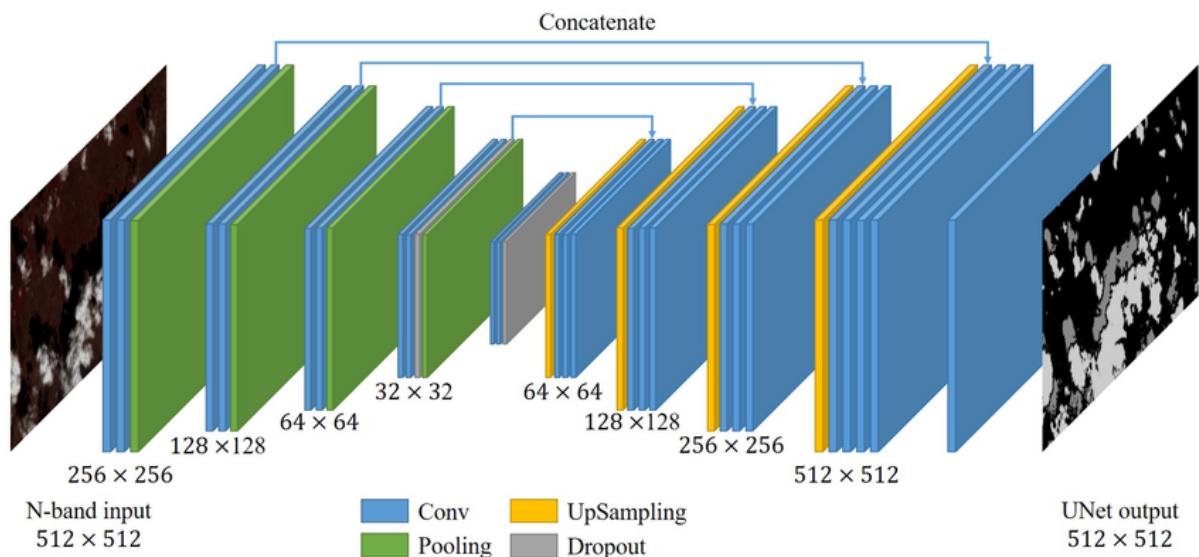
Всяко изображение се представя като **масив от пиксели**, където всеки пиксел е наредена n-торка от стойности. Съответно, всяко изображение може да бъде разделено на **канали**, които представляват масиви със само една стойност от n-торката. Следователно, и RGB, и CIELAB разделят изображението на три канала.



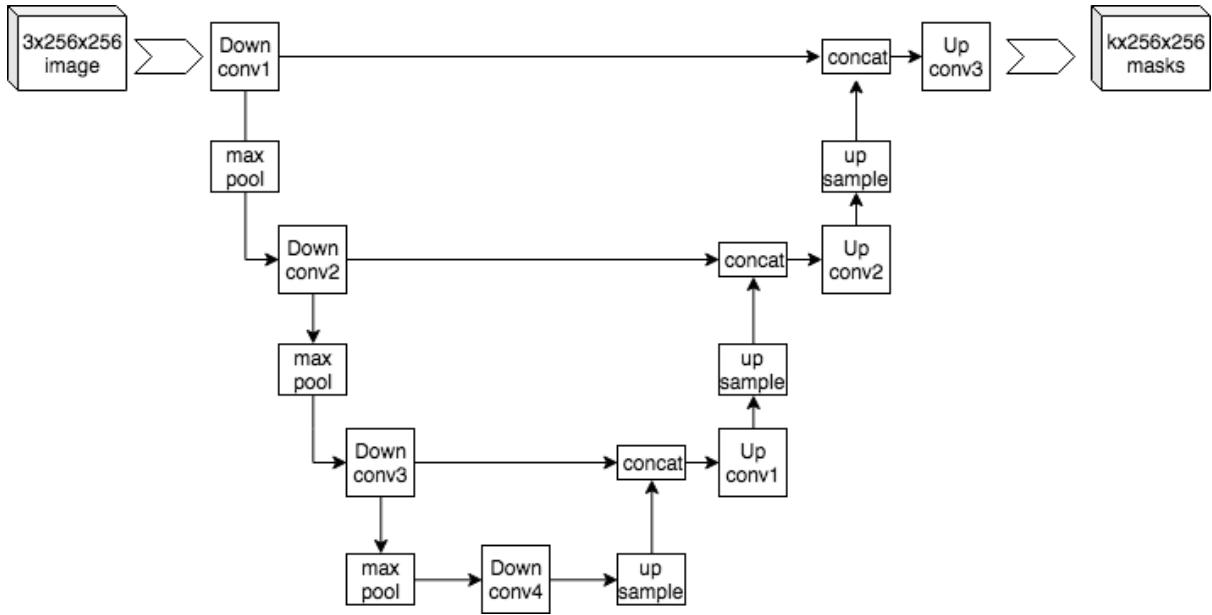
При RGB каналите са: червен, зелен и син цвят. При CIELAB каналите са: канал на осветеността, на червения и зеления цвят и на синия и жълтия. Лесно се забелязва, че **L каналът на CIELAB е на практика черно-бялото изображение**, което се подава на входа на невронната мрежа. Следователно, ако се използва CIELAB, е необходимо да се изчислят само две стойности за всеки пиксел, а не 3. Според модела CIELAB, а и в каналите имат неограничен брой стойности, но в софтуера те се свиват до броя 255 всяка. Следователно, вместо 265^3 се изчисляват само 265^2 стойности.

3.2 U-Net генератор

За архитектура на генератора е избрана **U-Net** мрежа, която дава отлични резултати за кратко време. На практика, U-Net е конволюционна невронна мрежа, която се състои от две части - свиваща се част (**енкодер**) и разширяваща се част (**декодер**). Формата на мрежата наподобява буквата U, откъдето идва и името ѝ.



Фигура 5: 3D илюстрация на U-Net



Фигура 6: Теоретичен модел на U-Net

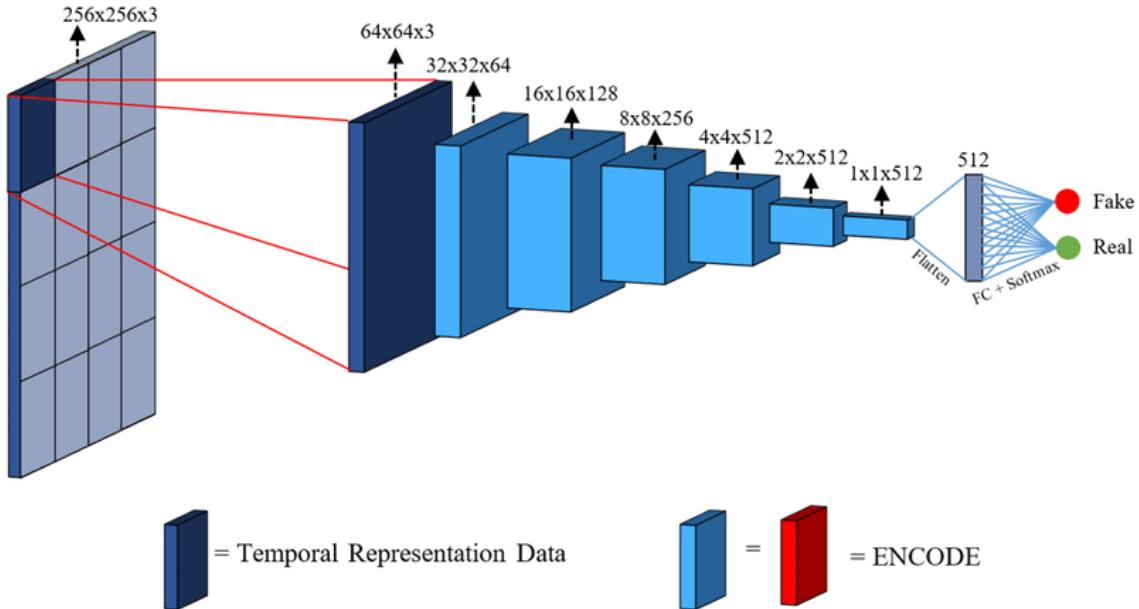
За разлика от традиционните невронни мрежи, тук липсват напълно-свързани слоеве.

Те са заместени от **skip connections** - връзка между два симетрични слоя от енкодера и декодера. Те позволяват комбинирането на първоначална версия на изображението, което се отличава с висока резолюция (с богато съдържание), с цветови пластове. Получава се ефект, който наподобява оцветяването на илюстрации от книжка за оцветяване.

Друга разлика е наличието на **dropout пласт** в най-вътрешния слой. Предназначенето му е да прекъсне част от връзките между невроните, създадени до момента, с цел предотвратяване на overfitting (прекомерно нагаждане към тренировъчните данни, което срива модела при невижданите до момента входни данни). Решението кои връзки да бъдат изтрити се взима произволно с вероятност p .

3.3 Patch Discriminator

За разлика от традиционните дискриминатори, които оценяват изображението в неговата цялост, **patch дискриминаторът** разделя изображението на части (**patches**), които оценява поотделно с цел по-голяма точност в разпознаването на изображения, подадени от генератора. В конкретния случай, това е 3-блокова конволюционна мрежа с Conv-BatchNorm и LeakyReLU слоеве.



Фигура 7: 3D илюстрация на Patch Discriminator

3.4 Функция на загубата

Използването на GAN невронна мрежа налага използването на **Adversarial Loss** функция на загубата. За загубата на генератора, ще бъде използвана функцията **L1 Loss**.

3.5 Избор на база данни

От всички възможни бази данни, **ImageNet** се отличава с най-голям брой класове различни обекти спрямо размера си. Това е важно, за да се осигурят обучителни изображения за възможно най-много обекти. Поради ограничения в хардуера за изпълнение, ще бъде избрано малко подмножество от ImageNet от **50 000** изображения. От тях, на произволен принцип ще бъдат избрани **15 000** за обучение чрез предварително определен константен seed, с цел постигане на детерминизъм (повтаряемост) при обучението.

3.6 Оценка на качеството

Секцията Заключение съдържа коментар с успехите и недостатъците на модела.

4 Реализация

Алгоритъмът е имплементиран на **Python** с помощта на библиотеката **PyTorch**, която предлага готови имплементации на дълбоки невронни мрежи или части от тях. Реализацията е базирана основно на статията *Image Colorization with U-Net and GAN* (Shariatnia), която от своя страна е основана на научното изследване *Colorful Image Colorization* (Zhang и др.). Заедно с този документ е приложен и използваният програмен код. Папката **Legacy** съдържа първоначалният опит за решение, който беше неефективен и преосmisлен. Папката **Image Colorizer** съдържа окончателния **Python** код, с който е имплементирана мрежата:

```

ImageColorizer
├── data                      # папката с тренировъчните изображения
├── dataset.py                # класове и функции за зареждане на данните за трениране
├── loss.py                   # класове за измерване на загубата
├── models.py                 # клоове, имплементиращи U-Net и Patch Discriminator
├── requirements.txt          # библиотеки и зависимости за автоматична конфигурация
├── testing.py                # скрипт за тестване с изображение на трениран модел
├── train.py                  # скрипт за трениране на модел
└── utils.py                  # помощни функции

```

Данните се зареждат в невронната мрежа чрез `Dataset` и `DataLoader` от библиотеката `pytorch`. Структурата на генератора и дискриминатора се изгражда като `nn.Module` чрез `torch.nn` и компонентите `nn.Conv2D`(конволюционен слой), `nn.LeakyReLU` и `nn.ReLU`(активиращи функции) и `nn.BatchNorm2d`(нормализация). Оптимизацията на невронната мрежа по време на обучение се осъществява чрез `torch.optim.Adam`. Приложението заедно с документацията код съдържа документация с подробно описание на параметрите и предназначението на функциите и класовете. Хиперпараметрите на модела могат да бъдат обобщени в следната таблица:

Параметър	Стойност	Описание
<code>lr_generator</code>	<code>2e-4</code>	Степента на обучение на генератора. Тя определя размера на стъпката за актуализиране на параметрите на генератора и влияе върху сходимостта и стабилността на обучителния процес.
<code>lr_discriminator</code>	<code>2e-4</code>	Степента на обучение на дискриминатора. Този параметър контролира размера на стъпката за актуализиране на параметрите на дискриминатора и също влияе на сходимостта и стабилността на обучението.
<code>beta1</code>	0.5	Параметърът <code>beta1</code> за оптимизатора Adam, който контролира градиентното спускане в първия момент на градиента. По-малка стойност дава по-малко тегло на по-старите градиенти.
<code>beta2</code>	0.999	Параметърът <code>beta2</code> за оптимизатора Adam, който контролира градиентното спускане във втория момент на градиента. По-малка стойност дава по-малко тегло на по-старите квадратни градиенти и влияе на адаптацията на скоростите на учене.
<code>lambda_l1</code>	100.0	Параметърът за тегло за L1 загубата в обективната функция на генератора. По-висока стойност акцентира върху верността на цветовете в генерираните изображения по време на обучението.

Таблица 1: Описания на хиперпараметрите

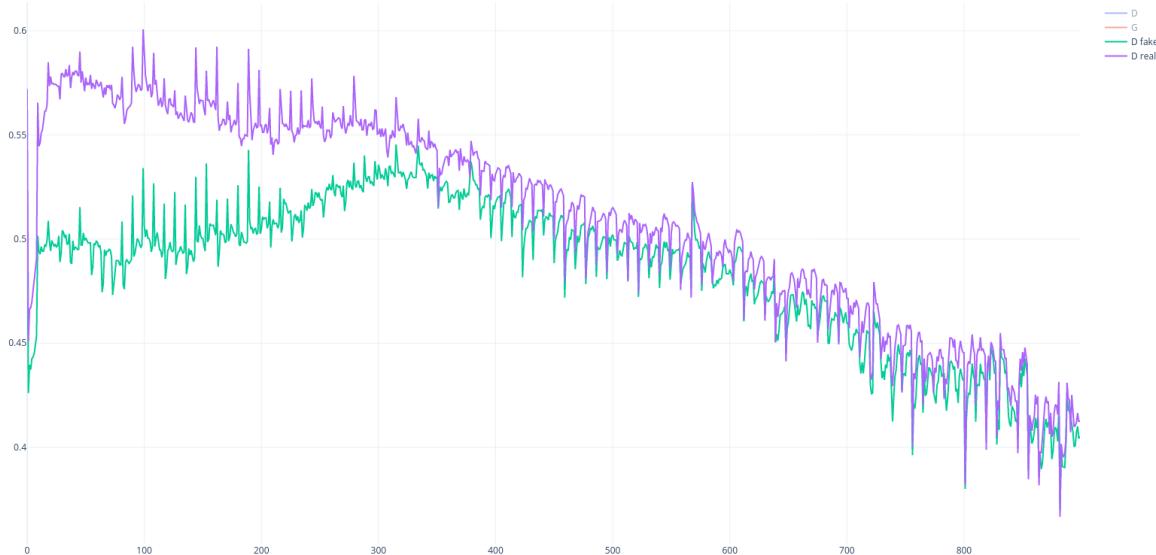
5 Заключение

Разработеният модел премина обучение на следния хардуер, до който временно получих достъп за целите на проекта:

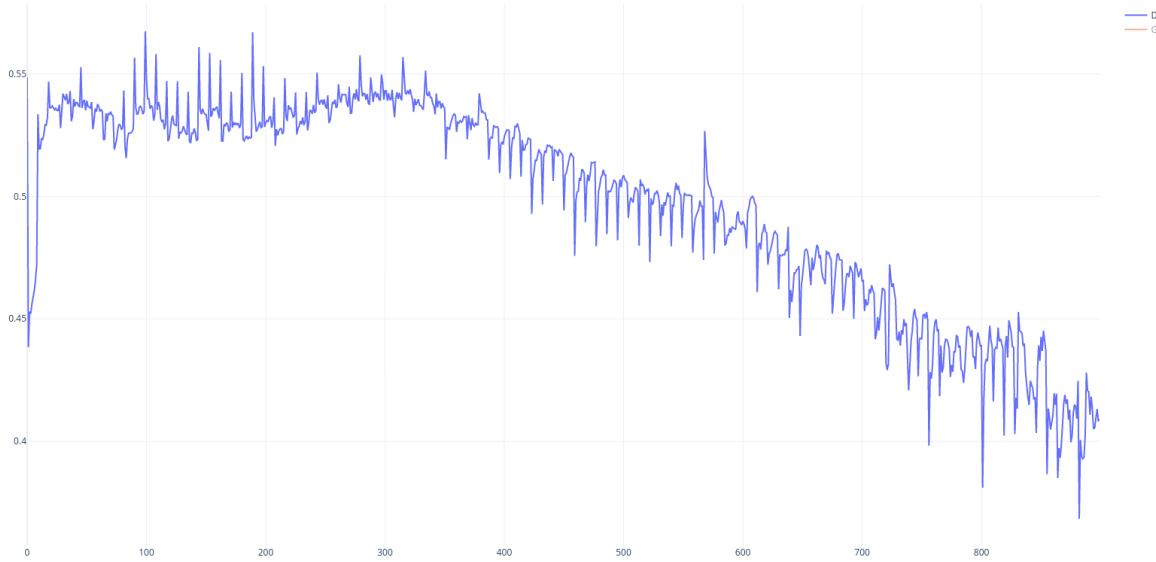
Host: Microstar Pro X670-P
CPU: AMD Ryzen 9 7950X @5.881GHz
GPU: AMD ATI Radeon RX 7700 XT / 7800 XT
GPU: AMD ATI Raphael

На графиките по-долу може да бъде проследено изменението на стойностите на функциите на загубата.

Сравнение между загубата на дискриминатора по отношение на истински и генерирали изображения показва относително равновесие между стойностите, което е нормално. Средна загуба от порядъка на 0.5 е очаквана и в границите на нормалното. Следователно няма изненада и в крайните стойности на загубата, които са средно аритметично от загубата при истински и генерирали изображения. Има ясна тенденция на намаляване на загубата на дискриминатора, но реалното изменение е минимално по стойност и с временен характер. Очакваше се по-голяма дисперсия в стойностите, но това може да бъде обяснено със загубата на генератора.

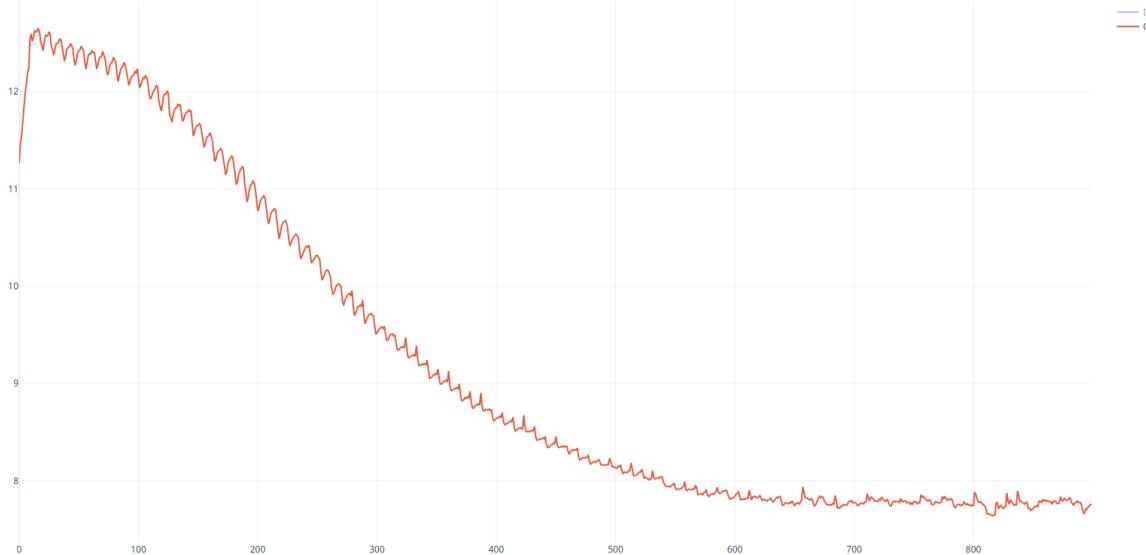


Фигура 8: Загуба на дискриминатора при истински и генерирали изображения D Real Loss & D Fake Loss



Фигура 9: Загуба на дискриминатора D Loss

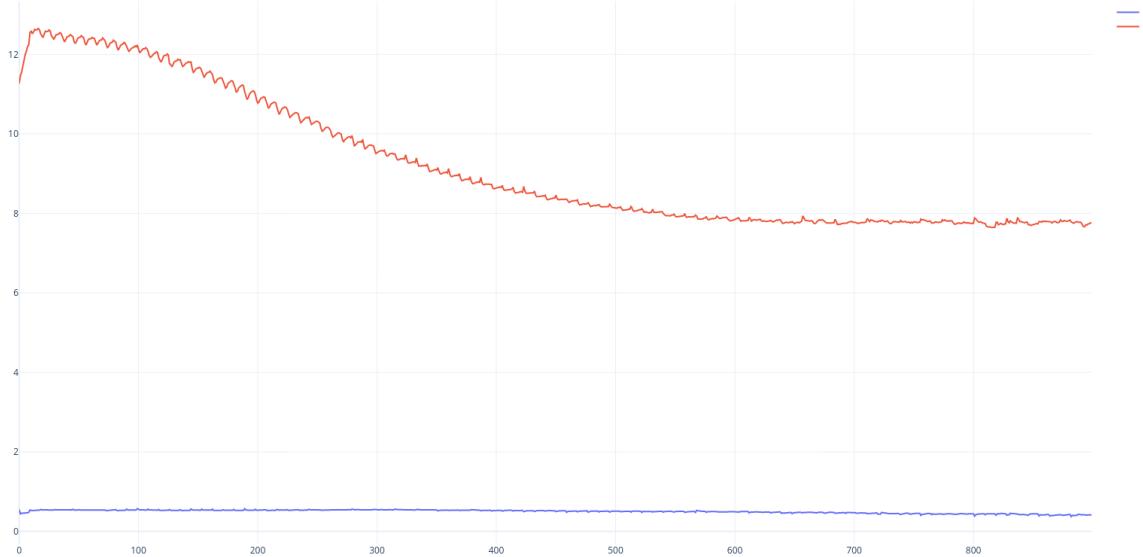
Изненадващи са резултатите в загубата на генератора. Наблюдават се типичните малки скокове и спадове, отразяващи промяна в стратегията на генератора на всяка итерация. Тенденцията към намаляване на загубата също е благоприятна. В последната епоха (итерации 800-900) се наблюдава относително стабилизиране на стойностите, но такива периоди са очаквани и почти винаги биват следвани от промяна в динамиката при следващи епохи.



Фигура 10: Загуба на генератора G Loss

За съжаление, средните стойности на загубата не са желани. Загубата от средно 10

единици на генератора ярко контрастира на загубата на дискриминатора от не повече от 1 единица. Тези различия могат да бъдат обяснени с разликата в трудностите на двете задания - разпознаването на истинността на едно изображение е значително по-лесно от създаването на реалистично такова. В допълнение, ограничението в броя тренировъчни изображения със сигурност е основната причина за по-слабото представяне на генератора.



Фигура 11: Загуба на генератора и дискриминатора съпоставени

За щастие, резултатите не са изцяло разочароваващи. В папката stages могат да бъдат разгледани примери от обучението през различните етапи. Сред тях се наблюдават не само забавни опити, но и някои сполучливи изображения.



Фигура 12: Временни резултати по време на тренирането, епоха 31

При валидацията на модела за човек е очевидно, че изображението е било черно-бяло и оцветено впоследствие, но и множество обекти са оцветени в подходящи цветове.

В заключение, хардуерните ограничения бяха основна пречка пред тренирането на модела и наложиха компромиси, които доведоха до по-лоши резултати от теоретично възможните. Но, с оглед на ограничения брой тренировъчни изображения и ограничения брой епохи за трениране, постигнатите резултати могат да бъдат считани за успех, защото моделът категорично не е безполезен и постига много спрямо инвестираното в него време за трениране. В това отношение, тази курсова работа не постига убедителни резултати, способни за заблудят човек в истинността на изображенията, но въпреки това успешно внася цвят в черно-бели изображения и демонстрира потенциала на избраната архитектура да реши поставения проблем, което беше и основната идея на проекта. След като с едва 15 000 изображения и 100 епохи бяха постигнати видими резултати, без съмнение тренирането на модела с цялата база данни ImageNet (14 000 000 изображения) за 200-300 епохи би довело до забележителни резултати.

Литература

- Cheng, Zezhou, и др. “Deep Colorization”. *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, дек. 2015, <https://doi.org/10.1109/iccv.2015.55>.
- Futurology. “Convolutional Neural Network Explained (CNN Visualised)”, дек. 2020, www.youtube.com/watch?v=pj9-rr1wDhM.
- Gravesen, Jens. “The metric of colour space”. *Graphical Models*, том 82, ноем. 2015, страници 77—86. <https://doi.org/10.1016/j.gmod.2015.06.005>.
- Hopfield, J J. “Neural networks and physical systems with emergent collective computational abilities.” *Proceedings of the National Academy of Sciences*, том 79, номер 8, апр. 1982, страницы 2554—58. <https://doi.org/10.1073/pnas.79.8.2554>.
- Huang, Shanshan, и др. “Deep learning for image colorization: Current and future prospects”. *Engineering Applications of Artificial Intelligence*, том 114, септ. 2022, страница 105006. <https://doi.org/10.1016/j.engappai.2022.105006>.
- Karras, Tero, и др. “A Style-Based Generator Architecture for Generative Adversarial Networks”. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, юни 2019, <https://doi.org/10.1109/cvpr.2019.00453>.
- Kawulok, Michal, и Bogdan Smolka. “Competitive image colorization”. *2010 IEEE International Conference on Image Processing*. IEEE, септ. 2010, <https://doi.org/10.1109/icip.2010.5653544>.
- Levin, Anat, и др. “Colorization using optimization”. *ACM Transactions on Graphics*, том 23, номер 3, авг. 2004, страницы 689—94. <https://doi.org/10.1145/1015706.1015780>.
- Shariatnia, Moein. “Image Colorization With U-Net and GAN”, 2020, colab.research.google.com/github/moein-shariatnia/Deep-Learning/blob/main/Image%20Colorization%20Tutorial/Image%20Colorization%20with%20U-Net%20and%20GAN%20Tutorial.ipynb.
- Venkatesan, Ragav, и Baoxin Li. “Convolutional Neural Networks”. *Convolutional Neural Networks in Visual Computing*, CRC P, окт. 2017, страницы 89—116, <https://doi.org/10.4324/9781315154282-4>.
- Wang, Zhengwei, и др. “Generative Adversarial Networks in Computer Vision: A Survey and Taxonomy”. *ACM Computing Surveys*, том 54, номер 2, февр. 2021, страницы 1—38. <https://doi.org/10.1145/3439723>.
- Yatziv, L., и G. Sapiro. “Fast image and video colorization using chrominance blending”. *IEEE Transactions on Image Processing*, том 15, номер 5, май 2006, страницы 1120—29. <https://doi.org/10.1109/tip.2005.864231>.
- Zeger, Ivana, и Sonja Grgic. “An Overview of Grayscale Image Colorization Methods”. *2020 International Symposium ELMAR*. IEEE, септ. 2020, <https://doi.org/10.1109/elmar49956.2020.9219019>.
- Zhang, Richard, и др. “Colorful Image Colorization”. *Computer Vision – ECCV 2016*, Springer International Publishing, 2016, страницы 649—66, https://doi.org/10.1007/978-3-319-46487-9_40.
- Zhu, Jun-Yan, и др. “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks”. *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, окт. 2017, <https://doi.org/10.1109/iccv.2017.244>.