SAPIENTIA ERDÉLYI MAGYAR TUDOMÁNYEGYETEM
MŰSZAKI ÉS HUMÁNTUDOMÁNYOK KAR,
MAROSVÁSÁRHELY
SZOFTVERFEJLESZTÉS SZAK

Párhuzamos képstílus átruházás konvolúciós neuronhálokkal

MESTERI DISSZERTÁCIÓ

TÉMAVEZETŐ:
dr. Iclănzan Dávid
Egyetemi tanár

SZERZŐ:
Szilágyi Ervin

2017 Július

UNIVERSITATEA SAPIENTIA TÂRGU-MUREŞ
FACULTATEA DE ŞTIINŢE TEHNICE ŞI UMANISTE
SPECIALIZAREA DEZVOLTARE DE SOFTWARE

DeepArt

Lucrare de master

Coordonator ştiinţific:
dr. Iclănzan Dávid

Absolvent:
Szilágyi Ervin

2017 Iulie

SAPIENTIA UNIVERSITY TÂRGU MUREŞ
FACULTY OF TECHNICAL AND HUMAN SCIENCES
SOFTWARE DEVELOPMENT SPECIALIZATION

Parallel artistic style transfer using deep convolutional neural networks

Master Thesis

Advisor:
dr. Iclănzan Dávid

Student:
Szilágyi Ervin

2017 July

eredetisegi nyilatkozat

KIVONAT

kivonat

Szilagyi Ervin,

.....

ABSTRACT

abstract

Szilagyi Ervin,

.....

ABSTRACT

english abstract

Szilagyi Ervin,

.....

Tartalomjegyzék

Cuprins

Table Of Contents

1. fejezet

Bevezető

Napjainkban a képfeldolgozás egy eléggé elterjedt kutatási terület. A kutatások célja főleg az információ kinyerésére, gépi látás kivitelézésére irányult. Minderre kiváló megoldást jelentett a deep konvolúciós hálok (ConvNets)[1][2] sikeres használata növelve ezzel ezek népszerűségét. Fontos megjegyezni, hogy a konvolúciós neuron hálok felfedezése már pár évtizede történt, tehát maga a technológia már régebb ismert volt. Az újrafelfedezésüket és hirtelen népszerűség növekedését annak köszönhetik, hogy az utóbbi években olyan hardveres megoldások jelentek meg amik lehetővé teszik az ilyen típusú hálok létrehozását és működtetését.

Az Nvidia cég 2007-ben bevezette az Nvidia CUDA platformot[3]. Ez egy komoly, használható fejlesztőkörnyezetett jelentett olyan fejlesztők számára akik nagy méretű adatpárhuzamos algoritmusokat szerettek volna fejleszteni. A CUDA környezet direkt elérhetőséget nyújt a videókártya utasításkészletéhez megengedve ezzel ennek a programozását. Ugyanakkor számos olyan videókártya került piacra ami egyre komolyabb számítási készségekkel bírt. Ezt a lehetőséget értelemszerűen a kutatók ki is használták így számos újabb publikáció és javaslat jelent meg amik neuron hálókat használnak az illető probléma megoldására.

A deep konvolúciós hálok népszerűségének növekedésével egyre több olyan fejlesztői környezet jelent meg amiknek célja a mesterséges intelligencia feladatok megoldása. Ilyen könyvtárak például a Caffe[4], Keras[5], Theano[6], Tensorflow[7], Torch[8] stb. Ezek a környezetekben, habár különböző stílusban de egyazon problémákra hivatottak gyors és egyszerű megoldásokat ajánla ugyanúgy mezei szoftverfejlesztők, mint kutatók számára.

Az gépi látás egyik fontos alkalmazási területe a képen levő tárgyak, élőlények emberek felismerése. Ilyen területen a konvolúciós hálok kimagasló teljesítményt nyújtanak, olyannyira, hogy egyes kisérletek szerint ez már nemhogy az emberi látással megegyező, hanem azt felülmúló teljesítméynt nyújtanak[9]. Feltevődik a kérdés, hogyha ennyire szofisztikált a gépi látás, akkor nem-e lehetne használni arra, hogy új képeket alkosson. Amint kiderült erre is alkalmasak. Az általam bemutandó dolgozat is ezt a témát próbálja megcélozni. A gépi látás a tárgyak, élőlények mellett képes felismerni maga a kép

művészeti stílusát. Ez elsősorban kihasználható arra, hogy híres művészek alkotásait csoportosítsuk, rendszerezzül[10], de amint e dolgozatból ki fog derülni, ki lehet használni arra is, hogy egy művészeti stílust egy adott festményről átvigyük egy mindennapi képre, fotóra.

A dolgozatom célja magyar híres festőművészek festészeti stílusát átvenni és ezt alkalmazni midennapi képekre illetve mozgóképekre. Eddigiekben, ahhoz hogy egy mindennapi fényképből művészeti képet varázsoljunk, képszerkesztő szoftverek segítségével lehetett elérni manuálisan. Mindezt egy olyan egyén végezhette, akinek képszerkesztési illetve képmanipulálási szakismere volt adott képszerkesztési szoftverkörnyezetben. Magától értődik az, hogy ez mozgóképek esetében egy időigényes folyamat. Dolgozatom mindezekre megoldást próbál adni, azáltal, hogy az általam elkészített szoftvert bárki használhatja, nincs szükség különböző képszerkesztői szakértelemre, emellett a folyamat ideje jelentősen csökkenni fog.

2. fejezet

Hasonló rendszerek feltérképezése

A neuron hálok használata a számítástechnikában nem egy újonnan kialakult terület. Frank Rosenblatt 1958-ban publikált egy olyan mintafelismerő algoritmust[11], ami egyszerű összeadást és kivonást használva képes volt "tanulni". A rendszer képes volt finomhangolni állapotát a bekövetkező iterációk során. Ezt az alogritmust perceptronnak nevezzük. 1975-ben Paul Werbos bevezette a backpropagation algoritmust[12], amit a perceptronnal együtt használva megoldotta a perceptron azon problémáját miszerint az csak lineárisan elválasztható osztályokat volt képes kategorizálni. Habár a neuron hálok tanulmányozása eléggé igéretesnek látszott, számítási igényük, komplexitásuk és lassú válaszidejük miatt a kutatók arra következtetésre jutottak, hogy a gyakorlatban még nem lehet alkalmazni őket.

Yann LeCun professzor és csapata 1998-ban egy újabb topologiájú hálot vezetett be[13]. A LeNet-5 elnevezésű háló konvolúciós rétegeket is tartalmazott ezért konvolúciós neuron hálonak nevezzük. A publikáció célja kézzel írott számjegyek kategorizálása volt, létrehozva ezáltal a MNIST adatbázist, ami 60000 28x28-as felbontású kézzel írott számjegyet tartalmaz, emellett tartalmaz egy 10000 tagból álló teszthalmazt. A dolgozatban bemutatott LeNet-5 háló 0,7%-os hiba aránnyal volt képes kategorizálni a számjegyeket, ami messze felülmúlta a többrétegű perceptronos megoldást.

Dave Steinkraus, Patrice Simard és Ian Buck 2005-ben publikált dolgozata[14] letette az alapjait a neuronhálók videokártyán történő programozásának. A videókártyán történő adatpárhuzamos programozás hatalmas performancia növekedést jelentett a processzoron futó neuronhálókkal szemben. Előtérbe kerül a deep learning és a mély konvolúciós hálok használata[1][2].

Eddigiekben sikerült nagyon pontos felismerő illetve osztályozó rendszereket alkotni. A mély konvolúciós hálok használata azonban nem merül ki ennyiben. 2015-ben publikálásra került egy olyan deep learning-et használó algoritmus, ami képes képes illetve festmények művészeti stílusát átvinni egy másik digitális képre[15]. Mostani dolgozatom is erre a publikációra alapoz, az ebben bemutatott módszereket próbálja alkalmazni illetve továbbfejleszteni. A tanuláshoz egy korábban bevezetett és gépi látáshoz használt, előre

betanított neuron hálót használnak fel, a VGG-19-et. Yaroslav Nikulin és Roman Novak tudományos kutatása[16] ezzel szemben eddig ugyanezt a módszert alkalmazta más ismertebb előre betanított hálókra, mint például AlexNet, GoogLeNet vagy VGG-16. Ugyanúgy a VGG-16 háló használata is kíváló eredményeket mutatott míg a GoogLeNet és az AlexNet architektúrájuk miatt komolyabb információvesztéshez vezetnek így a végeredmény nem lesz annyira látványos. Ugyanúgy kísérletek irányultak az eredi eljárás optimalizálására, megjelentek olyan rendszerek amik sajátos, erre a célre betanított neuron hálókat alkalmaznak[17][18][19].

2016-ban a Prisma labs inc. kiadta mobilos applikációját Prisma név alatt[20]. Az aplikáció előre megadott ismert festői/grafikai stíusokat alkalmazza a telefon kamerája által készített képekre. Az applikáció az előbbiekben bemutatott kutatásokra alapoz. Ugyanakor fontos megjegyezni, hogy maga a stílus alkalmazását a különböző fotókra nem az okostelefon végzi. A szerkeszteni kívánt képet a telefon felkülde egy szervergépre ami majd válaszként a szerkeztett képet küldi vissza.

Maga stílusátvitel nem csak állóképekre alkalmazható, ezt bizonyította Manuel R., Alexey D., Thomas B. tudományos dolgozata[21], valamit ezt próbalja megoldani a jelenlegi dolgozatom is. Értelemszerűen egy adott videót több álló képkocka alkot. Viszont ahhoz, hogy látványos művészeti mozgóképet gyátsunk, nem elegendő maga a videót darabokra vágni és minden képkockára alkalmazni a stílust. Erre adott megoldást Manuel R. és társainak kutatása.

3. fejezet

A rendszer

3.1. Áttekintés

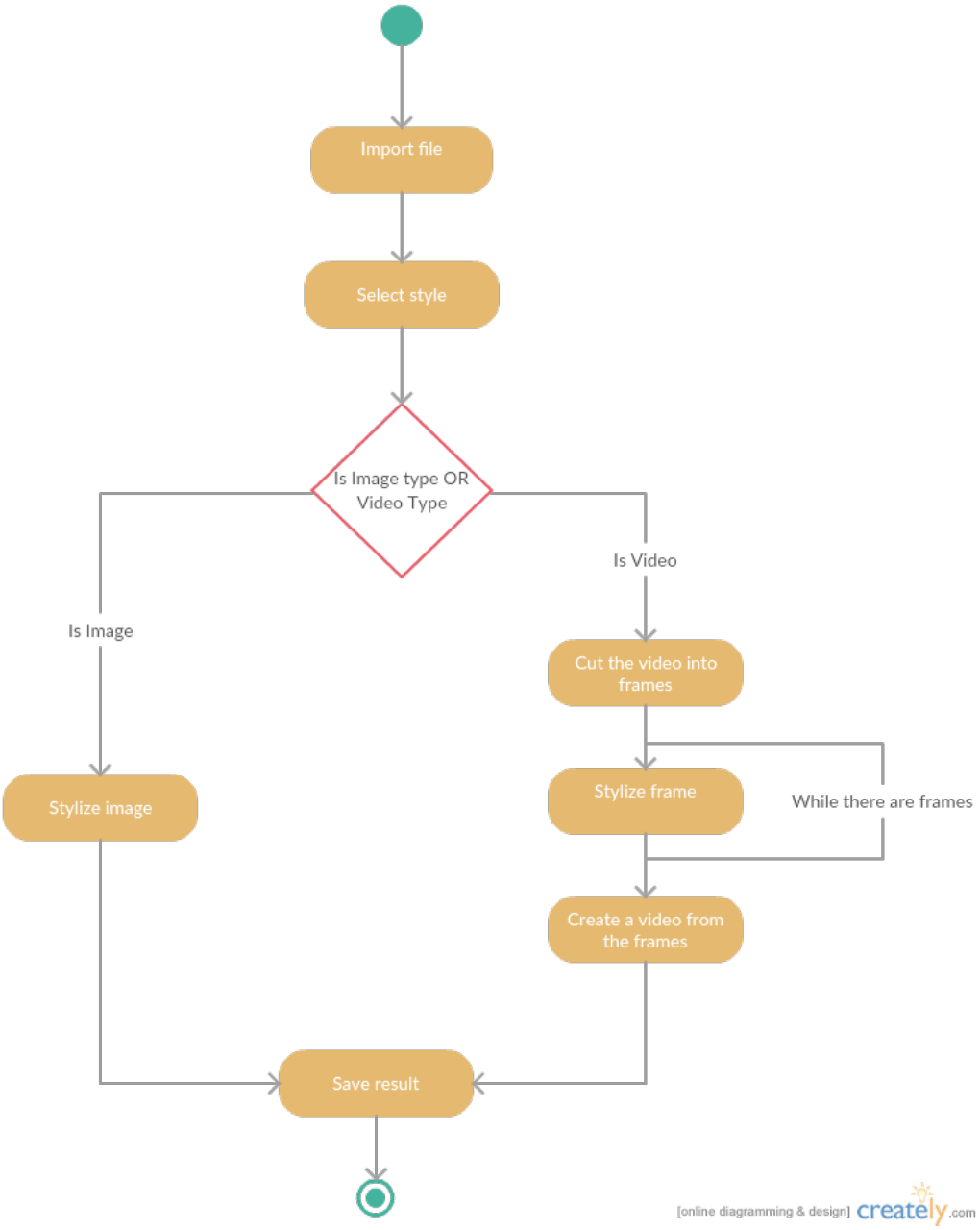
Dolgozatom célja egy olyan multiplatform számítástechnikai szoftver tervezése és fejlesztése ami deep learning-et használva képes híres magyar festők festményeinek a stílusát átvenni és alkalmazni a felhasználó által megadott digitális képekre illetve videókra. A fejlesztett szoftver könnyen használható grafikus felhasználói felülettel rendelkezik és támogatja a Linux valamint a Microsoft Windows alapú operációs rendszereket. A szoftver futtatásához a felhasználónak rendelkeznie kell egy olyan videókártyával ami támogatja az Nvidia CUDA platformot.

A szoftver fejlesztése Python3.5[23] programozási nyelvben történt, viszont egyes esetekben felhasználásra kerülnek egyes előre legyártott önállóan is futtatható állományok. Emellett még használva vannak a következő Python könyvtárak:

- numpy[24]: használata elengedhetetlen akkor ha többdimenziós tömbökkel szeretnénk dolgozni. Nagyon sok matematikai próblémára tartalmaz előre definiált megoldást és efelett tökéletesen használható a tensorflow könyvtár mellett
- tensorflow[7]: talán egyik legismertebb deep learning és mély konvolúciós hálók tanítására kifejlesztett könyvtár. Teljes mértékben támogatja a videókártyán történő programozást.
- PyQt[25]: a szoftver grafikus felhasználói felületének a megvalósításához használatos, emellett komoly feladat orientált párhuzamosítást tud biztosítani.
- OpenCV[26]: képfeldolgozásra szakosodott könyvtár, főleg a különböző kiterjesztésű képek beolvasására és mentésére volt használva.

A rendszer működése felülnézetből nagyon egyszerű 3.1. A felhasználó kiválaszt egy bemeneti állományt, ami kép kiterjesztésű (.jpg, .png) vagy mozgókép kiterjesztésű (.gif, .avi, .mp4) lehet valamint kiválaszt egy stílust a megadottak közül. A rendszer annak

függvényében, hogy milyen bemenetet adtunk, eldönti, hogy kép vagy mozgóképpel kell dolgoznia. Kép esetén egyszerűen lefuttatja a tanulási algoritmust amely során az átveszi a művészeti kép stílusát. Mozgókép esetén képkockákra bontja azt, majd minden képkockára alkalmazva lesz a tanítási algoritmus. Ha összes képkocka szerkesztve lett, akkor a rendszer felépíti a képkockákból a kimeneti videót. Ezek után, függetlenül, hogy kép vagy videó lett a végeredmény, a rendszer kimenti azt egy megadott folderbe.



3.1. ábra. A rendszer működése felülnézetből

3.2. A tanítási módszer állóképek esetében

A rendszer tanításához állokép esetében két legalább képt bemenet szükséges, az eredeti kép amire át szeretnénk ruházni a stílust és a stílust tartalmazó kép, aminek a stílusát át szeretnénk ruházni. Mindkét bemenet esetében felírunk egy veszteség függvényt amik részei a végső nagy tanítási függvénynek.

3.2.1. Az eredeti kép tanításának a veszteségi függvénye

A mély neurálos hálók (Deep Neural Networks) azon típúsai amik a legeredményesebbek a képfeldolgozási feladatok elvégzésben a konvolúciós hálok. Mesterséges intelligencia területén a konvolúciós hálók olyan feed-forward típusú neuronhálók, amiket a biológiai elsődleges látókéregről mintáztak. A háló kifejezetten arra volt tervezve és kifejlesztve, hogy kétdimenziós formákat ismerjen fel. A háló alapértelmezetten több rétegből tevődik össze:

- konvolúciós réteg: a konvolúciós hálók alapkövei. A réteg súlyai úgynevezett konvolúciós szűrők alkotják, amelyek a forward pass lépés során tanítva vannak. Ez a tanítás lépés úgy történik, hogy a szűrőt végigtoljuk a bemeneten és konvolúciónak nevezett műveletet végzünk.

- pooling layer: arra használatos, hogy a bemeneti adathalmazt méretét leszűkítsük úgy hogy a halmaz adott értékein valamely matematikai műveletet végzünk (átlagszámolás, maximum számolás, minimum számolás). Erre azért van szükségünk, hogy növeljük a tanulás gyorsaságát, csökkentsük a számítások komplexitását megakadályozva ezzel az "overfitting" bekövetkezését.

- fully connected layer: minden bemenet mindem más bemenettel kapcsolatban áll.

A konvolúciós hálók súlyzókként konvolúciós szűrőket tartalmaznak. Ezek, tárgyfelisrmerés tanítása esetében, az adott bemeneti kép különböző tulajdonságait fogják tartalmazni. Annak függvényében, hogy a háló topológiájában egy adott réteg hol helyezkedik el, más tulajdonságokat fognak raktározni a szűrök. Az bemeneti réteghez közel álló alacsony szintű rétegek az adott kép pixelinformációit próbálják megjegyezni ezzel szemben a magasabb szinten levő rétegek szűrői különböző tárgyakat, formákat próbálnak megjegyezn[27][28]. A hálók által tartalmazott információ vizualizálható azáltal, hogy rekonstruáljuk a bemeneti képet a súlyzók alapján. Alacsony rétegek esetében apró módósításokkal visszanyerhető az eredeti kép míg magasabb rétegek esetében objektumok, formák nyerhetők vissza.

Az eredi kép tanításához egy már előre betanított mély konvolúciós neuronhálót használunk fel. A háló tudományos kontextusban VGG-19 név alatt terjedt el amit Simonyan

K. és Zisserman A. vezetett be publikációjukban[29] ahol még a "model E" nevet viselte. Ez a háló gép látás és tárgyfelismerés céljából volt bevezetve olyan eredményeket produkálva e téren amik az emberi látással versengenek. Rétegei és topologiájának részletes bemutatása a [29] publikációban történik, valamint a 3.2 figurán is látható. Fontos megjegyezni, hogy a háló rétegeiből használva volt a 16 konvolúciós réteg, valamint az 5 pooling réteg, a teljesen összekötött rétegek nem voltak használva.

A választás azért esett erre az előre betanított hálóra, mivel Yaroslav N. és Roman N. kutatása alapján[16] összehasonlítva a AlexNet, GoogLeNet VGG-16 és VGG-19 halókat, a VGG-19 használata során sikerült a leglátványosabb képeket készíteni.

A konvolúciós neuron háló minden rétege tartalmaz egy adott számú szűrőt, amik különböző tulajdonságokat tartalmaznak a bementi képről. Ebből kifolyólag maga a bemenet kódolva van a szűrőkben. Egy N tulajdonságót tartalmazó réteg N szűrővel rendelkezik amelyek mérete M. Maga a réteg kimenete lementhető egy $N * M$ -es mátrixban. Egy adott réteg válasza egy bemeneti képre vizualizálható, ha gradient descent módszert alkalmazunk egy fehér zajt tartalmazó bemeneti képen. Tehát, legyen R^l egy adott az l haló válasza egy adott bemeneti képre. Legyen W^l ugyanaz ar l háló válasza egy adott fehér zajt tartalmazó bemeneti kép esetében. Akkor felírható a veszteség függvény mint:

$$L(\vec{x}, \vec{r}, l) = \frac{1}{2} \sum R_{ij}^l - W_{ij}^l \tag{3.1}$$

Ahol $\vec{x}$ a bemeneti képet jeleni, $\vec{r}$ pedig azt a kimeneti képet jelenti amit a rendszer generál a rétegek tulajdonságaiból. Mindez Tensorflow környezetben a következőképpen nézne ki:

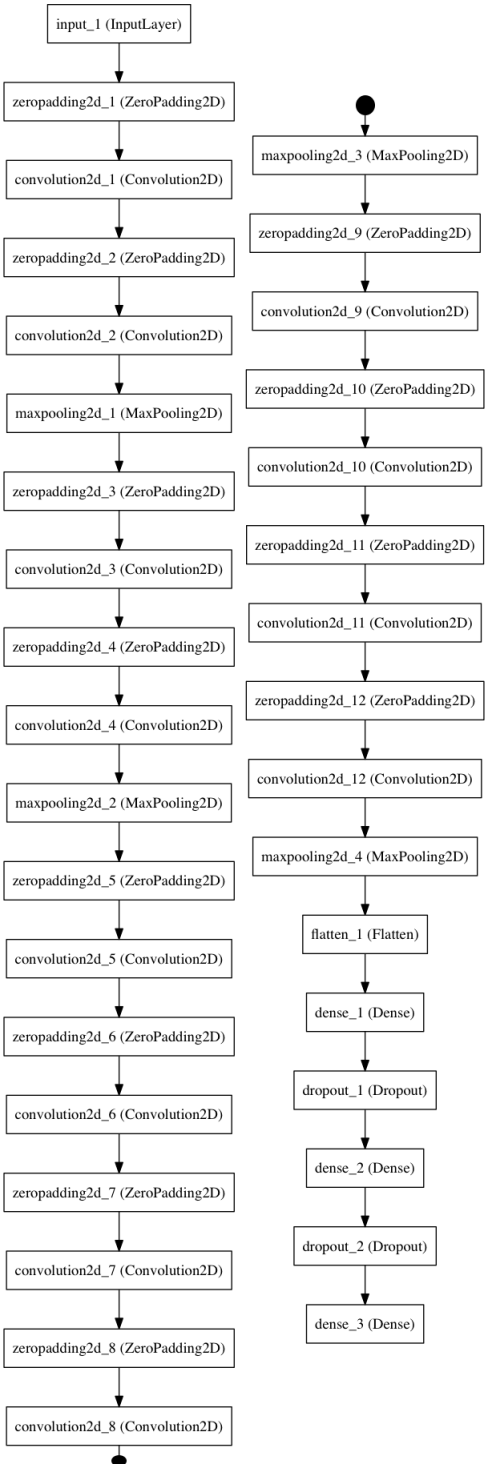
```
for layer_name in CONTENT_LAYER:
        content_loss += content_weight * (2 * tf.nn.l2_loss(
                all_layers[layer_name] - content_features[layer_name]) /
                content_features[layer_name].size)

content_loss /= len(CONTENT_LAYER)
return content_loss
```

A $CONTENT\_LAYER$ tuple típusú ami tartalmazza azoknak a rétegeknek az azonosítóját amik részt vesznek a veszteség függvény kiszámításában. A $content\_features$ valtozó egy lista, ez tartalmazza az összes réteg válaszát a bemeneti eredeti képre. A visszaküldött érték egy tensor típusú, egyik részét fogja képezni a végső optimalizálandó veszteségfüggvénynek.

3.2.2. Az stílus kép tanításának a veszteségi függvénye

as



3.2. ábra. VGG-19 háló topológiája[30]

4. fejezet

A rendszer tesztelése

testing

5. fejezet

Összefoglaló

összefoglaló

Ábrák jegyzéke

Irodalomjegyzék

[1] Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural networks (2012)

[2] Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks (2013)

[3] `https://en.wikipedia.org/wiki/CUDA` (2017.04.24)

[4] `http://caffe.berkeleyvision.org` (2017.04.24)

[5] `https://keras.io/` (2017.04.24)

[6] `http://deeplearning.net/software/theano/` (2017.04.24)

[7] `https://www.tensorflow.org/` (2017.04.24)

[8] `http://torch.ch/` (2017.04.24)

[9] `https://computerstories.net/microsoft-computer-outperforms-human-image-recogniti` (2017.04.29)

[10] Kevin Alfianto, Mei-Chen Yeh, Kai-Lung Hua - Artist-based Classification via Deep Learning with Multi-scale Weighted Pooling (2016)

[11] Rosenblatt F. - The Perceptron: A Probabilistic Model For Information Storage And Organization In The Brain (1958)

[12] Werbos, P.J. - Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences (1975)

[13] LeCun, Yann, Léon Bottou, Yoshua Bengio, Patrick Haffner - Gradient-based learning applied to document recognition (1998)

[14] Dave Steinkraus, Patrice Simard. Ian Buck - Using GPUs for Machine Learning Algorithms (2005)

[15] Gatys, L. A., Ecker, A. S., Bethge - A neural algorithm of artistic style (2015)

[16] Yaroslav Nikulin, Roman Novak - Exploring the Neural Algorithm of Artistic Style (2016)

[17] Justin Johnson, Alexandre Alahi, Li Fei-Fei - Perceptual Losses for Real-Time Style Transfer and Super-Resolution

[18] Ulyanov, D., Lebedev, V., Vedaldi, A., and Lempitsky - Texture networks: Feed-forward synthesis of textures and stylized images

[19] Ulyanov, D., Lebedev, V., Vedaldi, A., and Lempitsky - Instance Normalization: The Missing Ingredient for Fast Stylization

[20] `https://en.wikipedia.org/wiki/Prisma_(app)` (2017.04.29)

[21] Manuel Ruder, Alexey Dosovitskiy, Thomas Brox - Artistic style transfer for videos (2016)

[22] asd

[23] `https://www.python.org/` (2017.04.30)

[24] `http://www.numpy.org/` (2017.04.30)

[25] `https://www.riverbankcomputing.com/software/pyqt/intro` (2017.04.30)

[26] `http://opencv.org/` (2017.04.30)

[27] Gatys, L. A., Ecker, A. S., Bethge, M. Texture synthesis and the controlled generation of natural stimuli using convolutional neural networks (2015)

[28] urlhttps://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html (2017.05.01)

[29] Simonyan, K., Zisserman, A. - Very Deep Convolutional Networks for Large-Scale ImageRecognition (2015)

[30] http://dandxy89.github.io/ImageModels/vgg19/ (2017.05.01)