Neural Style Transfer

Lukáš Florner
Fakulta Informatiky a Managementu
Univerzita Hradec Králové,
Hradec Kralové, Česká Republika
flornlu1@uhk.cz

***Abstrakt*—Cílem této seminární práce je představení a praktická ukázka přenesení stylu, tzv. Style Transfer (ST). Součástí je také stručné uvedení do problematiky Convolutional Deep Network (CNN) a přeneseného učení - Transfer Learning (TL), které jsou pro pochopení a tvorbu ST nezbytné. Také je představen model VGG19, který jsem v práci využil. Práce se krátce zmiňuje i o problematice související s ochranou autorských práv, která s rozvojem neurálních sítí a v nich aplikovaných uměleckých technik nabývá na důležitosti. Závěrem předpokládám spuštění a otestování funkčního modelu ST s otestováním klíčových parametrů sítě (Learning Rate (LR) a počet epoch) na výbrané předloze obsahu a stylu. Aplikace ST je zpracována v jazyce Python a frameworku Keras.**

***Klíčová slova*—Style Transfer; Convolutional Neural Network; VGG19; Deep learning; Keras**

I. ÚVOD

A. OBSAH

Strojové učení, Machine Learning (ML), a jeho aplikace je asi nejrychleji se rozvíjející částí IT. V posledních deseti letech se navíc zdá, že vývoj v této oblasti roste přímo exponenciálně[1] a přináší spoustu nových možností (a u mnoha lidí i obav). Jeho čím dál širší využití je navíc i akcelerátorem vzniku a rozvoje nových služeb, např. cloud computingu včetně služeb jako je Google Colab nebo Azure Machine Learning[1]. Strojové učení dnes potkáme na každém kroku - ať už cestujeme letadlem, nebo se držíme při zemi a vyhledáváme jen spoje vlakové, nebo skřípeme zuby nad tím, že máme ještě dva týdny po nákupu vánočního dárku prohlížeč plný nabídek na odšťavňovače...všude a vždy jsme "sledováni"a každý krok vyhodnocovám algoritmy ML. Ani ne tak konkrétně a cíleně, ale jako součást ohromujícího datového toku, který naše společnost vytváří v každé sekundě své existence (např. v roce 2018 každou sekundu vzniklo hned 5 nových Facebookových účtů, řidiči služby Uber přijali 763 žádostí o jízdu a v aplikaci Tinder uživatelé stačili ohodnotit 16 500 fotografií [14])

Poznámkou o datech a jejich množství se dostáváme k podhoubí a výživě ML. Abychom mohli naučit neurální síť něco užitečného, třeba rozpoznat emoce podle hlasového projevu, musíme mít k dispozici velké množství dat - záznamů hlasu a k jednotlivým záznamům nejlépe i vlastní klasifikaci, s jakými emocemi na straně mluvčího byl daný projev spojen. Správně sestavená neurální síť pak projde naši databázi a spojí si určité rysy s konkrétními třídami emocí. Pěkné na tom je, že k sestavení a naučení takové sítě tvůrce nepořebuje žádný dechberoucí matematicko-statistický aparát nebo doktorát z teoretické informatiky. Vše, co neurální síť dělá, jsou jednoduché operace s vektory a maticemi - jejich sčítání a násobení, v zásadě je ML jen o jednoduché geometrii a hledání té správné "čáry v grafu"[7]. Nic víc. Jen to dělá v tak velkém množství a v tolika vzájemných kombinacích, že někde na konci z té závratné hromady dat vylezou obecné vzory, pomocí kterých lze klasifikovat i data, která síť při tréninku neviděla (věřím tomu, že 75% čtenářů se po takové informaci pro neurální sítě nadchne). I díky tak jednoduchým operacím by si kupříkladu čerstvě prozrazený uživatel Tinderu mohl nechat analyzovat záznam právě proběhlého telefonátu s manželkou a neurální síť mu poskytne predikci, jak moc se má dnes bát vrátit se domů.

Souběžně s tím, jak jednoduché neurální sítě ve své podstatě jsou, je práce s nimi také poměrně složitá. Ono už to tak bývá, že čím máme jednodušší a univerzálněji použitelný systém, tím více si musíme dávat pozor při jeho nastavení, musíme rozumět jeho specifikaci, vnitřní struktuře a parametrům. Musíme se učit s ním pracovat na nespočtu příkladů, sledovat trendy v oblasti a zloušet si je, abychom z onoho v zásadě jednoduchého a univerzálního systému vymáčkli maximum. Musíme věnovat hodiny (a spíš desítky hodin) jemnému ladění, protože velmi brzy zjistíme, že i malé změny v nastavení a struktuře neurální sítě vedou k podstatným změnám výstupu a zvláště na začátku cesty krajinou ML to zdaleka nejčastěji budou změny k horšímu. Taky budeme muset pracovat s daty, se spoustou dat. A ta data nejspíš nebudou v tom úplně *nejvíc nejlepším formátu*, tudíž budeme muset projít peklem preprocessingu. V neposlední řadě budeme muset přemýšlet o kontextu, v jakém chceme naše řešení nasadit, a možná se budeme muset orientovat v mnohem více frameworcích než jen v báječném Kerasu (třeba v samotném TensorFlow, PyTorch a co teprve při nasazení pro svět mobilních aplikací? Go 2 Caffe? Or go 2 Caffe2? A co na to Matlab?). Seznámit se s neurálními sítěmi a oblastí ML vyžaduje nadšení a pár hodin času. Pracovat s ML a pracovat s ním úspěšně už vyžaduje krev, pot, slzy a velkou dávku trpělivosti (a 97% čtenářů nadšených jednoduchostí ML se právě v téhle chvíli rozhodlo dělat raději něco úplně jiného).

[1]Kdo by si před 10-ti lety pomyslel, že bude mít téměř každý majitel téměř jakéhokoliv počítače nebo obyčejného chytrého telefonu možnost trénovat svou vlastní umělou neurální síť na GPU a to úplně zdarma? Vždyť snad málokdo touží v životě zažít něco ještě více vzrušujícího...

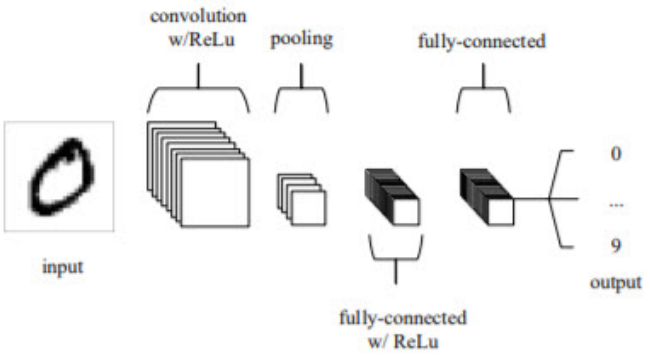
Aby ale autor této práce nebyl jen za šťourala[2], nabídneme na ML i pohled jiný. Mimo spousty užitečných věcí se lze s neurálními sítěmi i pobavit. Kdo má Facebook, resp. Messenger, určitě už si vyzkoušel různé filtry pro selfíčka, přidal svému obličeji brýle, měnil barvu očí, měnil pozadí... Nad ML lze skutečně postavit i zábavní průmysl. I v této práci půjde o jednu ze zábavných aplikací ML a to konkrétně o techniku přenesení stylu. Pomocí ST je možné vzít 2 různé obrázky, např. rodinnou fotografii z dovolené a třeba *Figury v noci provázené fosforeskujícími drahami šneků* od Joana Miro[10] (ano, mezi největší zdroje inspirace v umění skutečně patří alkohol, drogy, sex, bída a psychické poruchy) a zkombinovat je tak, aby výsledný obrázek obsahoval objekty z rodinné fotografie zachycené uměleckým stylem obrazu. Abychom ale mohli něco takového vyzkoušet, musíme rozumět hned dvěma důležitým oblastem ML - konvolučním sítím (CNN) a přenesenému učení - TL. A pak už to bude jízda. Skoro :).

II. ZÁKLADNÍ POJMY

V této kapitole se podíváme na zákoadní pojmy spojené s tématem práce. Jedná se o velmi stručné představení konvolučních sítí, přeneseného učení a přenosu stylu. V samém závěru se také ehce dotkneme otázky autorství.

A. KONVOLUČNÍ SÍTĚ

CNN se používají pro rozpoznávání vzorů v obrazových datech. Obrazová data, tedy jakýkoliv obrázek, si můžeme představit jako matici, kde každý pixel odpovídá jednomu členu a nabývá hodnot odpovídajících vlastní barevné hloubce (např. 0-255 pro každý ze tří barevých kanalů RGB). Taková matice by ale mohla být velmi rozměrná (už jen obyčejný obrázek s rozměry 256x256 px nám vrací úctyhodnou dimenzi matice a 3x tolik, pokud je v RGB...jak velkou matici asi dostaneme u 4K?) a pokud bychom ji chtěli zpracovat tradiční neurální sítí, museli bychom zvolit síť velmi rozlehnou. To je problematické hned ze dvou důvodů - jednak bychom potřebovali masivní výpočetní výkon (nebo spoustu volného času) a jednak bychom se velmi snadno dočkali přeučení sítě, tzv. overfittingu[15], kdy síť může pracovat dobře při tréninku, ale už ne tak dobře, a nebo vůbec, při práci s novými daty (naučí se chybné reprezentace, nebo je vlivem ztráty gradientu vůbec nerozezná). A zde přichází ke slovu konvoluce. Konvoluční síť totiž nezpracovává celý obrázek najednou (u 256x256 by to vyžadovalo 65 536 vstupů), ale rozdělí jej na menší matice (3x3, 5x5...) a pracuje teprve s těmito filtry, tzv. jádry. Z pohledu šířky se tím velikost sítě značně zredukuje. Pomocí těchto jader vyhledává vzory (features, lze říci, že je abstrahuje a hledá tedy obecná pravidla pro definici toho, co je vzor a jak vypadá, např. jak je definovaná hrana), při zpracování využije aktivační funkci (ReLu) a po zpracování předá výsledek další vrstvě, která může opět využít matici filtru a hledat detailnější vzory. U konvolučních sítí totiž platí, že čím hlubší je vrstva[3], tím menší úrovně detailu zpracovává - nejvyšší vrstvy vidí celkovou kompozici a jednotlivé objekty a nejnižší vrstvy pak už typy povrchů, vzor malby apod.[15]
Zpracování pomocí jader je sice báječný nápad, ale sám o sobě není k ničemu. Pokud bychom totiž využívali jen tento přístup, budeme ve stále hlubších a hlubších sítích zpracovávat ten samý obraz a nic moc nového se nenaučíme. Aby tomu tak nebylo, používá se v konvolučních sítích tzv. pooling. Poolingová vrstva mění dimenzi dat a opět to dělá pomocí vlastních jader. Velikost jádra v poolingové vrstvě je většinou menší než ve vrstvě konvoluční - bývá to 2x2. Z této matice se pak, dle typu poolingu, vyextrahuje jediná požadovaná hodnota, čímž dojde k redukci dimenze dat - u matice 2x2 na 1/4[15]. Z toho i vyplývá, proč není dobré volit větší velikost poolingových jader - jedná se o, v určitém smyslu, destruktivní činnost, při které se ztrácí informace a čím větší jádro bychom použili, tím více informací ztrácíme. Zredukovali bychom tak sice hloubku sítě, ale i kvalitu výstupů. Nejčastějším typem poolingu je Max pooling, který z jádra vybere člen (pixel) s nejvyšší hodnotou (tím posilujeme roli kontrastu). Existují ale i další typy (General pooling, Average pooling apod.), které se většnou používají ve specifických případech, nebo u některých modelů (volba typu poolingu značně ovlivňuje výsledek, např. Average pooling vybírá střední tóny obrazu, Max pooling lépe pracuje v případě identifikace ostrých přechodů[3])
Na samém konci CNN je umístěn už úplně obyčejný klasifikátor. Vstupem do klasifikátoru bývá speciální zplošťovací vrstva Flatten, která převede mnohdy multidimenzionální data vycházející z poslední konvoluční vrstvy na jednoduchý vektor (pole) a tím nakrmí hustě propojené vrstvy klasifikátoru (Dense vrstvy), jejichž úkolem je naučit se reprezentovat zjištěné features a provádět predikce.



Obrázek 1: Schéma jednoduché CNN[15] (dostupné z arxiv.org)

B. PŘENESENÉ UČENÍ

Kéž by přenesené učení (TL) fungovalo i ve skutečnosti. Jen si představte - místo několikadenního učení se na zkoušku

[2]Je ale celkem pozdě a z kombinace práce na seminární práci, práce v práci a práce pro rodinu prostě nemůže nekoukat nějaká ta únava jistě generující i právo být za šťourala

[3]Hloubka sítě je vlastně počet vrstev[15] v řadě za sebou - v neurální síti tedy máme vstupní vrstvu přebírající vstupní data, za ní určité množství dalších vrstev, kterým říkáme skryté a které extrahují informace pomocí jednoduchých výpočtů a vzájemné kombinace výsledků, a na konci máme opět viditelnou vrstvu výstupní, která nám poskytuje predikce typu "na 86.8% je to kočička"

z Numerických metod, místo počítání nesčetného množství příkladů byste jen našli někoho, kdo už Numerické metody zvládl (ideálně za *A*) a zkopírovali si od něj jeho vědomosti - jeho reprezentace vzorů (zadání) a schopnosti přiřadit k nim (pokud možno) správné odpovědi. Tímto rádoby vtipným úvodem je o TL vlastně řečeno vše potřebné. Ale protože jsem šťoural, budu ještě chvíli pokračovat.

Ano, TL je o využití už jednou (někým jiným a někde jinde) získaných vědomostí. V oblasti CNN to realizujeme pomocí modelů, které byly otestovány a podávaly dobré výsledky při rozpoznáván a reprezentaci vzorů. Dokonce mohlo jít i o úplně odlišná data - pro klasifikaci typů květin můžeme např. využít model, který byl trénován pro klasifikaci motýlů. To je dáno tím, že modely CNN se neučí rozpoznávat objekty, nýbrž vzory, např. hrany nebo společné vlastnosti, a ty mohou být podobné u mnoha vzájemně nesouvisejících objektů (motýl i květina vskutku mají hrany a vlastnosti hran jsou si vzájemně opravdu hodně podobné - kupříkladu vždycky něco ohraničují). Základním kamenem v naučení se reprezentací jsou váhy sítě. Ty jsou na počátku zvoleny náhodně, ale nakonec se po mnoha průchodech sítí (ideálně) ustálí na určitých hodnotách tak, aby co nejlépe reprezentovaly danou úlohu (tj. minimalizovaly rozdíl ztrátových funkcí mezi trénovacími a testovacími daty). Jsou to pak právě váhy správně natrénované sítě, které obsahují vzory využitelné pro podobná i zcela odlišná zadání. Takové domácí stříbro neuronové sítě.

V praxi se asi nejvíce vychází z modelů trénovaných na databázi Imagenet, což je obrazová databáze s více než 14 miliony obrázků ve 20 000 kategoriích (třídách), z nichž každá obsahuje alespoň stovky souborů[12]. V rámci Imagenet se pořádají pravidelné soutěže ImageNet Large Scale Visual Recognition Challenge (ILSVRC) a vítězové si mohou být jistí, že o jejich modely a váhy bude určitě značný zájem. Příkladem úspěšného modelu je síť VGG. VGG existuje ve dvou variantách - VGG16 a VGG19, přičemž oba modely se liší hlavně počtem vrstev a parametrů. My se zde zaměříme na VGG19 a její velmi stručné představení.

VGG modely byly vytvořeny skupinou Visual Geometry Group na Univerzitě v Oxfordu a v roce 2012 se první z nich - VGG16 - zúčastnila a vyhrála soutěž ILSVRC. Sesadila tak z trůnu do té doby vládnoucí AlexNet[21]. VGG19 se skládá ze 16-ti konvolučních vrstev Conv s velikostí jádra 3x3 sestohovanými do skupin po 2 až 3 vrstvách. Mezi skupinami Conv jsou vkládány MaxPool vrstvy s jádrem 2x2, těch je celkem 5, a na konci je zařazen klasifikátor tvořený 3 plně propojenými vrstvami Dense s aktivační funkcí výstupní vrstvy softmax (což je nejlepší funkce pro klasifikační úlohy[5]). Číslo 19 v označení modelu říká, že model obsahuje 19 váhových vrstev (poolingové vrstvy vlastní váhy nemají). Analogicky VGG16 obsahuje celkem 16 váhových vrstev (plus 5 vrstev poolingu, celkem tedy 21 vrstev). Z hlediska ST je VGG velmi použitelnou vrstvou kvůli snadné implementaci a poměrně vysoké přesnosti klasifikace.

C. PŘENOS STYLU

Přenos stylu pomocí CNN byl poprvé představen v roce 2015 Leonem Gatysem a kol. Velmi poučné a rozsáhlé pojednání autor předkládá v dokumentaci k ST[11] (a existují i vysvětlení pro běžné modely lidských bytostí, např. zde[16]). Já se nyní pokusím o vlastní přiblížení tématu.

Přenosem stylu rozumíme sloučení obrazových informací ze dvou různých předloh. Jedna předloha obsahuje objekty, které mají být zachovány a druhá předloha obsahuje styl, který má být na objekty aplikován. Jde vlastně do jisté míry o kopírování práce skutečného malíře, který maluje objekty dle předlohy (např. potrét podle před ním sedící osoby, nebo město, či skutečně existující krajinu), ale aplikuje na ni svou uměleckou techniku sestávající z tahů štetce nebo uměleckého stylu. Výsledný obraz tak není přesnou kopií reality, ale její fúzí s uměleckým cítěním a šikovností autora. V případě ST budeme nazývat objektovou předlohu jako *content* a stylovou předlohu jako *style* a cílem tak je aplikovat obsah *style* na *content*.

U ST šikovně využíváme vlastností CNN, proto je žádoucí pro aplikaci přenosu stylu konvolucím velmi dobře porozumět. Jak již bylo řečeno dříve, v CNN se horní vrstvy konvolučního modelu zaměřují na celkovou kompozici, jednotlivé objekty a jejich vztahy - zaměřují se tedy na obsah. Zatímco čím hlouběji se do sítě noříme, tím dostáváme vyšší úroveň detailu a abstrakce. Nízké vrstvy modelu (blíže ke klasifikátoru) už nemají přehled o tom, jak vypadá celkový obraz (díky poolingovým vrstvám už vidí zcela jiný obrázek), ale obsahují informace, které lze reprezentovat jako tahy štětce, tvar a barvu linií apod. Tím jsem se dostali blíže stylu. To tedy znamená, že *content* budeme zpracovávat spíše v horních vrstvách sítě a *style* spíše ve vrstvách spodních a díky vlastnostem CNN máme také přístup k hodnotám vycházejícím z jednotlivých vrstev. Pro objektové rysy nám stačí odchytnout hodnoty z N horních vrstev a pro stylové rysy pak hodnoty vrstev zbývajících (pokud se vrátíme k naší analogii s malířem, pak vrchní vrstvy představují to, co malíř reálně vidí a spodní vrstvy představují jeho vnitřní pohled, dojem nebo pocity). Informace získané z obou předloh pak aplikujeme na obrázek stejných rozměrů jako *content*, jehož pixely jsou na počátku nastaveny náhodně (příp. na bílou, černou apod.). Takovému obrázku říkejme *target*. Úlohou sítě je pak počítat nákladovou funkci (průměr ztrátových funkcí) při porovnávání pixelů nastavených v dané iteraci na *target* proti *content* a *style* a snažit se tuto cenu minimalizovat (počítáme tedy ztrátové funkce zvlášť pro *style* a zvlášť pro *content* a cílem je sejít se v bodě, kdy jsou obě ztráty a jejich součet co nejmenší). Z hlediska vah zůstávají hodnoty *style* neměnné (protože styl je víceméně daný naší pozicí v hloubce sítě) a úpravy probíhají zejména na vahách *content* (protože celková kompizice a objekty se musí přizpůsobovat stylu, nikoliv naopak). Z hlediska obrázků se pak nemění ani *style*, ani *content*, pouze *target*.

Pro *style* je také důležitá tzv. Gram Matrix (GM), což je součin matice A s transponovanou maticí A. GM funguje vlastně podobně jako jádra Conv nebo MaxPooling a cílem je zachycení vzorů stylu. A druhá poznámka se týká LR. V běžné CNN je learning rate vždy menší než 1. Vychází to z faktu, že čím větší LR bychom ve fázi učení použili, tím větší je pravděpodobnost, že při sestupu gradientu optimalizátor mine (doslova přeskočí) globální minimum a spokojí se s minimem lokálním. To by vedlo k nízké efektivitě sítě a nepřesvědčivým schopnostem generalizace. Ještě horší případ by ale mohl

nastat, pokud by se sestup gradientu ve spolupráci s velkou hodnotou LR a váhami sítě dostal do pozitivní zpětnovazební smyčky a výstupy výpočtů by tak rostly při každém průchodu vrstvou. Taková situace pak může končit až havárií modelu[4]. Samozřejmě je špatný i přístup navrhovat v síti co nejmenší LR, k čemuž nás možná vede intuitivní předpoklad, že čím menší krok, tím přesnější výpočet[4]. V takovém případě ale můžeme snadno propadnout overfittingu, kdy model velmi přesně kopíruje charakteristiky trénovacích dat, ale nerozumí si s mírně odlišnými daty při validaci. Nebo dojde k situaci, kdy se vlivem menších a menších změn gradientu dostaneme na úroveň, kterou síť nedokáže zpracovat a učení tím končí (pravděpodobně bychom pak viděli řadu neměnných přesností v počtu odpovídajícímu zbývajícím epochám). Jak patrno, LR je vemi důležitý hyperparametr (a možná ten nejdůležitější), jehož nastavení by mělo spadat do oblasti v intervalu $(1.0; 10^{-6}\rangle$[4] (nejlepší hodnota je navíc věcí citu a výsledkem metody pokus-omyl, což mimo jiné znamená, že 2/3 zbývajících zájemců o ML právě opustily třídu). To, co jsme si řekli o LR je dogma. Ale ne pro ST. U ST je naopak žádoucí vyšší hodnota LR (ano, i dvojka funguje skvěle). Je tomu tak zřejmě proto (a zde se dopustím čiré spekulace založené na získaných znalostech a již nejednou klamající intuici), že potřebujeme jakési rozostření. Naším cílem přeci není dostat perfektní reprezentaci vstupních objektů, ale jakousi jejich fúzi, což znanemá, že my ztrátu do jisté míry potřebujeme a přeskoky globálních minim nám až tolik nevadí. Pokud nastavíme malé LR, síť bude fungovat, ale přepočtů bude řádově více a buď se ke stejným výsledkům budeme propracovávat mnohem delší čas, nebo dostaneme jako *target* jen mírně poupravenou předlohu *content*. Někdy až tak mírně, že skoro nepůjde poznat, zda jde o *content*, nebo *target*. Velmi pravděpodobně se nemusíme nijak výrazně obávat ani zpětnovazební smyčky a jejího demoličního působení na naši síť, protože už jsme si řekli, že váhy *style* se při výpočtech nemění.

Poslední, co ještě zbývá v této kapitole vysvětlit, je, co má ST společného s TL. Z odstavců výše vyplývá, že trénování neurální sítě pro ST je časově náročné. Nejde totiž jen o identifikaci vzorů pro klasifikaci objektů, jde i o klasifikaci vzorů stylu a sjednocení obou předloh do jednoho výstupu. Síť pracující na ST provádí o dost víc operací, než kdyby se jen snažila uhodnout, zda je na obrázku pejsek, či kočička nebo jakou mají zrovna náladu. TL nám tedy šetří čas a to je první důvod pro aplikaci. Tím druhým důvodem je, že sítě trénované na Imagenet už obsahují informace o objektech a to o opravdu široké škále objektů a nemusí se tedy znovu učit rozeznávat hrany, učit se, jak rozeznat doubarevně flekatou kočičku od dvou jednobarevných apod. V jistém smyslu se díky použití již předtrénovaného modelu může síť soustředit na to podstatné - na umění.

Dobře, proč TL jsme si vysvětlili (a mě osobně se to vysvětlení docela líbí), ale proč VGG? Protože VGG je model, se kterým se velmi dobře pracuje, v ST dosahuje velmi dobrých výsledků a pro začátečníka je snáze pochopitelný. Např. ResNet50, která je modernější a flexibilnější, nedosahuje tak dobrých výsledků, což je dáno nejspíš architekturou residuálních vrstev, kdy prakticky nedochází ke ztrátě informace do vrstvy vstupující (je přičtena k výstupu z konvoluční vrstvy a určitým způsobem je tak vlastně pronesena, byť ve velmi upravené verzi, skrz celou konvoluční část)[9]. Velmi zajímavou diskuzi k tomuto tématu lze nalézt i na redditu (tam se hovoří např. i o tom, že výhodou VGG v ST je i její robustnost). VGG19 je přitom o něco novější než VGG16, byť z hlediska architektury jsou jako roztomilá konvoluční dvojčátka.

D. PŘENOS STYLU A AUTORSKÉ PRÁVO

Alternativní název této sekce by mohl klidně znít *Bruslení na tenkém ledě*. Pokud je strojové učení revolucí v obasti IT, pak aplikace různých uměleckých stylů pomocí neurálních sítí a obecně umělé inteligence mají potenciál stát se revolucí v oblasti umění a autorských práv. Nejde zdaleka jen o ST, protože v nabídce jsou i jiné možnosti - např. DeepDream, což je technika skládající koláže z naučených vzorů a generující tak zcela nové a mnohdy psychedelické obrázky[8]. Pro mnoho lidí to bylo překvapení, pro mnoho lidí legrace...než se výtvory neurální sítě z této kategorie vydražily za 97 000 USD[6]. Peníze sice putovaly neziskové nadaci Grey Area Foundation, ale přesto...už to jsou peníze. Je samozřejmě otázkou, do jaké míry bude někdy v budoucnu poptávka po aukcích děl umělé inteligence a jestli bude někdo ochotný se takové aukce zúčastnit (zmíněná aukční událost plnila spíš dobročinné, osvětové a možná i personální účely), ale samotný fakt, že se našli kupci takových děl zasel semínka nejistoty zejména v právnických kruzích. Umělá inteligence, resp. hluboké učení, pracuje s existujícími daty a hledá jejich reprezentace. Techniky, jako je ST, pak tyto reprezentace kombinují a vytvářejí zcela nové artefakty. Jenže - na začátku jsou nějaké předlohy. Čí je tedy konečný produkt z autorského hlediska? Tvůrce sítě? Umělé inteligence? Autorů předloh? Z tohoto hlediska je právě ST oblastí nejcitlivější[8], protože při kombinaci dvou předloh se odpovědnost a vlastnická práva hledají nejsnáze. Problém také je, že vznikají veřejné služby, které uživatelům umožňují aplikaci nějaké formy ST doslova online.

V autorském právu neexistuje jednotná regule, která by chránila autory před neautorizovaným využitím jejich děl[8], ale existuje Bernská úmluva o ochraně literárních a uměleckých děl[5]. Bernská úmluva není závazným dokumentem, ale byla ratifikována 180 státy, které ji viceméně začlenily do svých právních rámců. Hlavní pilíře úmluvy jsou právo autora díla na identifikaci díla, tj. uvádění jména autora (nazývané *morální právo*) a dále právo upravující možnosti vytváření kopií, využívání částí díla, jeho veřejnou produkci, převod práv apod.(nazývané *ekonomické právo*). Toto ekonomické právo je vázáno k autorovi a jeho dědicům 50 let po datu úmrtí (délka ochrany ale záleží na lokální implementaci, výjimkou není ani 70-ti leté ochranná lhůta). Aby mohlo

[4]Možná že ne čtenáře, ale mě k takovému předpokladu vedla intuice skutečně zcela neomylně

[5]Sepsána a přijata v roce 1886 v Bernu[2]. Můžeme si jen domýšlet, zda již tenkrát někdo přemýšlel o budoucích třecích plochách mezi umělou inteligencí, právními regulemi, autory a jejich zastupujícími organizacemi

umělecké dílo požívat výhody autorské ochrany (tj. aby mohlo nést označení *umělecké dílo*), musí splňovat 3 podmínky[8]:

- musí existovat (tzn. ve smyslu fyzickém nebo duševním muselo být vykonáno zjevné úsilí a snaha, aby dílo vzniklo a trvalo[6])
- musí se vyznačovat originalitou (tzn. reprodukce uměleckého díla není považována za umělecké dílo[6])
- musí existovat entita, která je za dílo zodpovědná nebo ke kterému se dílo váže (musí tedy existovat člověk, skupina lidí, organizace a buď autor sám nebo jiná taková entita je držitelem autorských práv - tato třetí podmínka je mimochodem i podmínkou existence lidského elementu v procesu, protože i organizace je zastoupena lidmi[6])

Tento seznam podmínek je nicméně široce definovaný a rozhodující je vždy lokální právo, případně jeho mezinárodní interpretace. Pokud se ale těchto bodů budeme držet, pak lze dovozovat, že předlohy ST jsou předmětem autorského díla, pokud nadále trvá vztah autora nebo jeho nástupců k předmětným dílům. Výsledný výstup ST by bylo možné pokládat za autorské dílo, pokud jej generuje tvůrce (příp. skupina nebo organizace), která sestrojila algoritmus neurální sítě a/nebo pokud předlohy použité pro komponování díla jsou veřejné, či k nim má autor potřebná práva. V takovém případě je počítačový program prostředkem umělecké tvorby stejně, jako by to byl štětec nebo sochařské dláto. Ovšem pokud jsou předlohy rozponatelné a jejich autor identifikvatelný, se jedná spíše o reprodukci nebo modifikaci a v takovém případě by se o umělecké dílo jednat nemělo (z pohledu ST se jedná asi o většinu výstupů, pokud nejsou jako předlohy použity např. vlastní obrazy). A poslední (z hlediska přítomnosti bizarní) poznámka k tématu - v případě umělé inteligence, která by byla schopná nezávislého vývoje a generovala by vzory/umělecké výstupy bez nutnosti identifikovatelných předloh (jednalo by se tedy o skutečně originální díla) se dle Bernské úmluvy nemůže jednat o skutečné umělecké dílo, protože se na jeho tvrobě nepodílal člověk. A to už budí dojem úryvku z románu Isaaca Asimova.

Vzhledem k absenci mezinárodního a závazného práva a naopak existenci mnoha vzájemně nekompatibilních národních implementací autorských práv mám závěrem pro zájemce o ST asi jedinou radu - nejbezpečnější je používat předlohy osobní, nebo autorů, kteří jsou více než 70 let po smrti, tvořit pro ososbní potřebu a případné nabídky honorářů za výstupy ze svých neurálních sítí raději s díky odmítat :). A v každém případě uvádět autory použitých zdrojů. To je asi nejméně průstřelná konfigurace práce s uměleckými díly v oblasti umělé inteligence.

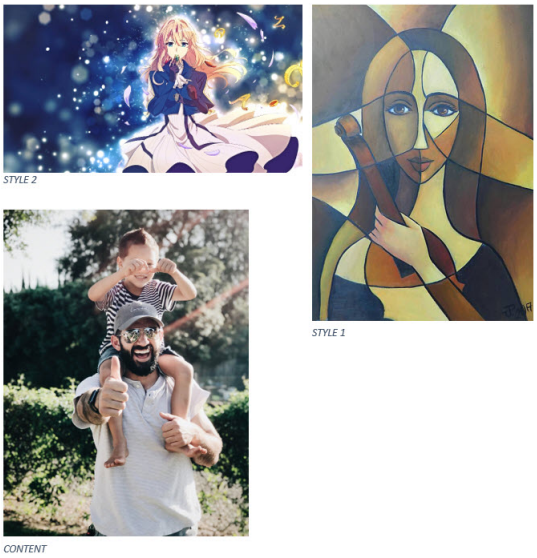
[6]poznámka autora

III. PRAKTICKÁ UKÁZKA

A. OBSAH

Pro svou aplikaci jsem si jako předlohu vybral dvě již zpracované verze ST - jedna pochází z portálu Keras[18] a druhá je upravenou implementací téhož algoritmu[20]. Oba programy jsem modifikoval pro své pořeby a otestoval, v čem se liší. Zdrojové kódy jsou k dipozici zde.
Prvním modelem je ST tutorial od Françoise Cholleta. Tento model prochází sítí ve 4000 iteracích a využívá learning rate s hodnotou 100 v adaptivní konfiguraci optimalizátoru SGD. Druhým modelem je upravený tutorial od Raymonda Yuana, který síť nechá běžet v 1000 iteracích a využívá learning rate 1 v optimalizátoru Adam. U tohoto modelu se také, mimo jiných drobností, mírně liší výpočet ztrátové funkce předlohy stylu. Obě sítě využívají model VGG19
První a prakticky okamžitě viditelné srovnání je rychlost průchodu algoritmu. Tady jednoznačně vyhrává implementace od Raymonda Yuana, jehož síť nejen dokončí celý běh rychleji (měla by, když má jen 1000 iterací), ale rychleji jsou dokončeny i jednotlivé epochy. Domnívám se, že hlavním důvodem je volba optimalizátoru, protože Adam dosahuje vyšší rychlosti zpracování a pomalejší SGD naopak lépe pracuje s adaptivní learning rate[17].

Druhé na řadě je srovnání výsledků. Nejprve se podíváme na předlohy - níže jsou 2 obrázky *style* a jeden cílový obrázek *content*(vlevo dole). V dalším postupu jsem nejprve zpracovával obrázek houslistky (vpravo nahoře) s obsahovou předlohou.



Obrázek 2: Předlohy; vlevo nahoře: Violet Evergarden (dostupné z honknews.com), vpravo nahoře: Houslistka, kubismus (Jiří Petr, dostupné z jiripetr.com), vlevo dole: Noname (dostupné z pinterest.com)

Nejprve otestujeme výchozí nastavení obou sítí (3). Zde je patrné prolínání stylu s obsahem, přičemž SGD přebírá více stylu a celkově působí hladším dojmem. U obrázku s optimalizátorem Adam je viditelný šum a výsledek je vizuálně celkově horší.

Obrázek 3: Výchozí nastavení optimalizátorů

Druhým pokusem je změna nastavení optimalizátorů (4) - pro optimalizátor Adam jsem zvolil LR=100, abych zjistil, jak vysoká hodnota u tohoto optimalizátoru ovlivní výsledek. Ukázalo se, že čím vyšší je hodnota LR, tím více se předlohy rozpouští, resp. vzájemně přemazávají (s extrémní hodnotou, např. LR=1000 u SGD dojde k úplné ztrátě informace a výsledný obrázek je jednolitý). Obrázek vlevo je SGD s LR=1, tedy pro tento případ velmi nízké nastavení hodnoty (podobně vypadá výstup z Adam při LR=0.1). Vidíme, že obrázek doznal jen velmi malých změn, styl se prakticky nepropsal. Řešením není ani zvýšení poštu epoch, z čehoj je patrné, jak důležitým hyperparametrem je LR.

Obrázek 4: Vysoké a nízké hodnoty LR optimalizátorů

Na závěr ukázka výběru jiné předlohy stylu (5). V tomto běhu jsem nechal nastavení optimalizátorů na výchozích hodnotách a ukázka tak dokresluje, že podstatné je nejen nastavení sítě, ale i volba vstupů. Vizuálně zajímavější je zde výstup z Adam, zatímco SGD je více zkreslené a působí deformovaně. To tedy znamená, že nalezení univerzálního nastavení sítě, které by vždy dávalo pěkné vypadající výstupy, je buď velmi obtížné, nebo přímo nemožné.

Obrázek 5: Vliv volby vstupních obrázků

IV. ZÁVĚR

Závěrem musím konstatovat, že i přes svou zábavnou povahu není porozumění, implementace a testování ST jednoduchou záležitostí. Naopak. Pro pochopení celé problematiky je naprosto nezbytné komplexně pochopit téma CNN jako takové a ideálně si nějakou CNN, ať už jednoduchou nebo s aplikací TL, zkusit sestavit a natrénovat. Mohu naštěstí říci, že takovou zkoušku jsem dobrovolně podstoupil a jsem tomu rád, protože bez ”umazaných rukou”bych se v tématu ST zorientoval jen velmi obtížně (díky, Kaggle, za data). Zatímco sestavení CNN je celkem jednoduché a s trochou citu pro hyperparametry (a hromadou dat) lze rychle dosáhnout zajímavách výsledků, u ST to zdaleka neplatí. Ukázal jsem to na příkladech dvou sítí, ve kterých se nedalo příliš odhadnout, jak dopadne výstup, protože nezáleželo jen na nastaveních sítě, ale i na povaze vstupních dat. Přesto a nebo právě proto jsem rád, že jsem si vybral právě toto téma[7], protože jsem se dozvěděl a naučil nejen spoustu zajímavých věcí, ale potvrdil si, že ML je báječná, vzrušující a přitažlivá oblast IT, za kterou rozhodně nechci zavřít dveře s (možným) absolvováním předmětu SMAP ani se (snad úspěšným) dokončením studia.
Jako vedlejší efekt této seminární práce jsem se také seznámil s Overleaf a LaTeX. A mohu potvrdit, že až se pustím do další kapitoly diplomové práce, MS Word k tomu mít otevřený nebudu. Je sice pravdou, že rudá obrazovka s výpisem nic neříkajících chyb, nebo obrázky odmítající se zařadit na správní místa, člověku na klidu nepřidají, ale *kdo se bojí, nesmí do lesa*. Přes úvodní dny hrůzy a zoufalství a přetrvávající mezery ve znalostech už si dnes nedokáži představit lepší nástroj pro tento tip prací, než je LaTeX.

[7] Vlastně jsem si vybíral ze dvou zajímavých témat - Style Transfer a Q-Learning. Při zpětném pohledu musím konstatovat, že to bylo jako vybírat si sebevraždu mečem nebo kanónem. Někdy je prostě náramné štěstí, když člověk vůbec neví, do čeho jde.

citations.bib

Odkazy

[1] *10 Years of Artificial Intelligence and Machine Learning*. en-US. Dub. 2020. URL: https://www.simplilearn.com/ten-years-of-artificial-intelligence-and-machine-learning-article (cit. 07. 01. 2021).

[2] *Bernská úmluva o ochraně literárních a uměleckých děl – Wikizdroje*. URL: https://cs.wikisource.org/wiki/Bernsk%C3%A1_%C3%BAmluva_o_ochran%C4%9B_liter%C3%A1rn%C3%ADch_a_um%C4%9Bleck%C3%BDch_d%C4%9Bl (cit. 15. 01. 2021).

[3] Jason Brownlee. *A Gentle Introduction to Pooling Layers for Convolutional Neural Networks*. en-US. Dub. 2019. URL: https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/ (cit. 15. 01. 2021).

[4] Jason Brownlee. *How to Configure the Learning Rate When Training Deep Learning Neural Networks*. en-US. Led. 2019. URL: https://machinelearningmastery.com/learning-rate-for-deep-learning-neural-networks/ (cit. 15. 01. 2021).

[5] Jason Brownlee. *Softmax Activation Function with Python*. en-US. Říj. 2020. URL: https://machinelearningmastery.com/softmax-activation-function-with-python/ (cit. 15. 01. 2021).

[6] *Can Google's Deep Dream become an art machine?* en. Břez. 2016. URL: http://www.theguardian.com/artanddesign/2016/mar/28/google-deep-dream-art (cit. 15. 01. 2021).

[7] *Deep Learning: Advanced Computer Vision (GANs, SSD, +More!)* en-us. URL: https://www.udemy.com/course/advanced-computer-vision/ (cit. 07. 01. 2021).

[8] Jean-Marc Deltorn. "Deep Creations: Intellectual Property and the Automata". English. In: *Frontiers in Digital Humanities* 4 (2017). ISSN: 2297-2668. DOI: 10.3389/fdigh.2017.00003. URL: https://www.frontiersin.org/articles/10.3389/fdigh.2017.00003/full (cit. 15. 01. 2021).

[9] Priya Dwivedi. *Understanding and Coding a ResNet in Keras*. en. Břez. 2019. URL: https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33 (cit. 16. 01. 2021).

[10] *Figures in the Night Guided by the Phosphorescent Tracks by Joan Miro*. en-US. URL: https://galleryintell.com/artex/figures-night-guided-phosphorescent-tracks-snails-joan-miro/ (cit. 15. 01. 2021).

[11] Leon A. Gatys, Alexander S. Ecker a Matthias Bethge. "A Neural Algorithm of Artistic Style". In: *arXiv:1508.06576 [cs, q-bio]* (zář. 2015). arXiv: 1508.06576. URL: http://arxiv.org/abs/1508.06576 (cit. 15. 01. 2021).

[12] *ImageNet*. en. Page Version ID: 998707267. Led. 2021. URL: https://en.wikipedia.org/w/index.php?title=ImageNet&oldid=998707267 (cit. 15. 01. 2021).

[13] *Latex documentation*. en. URL: https://www.overleaf.com/learn (cit. 06. 01. 2021).

[14] Bernard Marr. *How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone Should Read*. en. URL: https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/ (cit. 06. 01. 2021).

[15] Keiron O'Shea a Ryan Nash. "An Introduction to Convolutional Neural Networks". In: *arXiv:1511.08458 [cs]* (pros. 2015). arXiv: 1511.08458. URL: http://arxiv.org/abs/1511.08458 (cit. 15. 01. 2021).

[16] *Style Transfer Guide — Fritz AI*. URL: https://www.fritz.ai/style-transfer/ (cit. 15. 01. 2021).

[17] Synced. *ICLR 2019 — 'Fast as Adam & Good as SGD' — New Optimizer Has Both*. en. Břez. 2019. URL: https://medium.com/syncedreview/iclr-2019-fast-as-adam-good-as-sgd-new-optimizer-has-both-78e37e8f9a34 (cit. 15. 01. 2021).

[18] Keras Team. *Keras documentation: Keras API reference*. en. URL: https://keras.io/api/ (cit. 14. 01. 2021).

[19] Keras Team. *Keras documentation: Neural style transfer*. en. URL: https://keras.io/examples/generative/neural_style_transfer/ (cit. 14. 01. 2021).

[20] TensorFlow. *Neural Style Transfer: Creating Art with Deep Learning using tf.keras and eager execution*. en. Zář. 2018. URL: https://medium.com/tensorflow/neural-style-transfer-creating-art-with-deep-learning-using-tf-keras-and-eager-execution-7d541ac31398 (cit. 15. 01. 2021).

[21] *Understanding the VGG19 Architecture*. en. Ún. 2020. URL: https://iq.opengenus.org/vgg19-architecture/ (cit. 15. 01. 2021).