FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání diplomové práce

Název:	Augmentace medicínských dat pomocí hlubokých generativních modelů
Student:	Bc. Michal Přibyl
Vedoucí:	Ing. Jakub Žitný
Studijní program:	Informatika
Obor / specializace:	Znalostní inženýrství
Katedra:	Katedra aplikované matematiky
Platnost zadání:	do konce letního semestru 2021/2022

Pokyny pro vypracování

Prozkoumejte současné techniky, které používají hloubkové generativní modely (DGM) pro augmentaci dat.
Natrénujte segmentační nebo klasifikační model na dostupných medicínských zobrazovacích datech (kožní, rentgen nebo CT) a analyzujte, jak DGM zlepšují výsledky ve srovnání s tradičními metodami předzpracování dat.
Porovnejte efektivitu svého modelu s referenčními výsledky z literatury nebo open-source modelů a prodiskutujte jejich výhody a nevýhody.
Kód publikujte na internetu jako open-source a zajistěte, aby výsledky byly reprodukovatelné.

Elektronicky schválil/a Ing. Karel Klouda, Ph.D. dne 16. června 2020 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Augmentace medicínských dat pomocí hlubokých generativních modelů

Bc. Michal Přibyl

Katedra aplikované matematiky
Vedoucí práce: Ing. Jakub Žitný

6. května 2021

Poděkování

Na tomto místě bych chtěl poděkovat svému vedoucímu Ing. Jakubovi Žitnému za cenné rady při realizaci této práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 2373 odst. 2 zákona č. 89/2012 Sb., občanský zákoník, ve znění pozdějších předpisů, tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo"), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu) licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 6. května 2021 .....

České vysoké učení technické v Praze
Fakulta informačních technologií

Odkaz na tuto práci

Přibyl, Michal. *Augmentace medicínských dat pomocí hlubokých generativních modelů*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2021.

Abstrakt

Rakovina kůže je dnes jedna z nejčastěji se vyskytujících rakovin na světě [1]. Pro vyšetření povrchu kůže se používá dermoskop, jenž pořizuje samotné dermoskopické snímky kůže. Tyto snímky dokáží zachytit detaily, které mohou lidskému oku uniknout. Tyto snímky pak posuzují v mnoha případech dva lékaři, jeden který určuje diagnózu a druhý který ho kontroluje (určuje vlastní diagnózu). V dnešní době tato role kontrolujícího lékaře může být nahrazena přesnými klasifikačními modely, jenž potřebují velké množství dat, kterých je bohužel nedostatek.

Tato práce zkoumá využití hlubokých generativních modelů v generování dermoskopických obrázků, které by pomohly řešit zmíněný problém s nedostatkem dat. S generativním modelem StyleGAN2ADA bylo v této práci dosaženo zlepšení klasifikačního modelu v rozeznání dvou typů kožních onemocnění NV a MEL, resp. jejich sensitivit o 4.29 % a 0.72 % oproti použití klasické augmentace dat.

Klíčová slova GAN, StyleGAN2ADA, Pix2Pix, CDCGAN, ISIC, augmentace dat, dermoskopické obrázky, medicínská data

Abstract

Skin cancer is now one of the most common types of cancer [1]. A dermatoscope is a tool, which takes the dermoscopic images of the skin. These images can capture details that can escape the human eye. These images are then evaluated, in many cases, by two doctors, one who determines the diagnosis and the other who checks it (determines his diagnosis). Nowadays, this role of the examining physician can be replaced by precise classification models, which need a large amount of data, which is unfortunately lacking.

This work examines the use of deep generative models in the generation of dermoscopic images, which would help solve the mentioned problem of lack of data. With the generative model StyleGAN2ADA, the classification model was improved in the recognition of two types of skin diseases NV and MEL. Class sensitivities improved by 4.29 % and 0.72 % compared to using classical data augmentation.

Keywords GAN, StyleGAN2ADA, Pix2Pix, CDCGAN, ISIC, data augmentation, dermoscopic images, medical data

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Rakovina kůže je jedna z nejčastěji se vyskytujících rakovin na světě, kde melanom je jeden z nejsmrtelnějších druhů [1]. Dermoskopie je vyšetření povrchu kůže, ve kterém se využívá dermoskop, jenž pořizuje samotné dermoskopické snímky kůže.

Dermoskopie pomáhá diagnostikovat rakovinu kůže lépe než samotná vizuální kontrola, avšak aby toho bylo docíleno, tak samotnou diagnostiku dermoskopických snímků musí provádět zkušený dermatolog. [9]

Motivací této práce je vylepšení klasifikačních modelů dermoskopických obrázků provádějící automatickou diagnózu nádorových onemocnění a pomoci tak lékařům k určení diagnózy a nebo její kontrole. Práce zkoumá využití hlubokých generativních modelů (GAN) v generování dermoskopických obrázků, které mohou být využity pro rozšíření trénovacího datasetu zmíněného klasifikátoru.

V první části této práce je provedena analýza tohoto problému. Zabývá se problematikou augmentací dat, jejich případných výhod a nevýhod. Dále pak je zde popsáno detailní fungování GAN modelu, popis jeho trénování a evaluace. Na konci této části jsou pak analyzovány moderní typy GAN modelů.

Následující část se věnuje tvorbou klasifikačního modelu dermoskopických obrázků, jenž bude využit ve finálním porovnání typů augmentací dat.

Ve třetí části je popsáno trénování GAN modelů, které budou porovnány mezi sebou a nejlepší z nich pak poslouží ke generování dermoskopických obrázků použité pro výsledné porovnání typů augmentací dat v následující části této práce.

Kapitola 1

Cíl práce

Jednotlivé cíle této práce jsou popsány v bodech níže:

- analyzovat využití moderních hlubokých generativních modelů pro augmentaci dat
- natrénovat klasifikační model na dermoskopických obrázcích
- zveřejnit kód klasifikačního modelu
- zveřejnit natrénovaný klasifikační model
- natrénovat hluboké generativní modely a vybrat z nich ten nejlepší
- zveřejnit natrénovaný hluboký generativní model
- analyzovat efektivitu augmentace dat pomocí hlubokého generativního modelu s klasickou augmentací dat
- porovnat efektivitu augmentace dat pomocí hlubokého generativního modelu s jiným, již existujícím modelem

Kapitola 2

Analýza problému

V této kapitole se analyzuje využití GAN modelů pro medicínská data. Nejprve se zde popisují informace o medicínských datech, o tom jaké existuje jejich využití, typy a jaké problémy jsou spojené s jejich použitím v praxi.

V další části této analýzy se nastiňuje problematika augmentace dat, ve které je stejně jako u medicínských dat, popsáno její rozdělení do typů. U jednotlivých typů se pak rozebírá jaké jsou jejich výhody a nevýhody.

Následující část se věnuje GAN modelu, kde je popsaná jeho základní architektura, tj.např. z jakých všech komponent se skládá. Dále pak se tato část věnuje problematice trénování GAN modelů a všem známým problémům, které se objevují při tréninku. Další problematikou kterou se zabývá tato část je pak evaluace GAN modelů, kde je podrobný popis různých evaluačních metod i s popisem jejich výhod a nevýhod.

Poslední část této kapitola je věnována moderním typům GAN modelů, u nichž je popsáno jejich fungování, spolu se srovnáním jejich výhod a nevýhod.

2.1 Medicínská data

V medicíně se data využívají k mnohým účelům od určování pravděpodobnosti výskytu chorob až po pomoc lékařům zvolit vhodnou léčbu nebo určit test, který se má provést pro upřesnění diagnózy pacienta. Data však nejsou využívána jenom lékaři, ale jsou použita i různými matematickými modely, neuronovými sítěmi a dalšími k automatickým diagnózám [10]. Tyto modely jsou pak určitým nástrojem, který pomáhá lékařům v rozhodování, výzkumu nebo studiu. V budoucnu by pak tyto nástroje mohly i nějaké specializované lékařské profese nahradit.

2.1.1 Typy dat

Typů dat, které využívají výše zmíněné modely, existuje velké množství. Data se dají rozdělit do dvou základních skupin: strukturovaná a nestrukturovaná.

Mezi strukturovaná data v medicíně patří např. různé informace o pacientech, jako jsou diagnózy, choroby, nebo jejich celá zdravotní historie.

Nestrukturovaná data jsou pak různé obrazové snímky nebo text. Mezi obrazové snímky v nestrukturovaných datech patří např. dermatologické snímky, jenž zachycují choroby kůže, CT snímky zobrazující vnitřní orgány a histologické snímky, které zobrazují mikroskopické struktury nemocných tkání. Příkladem textu v nestrukturovaných datech jsou různé poznámky nebo příkazy lékařů, kde se využívá NLP (natural language processing).

2.1.2 Problém s daty

Výše zmíněné modely, především však neuronové sítě, potřebují pro svoje fungování velké množství kvalitních dat, kterých je však nedostatek. Je to tím, že medicínská data jsou velmi těžko získatelná. To je způsobeno např. nutností použít drahé lékařské přístroje nebo následné olabelování dat (rozdělení dat do kategorií), k čemuž jsou potřeba odborníci daných oborů, jako jsou dermatologové a jiní, to je časově i finančně velice náročné. [11]

2.2 Augmentace dat

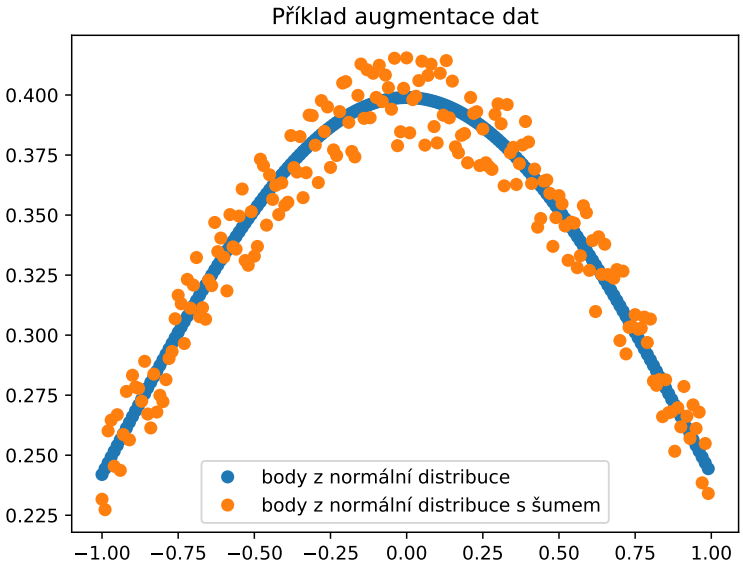
Augmentace dat označuje metody používané k tvorbě nových dat, resp. k rozšíření původního datasetu. Klasické metody augmentace dat pak rozšiřují dataset pomocí náhodných změn provedených na původních datech. U obrázků to může být např. náhodná změna rotace nebo barev. Nově vytvořená data by měla být ze stejné distribuce jako jsou data původního datasetu, tj. náhodné změny, které se při tvorbě nových dat provádějí nad daty, by neměly být příliš velké.

Na Obrázku 2.1 je znázorněn příklad klasické augmentace dat, kde modré body v grafu jsou původní data z normální distribuce a oranžové body ty augmentované, jenž jsou vytvořené z modrých bodů, ke kterým byl přidán náhodný šum.

Augmentace dat je tedy jeden z možných způsobů, jak řešit nedostatek dat a zároveň zvyšuje generalizaci trénovaných modelů, díky novým datům jenž nebyly v původním datasetu.

2.2.1 Typy augmentací dat

Metody pro augmentaci dat se dají rozdělit do dvou skupin. První skupinou, která je nejčastěji používaná, je klasická augmentace dat [12], čímž se myslí jasně definované transformace dat, jako jsou např. u obrázků výše zmíněná rotace nebo přidaní šumu. Druhou skupinou pro augmentaci dat jsou pak různé generativní modely, jako jsou např. GANs (Generative Adversarial Networks), pomocí kterých se může rozšířit výsledný dataset.



Obrázek 2.1: Příklad augmentace dat.

2.2.1.1 Klasické metody augmentace dat

Hlavní výhodou klasických metod pro augmentaci dat je především jejich jednoduchost použití. Tyto metody se dají ihned použít, jelikož jsou dané transformace známe a na rozdíl od generativních modelů se před použitím nemusí trénovat. Mezi její velké nevýhody však patří to, že pro každý dataset se musí nalézt vhodná množina transformací, která příliš nepozmění datový bod.

2.2.1.2 Augmentace dat pomocí generativních modelů

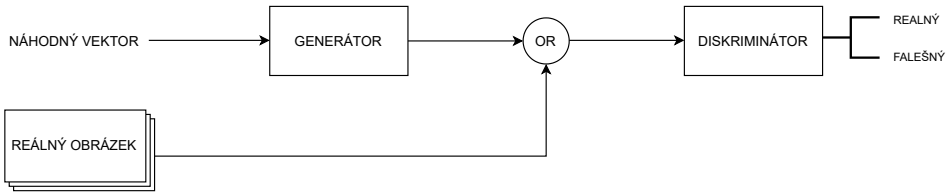
Jako generativní modely pro augmentaci dat se používají především GAN modely [13][14], jenž nyní dosahují dobrých výsledků u nestrukturovaných dat, především pak u obrázků. Avšak jak již bylo zmíněno výše, nevýhodou GAN modelů oproti klasickým metodám augmentací dat je komplexnost modelů a nutnost jeho trénování. Díky trénování se však dokáže automaticky přizpůsobit jakémukoliv datasetu.

2.3 GAN

Generative adversarial networks neboli GANs jsou machine learningové modely, které se dokáží naučit distribuci dat, nad kterými je trénován. Tvůrci původní GAN jsou Ian Goodfellow a jeho kolegové, kteří ho představili v roce

2014 [15]. Tyto modely se využívají převážně pak pro nestrukturovaná data, jako jsou obrázky, videa či zvuk [16].

GAN se skládá ze dvou neuronových sítí, generátoru a diskriminátoru. Generátor se snaží zlepšit v generování realističtějších dat, tak aby je diskriminátor nerozpoznal od těch reálných. Diskriminátor se naopak snaží zlepšit v identifikaci těchto vygenerovaných dat, tak aby je dokázal lépe rozpoznat, obě tyto sítě tedy mezi sebou soupeří.



Obrázek 2.2: GAN architektura. [2]

2.3.1 Generátor

Generátor dostane na vstupu náhodný vektor pevně dané délky, který je z nějaké známé distribuce, jako např. uniformní nebo normální, ze kterých jdou jednoduše generovat data. Výstupem generátoru jsou pak data připomínající ta reálná. Architektura generátoru je libovolná, tj. může být použita neuronová síť jako je multilayer perceptron (MLP), convolutional neural network (CNN), nebo jiné. [17][15]

2.3.2 Diskriminátor

Diskriminátor dostává na vstupu reálná data z původního datasetu a zároveň také i falešná data vygenerovaná generátorem. Jedná se o binární klasifikátor, jenž odhaduje pravděpodobnost, že daný vstup patří do reálného datasetu či nikoli. Stejně jako u generátoru zde může být libovolná architektura, pokud stále respektuje potřebnou dimenzi vstupu a výstupu. [17][15]

2.3.3 Trénování

Trénovací proces GAN modelu se skládá z minimax hry dvou hráčů, ve které se diskriminátor snaží učit rozlišovat reálná a generovaná data a generátor se snaží o generování co nejrealističtějších dat, které diskriminátor již nedokáže rozlišit od těch reálných.

GAN model, resp. jeho generátor, na rozdíl od jiných generativních modelů není trénován přímo, místo toho je generátor trénovaný pomocí diskriminátoru a diskriminátor se v průběhu trénování učí rozpoznávat reálná a vygenerovaná data.

Standardní GAN má dvě loss funkce: $\log(1 - D(G(z))$ a $\log(D(x)) + \log(1 - D(G(z))$, jednu pro trénování generátoru a druhou pro diskriminátor, kde se generátor snaží minimalizovat tuto funkci a diskriminátor maximalizovat [18]. Obě funkce jsou odvozené z jediné formule 2.1, ve které $D(x)$ je odhad pravděpodobnosti diskriminátoru o tom, zda je x reálný datový bod, E_x očekávaná hodnota přes všechny datové body, $G(z)$ pak je výstup generátoru se vstupním náhodným vektorem z, $D(G(z))$ je odhad pravděpodobnosti diskriminátoru o tom, zda je vygenerovaný datový bod reálný a E_z je pak očekávanou hodnotou přes všechny vygenerované datové body.

$$E_x[\log(D(x))] + E_z[\log(1 - D(G(z))] \quad \text{[18]} \quad (2.1)$$

Velmi často však generátor používá loss funkci $log(D(G(z))$, kterou se snaží maximalizovat, místo původní $log(1-D(G(z))$, kterou se snažil minimalizovat. Tato nová funkce pomáhá odstranit problém v počátku trénovacího procesu, kde se GAN model dostal do situace kdy se již nedokázal dále zlepšovat, což bylo způsobeno tím, že diskriminátor na začátku dokázal příliš jednoduše rozpoznat zda se jedná o reálný nebo vygenerovaný datový bod. [18]

Trénování tedy cílí k ideálnímu bodu, kdy diskriminátor dosahuje přesnosti kolem 50 %, tj. bodu kdy generátor generuje data, které již diskriminátor nedokáže rozlišit od těch původních.

V každém kroku trénování se aktualizují váhy generátoru nkrát a diskriminátoru mkrát, kde n a m jsou hyperparametry modelu, které je třeba vhodně zvolit.

2.3.4 Evaluace

V současné době se stále hledají vhodnější metody [19][20], kterými lze určovat kvalitu generátoru a jeho produkovaných dat v průběhu trénování nebo jakým způsobem porovnávat jeho architekturu či konfiguraci. Tento problém s evaluací GAN modelů je jeden z jeho současně známých otevřených problémů [16].

U GAN modelu, standardní loss funkce generátoru a diskriminátoru obvykle měří míru toho, jak dobře generátor zvládá vytvářet data, které oklamou diskriminátor, tj. poměřují se vůči sobě navzájem. A tak hodnoty standardní loss funkce příliš nevypovídají o výsledné kvalitě nebo diversitě generovaných dat.

Evaluační metody GAN modelů se dají rozdělit do dvou kategorií: kvalitativní a kvantitativní, kde kvalitativní metody jsou založené na lidském ověřovaní nebo analyzování vnitřních stavů modelů a kvantitativní jsou založené na výpočetních technikách, jejichž výstupem je nějaké číselné skóre. Obě tyto kategorie mají svoje výhody a nevýhody. [19]

2.3.4.1 Kvalitativní metody

Kvalitativní metody založené na lidském ověřování jsou hojně využívané pro jejich jednoduchost a intuitivnost. Mají však mnoho nevýhod, jako je cena, časová náročnost nebo neobjektivnost, kde různí lidé, subjekty, poskytnou velice rozdílné závěry o kvalitě dat, je tedy potřeba mít velké množství takovýchto subjektů jejichž závěry se výsledně průměrují.

Dalším způsobem kvalitativního ohodnocování modelů je zkoumání vnitřních vlastností generátoru, resp. diskriminátoru, a jeho latentního prostoru. [19][20]

Jedna z hojně využívaných kvalitativních metod pro ohodnocení reálnosti obrázků je např. Rapid Scene Categorization, kde má subjekt za úkol rozpoznat v krátkém časovém, měnícím se intervalu, zda je jemu právě zobrazený obrázek pravý či nikoliv.

Dalším příkladem kvalitativní metody je zkoumání latentního prostoru, tj. vstupních náhodných vektorů [21]. Procházení latentního prostoru se provádí nějakou malou změnou hodnot ve vstupním vektoru. V prostoru se pak hledají smysluplné směry, díky kterým se provede předvídatelná změna ve výstupu generátoru. V datasetu s obličeji pak takováto předvídatelná změna může být změna barvy očí ve výstupním obrázku.

Dále pak lze v latentním prostoru zkoumat úroveň toho, jak kvalitně umí model vytvářet nové obrázky (datové body), např. pokud existují nějaké dva vstupní vektory z_1 a z_2, které generují realistické obrázky, lze pak zkoumat obrázky které vzniknou ze vstupních vektorů, které leží na úsečce ohraničené body z_1 a z_2.

Zástupci kvalitativních metod:

- Rapid Scene Categorization
- Zkoumání latentního prostoru

2.3.4.2 Kvantitativní metody

Kvantitativní metody jsou založené na nějakých výpočetních technikách vracející číselné skóre, které výsledně určuje kvalitu či diversitu generovaných dat. Většina kvantitativních evaluačních metod bere generátor jako black box generující data a nevyžadují tak odhad pravděpodobnostní distribuce dat, na rozdíl od jiných metod které ji vyžadují, např. average log-likelihood [19]. Tyto metody jsou detailněji rozebrány níže.

Average log-likelihood
Tato metoda se využívá pro trénování a evaluaci generativních modelů. Average log likelihood se měří na N realných datových bodech pod generovanou hustotou viz výraz $\frac{1}{N}\sum_i \log p_g(x_i)$, kde x_i je i-tý trénovací datový bod a p_g je

hustota pravděpodobnosti generovaných dat, avšak tato hustota je neznámá a tak se musí odhadovat např. pomocí KDE (Kernel Density Estimation). [19]

KDE má však problém s daty s velkou dimensionalitou, jako jsou např. obrázky, kde i s obrovským množstvím dat nedokáže věrně aproximovat pravou hustotu p_g.

Dalším problémem této metody je, že obecně nevypovídá o kvalitě generovaných dat, tj. model může mít nízkou log-likelihood a generovat kvalitní data nebo mít vysokou log-likelihood a generovat nekvalitní data. [20]

Inception Score

Je metoda využívající pretrained model Inception Net (může být však využit jiný model), který pomáhá získat požadované vlastnosti z generovaných dat, tj. jejich kvalitu a diversitu vzhledem k dané třídě. Tato metoda pak ve svém výpočtu využívá pouze generovaná data a ta reálná data ignoruje.

Inception Score měří průměrnou KL divergenci mezi podmíněnou distribucí $p(y|x)$ a marginální distribucí $p(y)$ viz výraz 2.2, kde y označuje třídu a x je datový bod vygenerovaný generátorem.

U kvalitních a diverzních generovaných dat se očekává, že podmíněná distribuce $p\left(y|x\right)$ bude mít nízkou entropii a distribuce $p(y)$ ji naopak bude mít vysokou. Toto skóre tedy indikuje nekvalitní data svou minimální hodnotou 1 a kvalitní data pak maximální hodnotou, což je počet tříd. [22][23].

Tato metoda má však stejný problém jako Average log-likelihood s detekováním overfittingu a nebo mode collapsu.

$$exp\left(E_x\left[\mathrm{KL}\left(p(y|x)\|p(y)\right)\right]\right) \qquad [22] \quad (2.2)$$

Frechet Inception Distance

Frechet Inception Distance neboli FID metoda vznikla jako vylepšení oproti stávající Inception Score, která vůbec nesrovnávala generovaná data s těmi reálnými. FID skóre má tedy za cíl porovnávat statistiky generovaných a reálných dat.

Stejně jako Inception Score k výslednému výpočtu skóre používá pretrained model Inception Net, u které použije výstup ze specifické vrstvy, ze které získá výstupy, označované také jako activations, z generovaného, resp. reálného, datového bodu. Tyto activations všech generovaných, resp. reálných, datových bodů vytvoří vypočítáním průměrů a kovariancí dvě spojité vícerozměrné normální distribuce. Tato metoda předpokládá, že activations jsou z normální distribuce, což ale v případě datasetu nemusí vždy platit. [24]

Frechet distance 2.3 (vzdálenost) mezi dvěma výše zmíněnými distribucemi je použita k výslednému určení kvality či diversity generovaných datových bodů, kde μ_r, resp. μ_g, jsou vektory průměrů a Σ_r, resp. Σ_g, jsou kovarianční matice.

$$FID(r,g) = \|\mu_r - \mu_g\|^2 + Tr\left(\sum_r + \sum_g -2\left(\sum_r \sum_g\right)^{\frac{1}{2}}\right) \qquad [22] \ (2.3)$$

Tato metrika je založená na měření vzdálenosti reálné a generované distribuce dat, a tak její nízké hodnoty jsou známkou kvalitnějších dat a vyšší hodnoty méně kvalitních dat.

FID se dá využít i pro ohodnocení diversity v případě problému s více třídami [25]. Diversita se spočítá tím způsobem, že se pro každou třídu získá hodnota FID a průměr těchto hodnot pak určuje výslednou diversitu generovaných dat. Takto vypočítaná diversita se nazýva Intra FID.

Mezi výhody FID patří lepší odolnost vůči šumu a dokáže si lépe poradit s mode droppingem než Inception Score [22]. Další výhodou je korelace hodnoty FID s kvalitou obrázků [24]. Mezi největší nevýhody této metriky pak patří její výpočetní náročnost, především pak pro Intra FID, kde se výpočet opakuje pro každou třídu.

Adversarial Accuracy and Adversarial Divergence

Tyto metriky se snaží určit jak moc si jsou distribuce $p_g(x)$ a $p_r(x)$ (generovaných a reálných dat) blízké. K výpočtu používají dva klasifikátory. První klasifikátor se nejprve pomocí vygenerovaných dat musí naučit podmíněnou distribuci $p_g(y|x)$, kde y je vektor tříd a x datový bod, a druhý klasifikátor se musí naučit pomocí reálných dat podmíněnou distribuci $p_r(y|x)$. [22]

Adversarial Accuracy určuje přesnosti, které byly získány z výše zmíněných klasifikátorů na validační množině (množina reálných dat, kterou druhý klasifikátor nepoužil při trénování). Pokud jsou přesnosti obou klasifikátorů podobné, tak jsou si podobné i distribuce $p_g(x)$ a $p_r(x)$.

Adversarial Divergence počítá KL divergence mezi $p_g(y|x)$ a $p_r(y|x)$. Čím je tato metrika nižší, tím jsou si dvě distribuce podobnější. Nejnižší hodnotou pak je 0, což znamená že jsou dvě distribuce totožné pro všechny datové body z validační množiny.

Nevýhodou těchto metrik je, že se dají použít pouze pro data se třídami nebo se pro ně musí získávat např. manuální anotací, která může být cenově nebo i časově náročná. [22]

SSIM

SSIM je metrika, která se snaží zaměřit na určité aspekty v obrázcích, jenž pro člověka nejsou tolik rozpoznatelné nebo důležité. Porovnává dva obrázky x a y, resp. jejich pixely a určitý počet jejich sousedů, dle jejich vypočítaného jasu I, kontrastu C a struktury S [20]:

$$I(x,y) = \frac{2\mu_x\mu_y + C1}{\mu_x^2 + \mu_y^2 + C1}, C(x,y) = \frac{2\sigma_x\sigma_y + C2}{\sigma_x^2 + \sigma_y^2 + C2}, S(x,y) = \frac{\sigma_{xy} + C3}{\sigma_x\sigma_y + C3} \quad [20] \ (2.4)$$

Proměnné μ, σ značí průměr a standardní odchylku intensity pixelů v určitém okně obrázku centrované buďto v x nebo v y. Proměnná σ_{xy} je korelační koeficient mezi příslušnými pixely z obou oken obrázků centrované v v x, resp. v y. $C1$, $C2$ a $C3$ jsou pak konstanty, které přispívají k lepší numerické stabilitě.

Z výše zmíněných komponent I, C a S se pak vypočítá výsledná metrika:

$$SSIM(x,y) = I(x,y)^\alpha C(x,y)^\beta S(x,y)^\gamma \quad [20] \ (2.5)$$

Zástupci kvantitatiních metod:

- Average log-likelihood
- Inception Score
- Frechet Inception Distance
- Adversarial Accuracy
- Adversarial Divergence
- SSIM

2.3.5 Známé problémy při trénování

GAN modely již dosáhly slušných výsledků pro generování obrázků a jiných nestrukturovaných dat. Avšak stále se u nich objevují nevyřešené problémy (výzvy) se kterými se tyto modely potýkají v trénovacím procesu, jako je např. mode collapse, problém s dosáhnutím nebo udržením konvergence a nestabilita. [26]

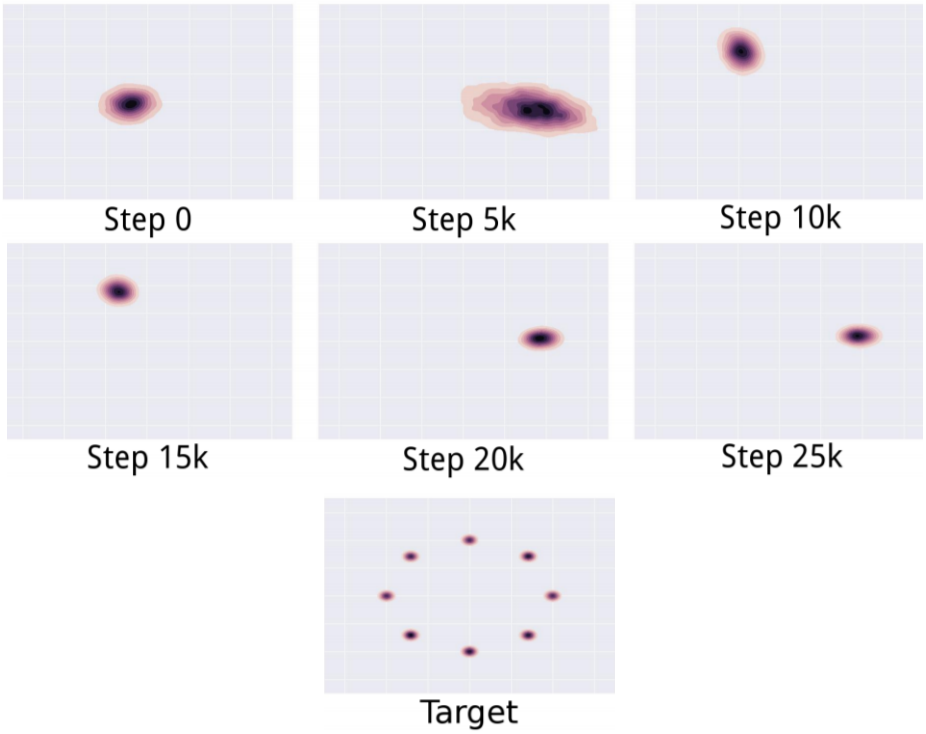
Tyto problémy mohou mít mnoho příčin, jako je nevhodný návrh architektur generátoru, resp. diskriminátoru, nebo nevhodná loss funkce a v neposlední řadě i nevhodnou volbou optimalizačního algoritmu.

Obtížné dosažení Nashovy rovnováhy
Trénování GAN modelů spočívá v nalezení Nashovy rovnováhy ve hře dvou hráčů (diskriminátoru a generátoru), kde se každý hráč snaží minimalizovat jeho loss funkci. Nashovou rovnováhou je pak dosažení bodu, kdy už se oba hráči nemohou více zlepšit.

Nalezení takového bodu je bohužel velice náročný problém, jelikož GAN modely používají nekonvexní loss funkce a zároveň mají mnoho parametrů (se spojitými hodnotami). S dosažením Nashovy rovnováhy pak souvisí další níže zmíněné problémy.

Mode collapse
Tento problém se objeví pokud generátor nalezne nějaký realistický výstup nebo resp. nějakou (malou) množinu realistických výstupů, tj. takové výstupy, které diskriminátor nedokáže rozlišit od těch skutečných, a začne je vytvářet stále dokola.

Nejlepší strategií diskriminátoru proti tomuto jevu by bylo odmítnutí takovýchto výstupů, avšak pokud diskriminátor uvízne v lokálním minimu a nedokáže nalézt (zvolit) takovou to strategii, je pak pro generátor velice snadné nalézt jiné realistické výstupy [27]. Z této situace se již diskriminátor už nikdy nemusí dostat.



Obrázek 2.3: Ukázka mode collapsu na multimodálním datasetu. [3]

Vanishing gradients
Při trénování GAN modelů mohou nastat dva problémy s přesností diskriminátoru [28].

V prvním případě může nastat situace, kdy má diskriminátor velmi malou přesnost při určovaní (rozlišování) pravosti dat, a tím pak loss funkce neodpovídá realitě a generátor pak kvůli tomu dostává špatnou zpětnou vazbu pro učení.

Může však nastat opačná situace a to že diskriminátor bude perfektní (bezchybný), tj. že je zaručeno, že pro všechna reálná data vrátí hodnotu 1

a pro všechna vygenerovaná (falešná) data vrátí hodnotu 0, tím ale nastane situace, že gradient loss funkce spadne k 0 a trénování se tím prakticky zastaví, resp. pokud je diskriminátor téměř perfektní tak se celý proces, kvůli nízkému gradientu, trénování značně zpomalí.

Konvergence
Diskriminátor má většinou vysokou (nebo nejvyšší) přesnost v začátcích trénování, kdy generátor vrací velice nekvalitní data. Avšak v průběhu trénování se generátor postupně zlepšuje a přesnost diskriminátoru tak začne postupně klesat, jelikož už diskriminátor dále nemůže lehce rozlišovat mezi reálnými a vygenerovanými daty.

Pokud je pak generátor perfektní, neboli generuje data z reálné distribuce dat, tak přesnost diskriminátoru bude 50 %. Tato situace však může způsobit problém s konvergencí GAN modelu jako celku, jelikož zpětná vazba od diskriminátoru sloužící generátoru pro učení (trénování) se bude s pokračujícím tréninkem zhoršovat. Tím se tedy může stát, že kvalita generátoru začne postupně klesat, tj. začne generovat data čím dál vzdálenější od těch reálných.

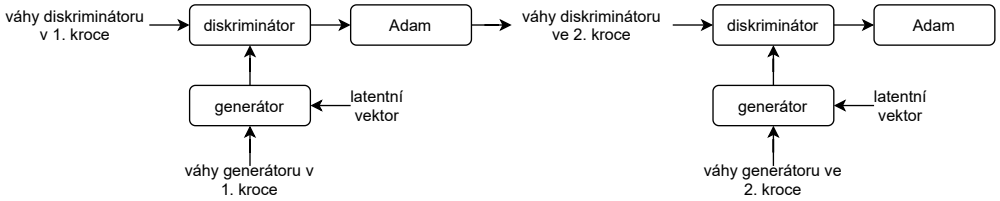
Konvergence v GAN modelu tedy nemusí být stálá, tj. stabilní dlouhodobý stav. [29]

2.3.6 Řešení známých problémů při trénování

Dosud vzniklo a dále vzniká mnoho výzkumů zaměřující se na řešení nebo zmírnění výše zmíněných problémů. Mnoho řešení se snažilo zlepšit současné optimalizační algoritmy (Adam, Sgd, ...) nebo vytvořit nové, dále pak je snaha nalézt vhodnější architekturu pro generátor, resp. diskriminátor, nebo vhodnější loss funkci. [26]

Řešení mode collapse
Mezi nejznámější pokusy o řešení problému mode collapse je použít Wasserstein loss funkci. Tato funkce zmírňuje mode collapse tím, že ve trénovacím procesu se diskriminátoru vyhne problém s vanishing gradients a pokud diskriminátor neuvázne v nějakém lokální minimu, tak se naučí odmítat výstupy od generátoru (vytvářená data), které generátor vytváří stále dokola, takže generátor je pak nucen svůj výstup změnit. [30]

Dalším známým řešením je pak použití Unrolled GAN, kde se generátor dívá (rozbaluje) k kroků dopředu, viz Obrázek 2.4, na to jak by se diskriminátor optimalizoval (učil, tj. měnil váhy) v dalších krocích. Generátor je trénován každých k kroků, kdežto diskriminátor je trénován pouze z prvního kroku. Tento model tedy tímto snižuje pravděpodobnost toho, že se generátor přetrénuje na konkrétní diskriminátor a tím tak zmírňuje mode collapse a zároveň zvyšuje stabilitu. [4]



Obrázek 2.4: Ukázka 2 kroků rozbalení pro Unrolled GAN. [4]

Řešení vanishing gradients
Stejně jako u řešení problému mode collapse, i zde může pomoci použití Wasserstein loss funkce s gradient penalty. Gradient penalty zde hraje klíčovou roli, jelikož bez ní (samotná loss funkce) může být model při nevhodně zvolených hyperparametrech (volba weight clippingu) náchylnější k vanishnig i exploding gradients. [3]

Řešení problému konvergence
Pro vyřešení problému s konvergencí, tj. její dosáhnutí, resp. udržení, se zkoumají různé regularizační techniky [31], jenž přidávají šum na vstupu diskriminátoru nebo penalizaci, jenž penalizuje váhy u diskriminátoru [32].

Problémy a jejich řešení:

- mode collapse
  - Wasserstein loss
  - Unrolled GAN
- vanishing gradients
  - Wasserstein loss s gradient penalty
- konvergence
  - regularizační techniky
  - penalizace vah diskriminátoru

2.4 DCGAN

DCGAN neboli Deep Convolutional GAN, se liší od běžných GAN modelů tím, že se pro implementaci diskriminátoru a generátoru použije CNN neboli Convolution Neural Network. Mnoho typů GAN modelů v dnešní době používá v diskriminátoru a generátoru CNN. Díky tomu se tedy jedná o jeden z nejrozšířenějších typů.

Tvůrci tohoto typu modelu ve své práci sepsali určitá doporučení, která vyplynula z jejich experimentů, jenž DCGAN dovede k lepší stabilitě při tréninku.

Jejich první doporučení je nahrazení všech pooling vrstev se strided convolutions vrstvami v diskriminátoru a v generátoru se fractional-strided convolutions vrstvami. Druhým doporučením je použití batchnorm vrstvy v generátoru i diskriminátoru. Dalším je odstranění fully connected hidden vrstev pro více hluboké architektury. Dále pak použití aktivační funkce ReLU ve všech vrstvách generátoru až na tu výstupní, kde by měla být použita Tanh. A posledním doporučením je použití aktivační funkce LeakyReLU ve všech vrstvách diskriminátoru. [7]

Loss funkce je zde stejná jako u standardní GAN 2.1.

Výhody:

- díky CNN je vhodnější pro nestrukturovaná data
- relativně jednoduchá architektura

Nevýhody:

- nestabilita
- nezohledňuje třídu, nemožnost ovlivnit generovaná data

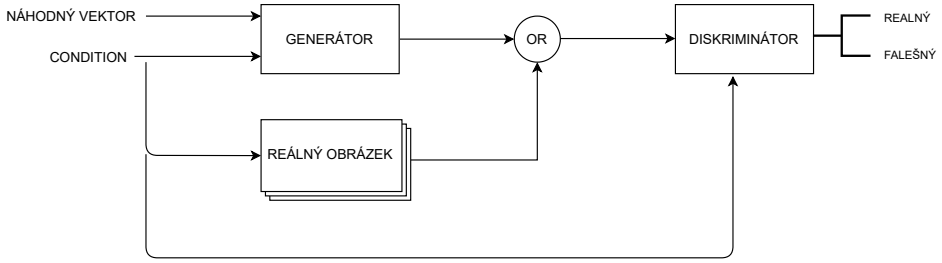
2.5 CGAN

CGAN neboli Conditional GAN je rozšířenou verzí GAN modelu, kde generátor i diskriminátor dostávají na vstupu navíc i informaci y (condition). Jako informace y je nejčastěji použita třída, který určuje typ výsledného vygenerovaného bodu v případě generátoru a k rozpoznání, jestli se jedná o pravý datový bod dané třídy v případě diskriminátoru. [2]

Avšak přidaná informace y může být i např. obrázek v případě modelu Pix2Pix, který na vstupu nahrazuje náhodný vektor. Pix2Pix se specializuje na transformaci obrázku z jedné domény do druhé.

V základní verzi CGAN se vstup generátoru a diskriminátoru skládá ze dvou částí. Tyto dvě části jsou náhodný vektor a třída. Existuje mnoho způsobů jak třídu začlenit do vstupu diskriminátoru a generátoru. Jedním z nich je použití embedding vrstvy následovaná plně propojenou vrstvu, jenž transformuje danou třídu do velikosti obrázku a tento obrázek je pak přidán jako další kanál do vstupního obrázku. [33]

Loss funkce je zde stejná jako u standardní GAN, tj. 2.1.



Obrázek 2.5: CGAN architektura. [2]

Výhody:

- kontrola nad generovanými daty
- vylepšená stabilita tréninku
- relativně jednoduchá architektura

Nevýhody:

- nutnost mít nebo dodat informaci y

2.6 Pix2Pix

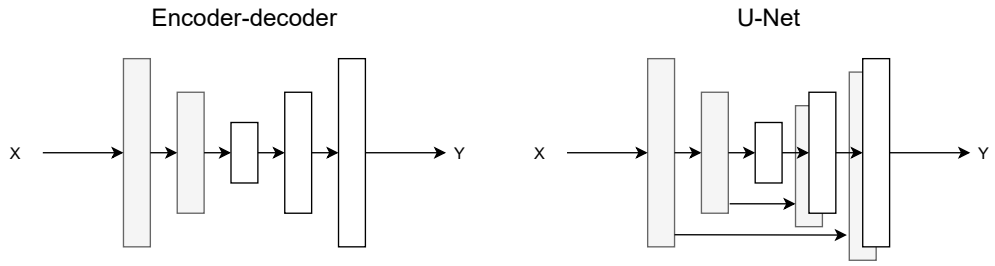
Tento typ je úspěšnou variantou CGAN zaměřující se úzce na obrázky. Model provádí image-to-image transformaci, kde se učí transformovat obrázek z jedné domény do druhé. Pix2Pix pak dokáže produkovat na rozdíl od klasických GAN modelů obrázky o vysokém rozlišení. [34]

Na vstupu generátor nedostává náhodný vektor, ale vstupní obrázek, dle kterého pak mapuje výsledný výstupní obrázek. Zdrojem náhodnosti jsou zde dropout vrstvy, které jsou použity při trénování a také při generování (predikci).

Diskriminátor na vstupu dostává pár obrázků, vstupních jenž jsou stejné jako u generátoru a výstupních což jsou ty reálné nebo vygenerované.

Dataset je složen z množiny párů (*vstupní obrázek*, *výstupní obrázek*). Vstupní a výstupní obrázky jsou odlišné, ale často mají podobné rysy, např. na vstupu může být obrázek s letní krajinou a na výstupu pak bude obrázek se zimní krajinou.

Architekturou generátoru je U-Net, což je fully convolutional architecture jenž vychází z encoder-decoder architektury. Obsahuje tzv. skip-connections, které jsou mezi encoding a decoding vrstvami, jak je znázorněno na Obrázku 2.6. Tyto skip-connections výsledně pomáhají k vygenerování obrázků s přesnějšími detaily. Náhodnost na vstupu generátoru je pak docílena dropout vrstvami, které jsou použity ve fázi tréninku, tak i ve fázi generování (predikce). [35]



Obrázek 2.6: Architektury Encoder-decoder a U-Net sítí. [2]

Jako diskriminátor je použit PatchGAN, který si rozdělí vstupní obrázek na části a u nich se snaží rozpoznat, zda se jedná o vygenerovaný nebo pravý obrázek, z průměrného výsledku z těchto částí se vytvoří finální rozhodnutí o pravosti původního vstupu.

Loss funkce diskriminátoru je stejná jako u původního GAN modelu, kdežto loss funkce generátoru se od té původní liší v tom, že k ní přičte navíc $\lambda L1$ loss funkci, jenž se snaží minimalizovat součet absolutních odchylek vygenerovaného obrázku od obrázku původního (po pixelech). Pomocí parametru λ se určuje důležitost funkce a její doporučené hodnoty jsou v intervalu od 10 do 100. Výsledně se pak díky funkci $L1$ docílí reálněji vypadajícího obrázku. [2]

Výhody:

- lze generovat obrázky s vyšším rozlišením
- stabilnější trénink

Nevýhody:

- vyžaduje množinu párů (vstupní obrázek, výstupní obrázek)

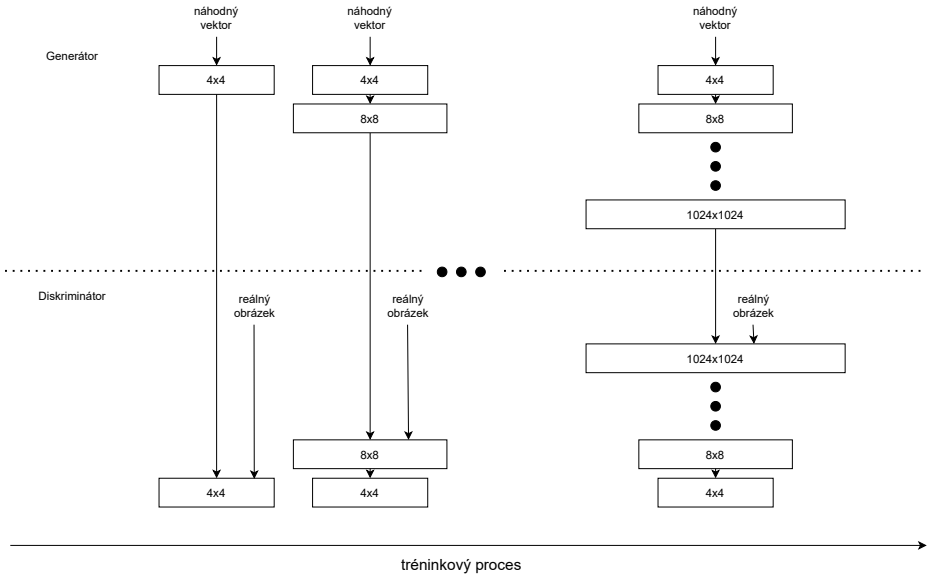
2.7 PGAN

PGAN neboli Progressive Growing of GAN model se stejně jako Pix2Pix úzce zaměřuje na obrázky. Klíčovou myšlenkou u tohoto typu je postupné progresivní vytváření generátoru a diskriminátoru. Nejprve se začne s pár vrstvami které začnou tvořit, respektive přijímat obrázky s nízkým rozlišením, až se síť naučí generovat tyto malé obrázky, tak se začnou přidávat další vrstvy, které generují větší obrázky, tento proces pokračuje do té chvíle než generované obrázky dosáhnou požadovaného rozlišení.

Generátor má na začátku trénování blok s convolutional vrstvou s výstupem o velikosti 4x4, jenž na vstupu dostane náhodný vektor. V další iteraci se přidá další blok na konec sítě, který má convolutional vrstvu s dvojnásobně větším výstupem, tj. 8x8.

Diskriminátor si bloky oproti generátoru tvoří opačně, tj. začíná s nejmenší vrstvou přijímající obrázky o velikosti 4x4 px. V další iteraci se přidá nový blok na začátek sítě, který má vrstvu přijímající obrázek o velikost 8x8 px.

Všechny právě existující vrstvy jsou trénovatelné v celém průběhu tréninku, tj. žádná z vrstev není zmražená.



Obrázek 2.7: PGAN architektura. [5]

Díky inkrementálnímu tréninku se nejprve zachytí obecné struktury obrázku a poté přejde pozornost na čím dál tím menší detaily, které se objeví s vyšším rozlišením. Výsledně se tak díky postupnému přidávání vrstev zrychlí trénink a zvýší stabilita celého modelu. [5]

PGAN používá Wasserstein loss funkci 2.6, jenž je velice rozdílná od té standardní použité v GAN modelu. Tato funkce je založená na Wasserstein vzdálenosti, která měří podobnost mezi dvěma distribucemi. Vzdálenost se spočítá pomocí formule $\sup_{\|f\|_L \leq 1} E_x [f(x)] - E_{\hat{x}} [f(\hat{x})]$, kde x a $\hat{x}$ jsou body ze vstupních distribucí, sup je nejmenší horní mez a f je 1-Lipschitz funkce pro kterou platí omezení $|f(x_1) - f(x_2)| \leq |x_1 - x_2|$. Wasserstein Loss pak vypadá následovně [18]:

$$E_{\hat{x}} [D(\hat{x})] - E_x [D(x)] \qquad [18] \quad (2.6)$$

WGAN je označení pro Wasserstein GAN, tj. GAN, která používá Wasserstein Loss. Pro využití této Wasserstein Loss funkce se musí mírně změnit architektura diskriminátoru tak aby se stal diskriminátor 1-Lipschitz funkcí. Aby se Diskriminátor ve WGAN stal 1-Lipschitz funkcí, tak musí vracet hodnoty z intervalu $[-\inf, \inf]$, tj. odstraňuje poslední aktivační funkci sigmoid, a

dále pak používá tzv. weight clipping, což je proces ve kterém se udržují váhy v určitém rozsahu $[-c, c]$. [5]

Diskriminátor se snaží vracet vyšší hodnoty pro skutečné obrázky a nižší hodnoty pro vygenerované obrázky, tj. snaží se maximalizovat výslednou loss funkci $D(x) - D(G(z))$ a generátor se snaží o maximalizaci $D(G(z))$. Výstup diskriminátoru tedy může být braný jako metrika reálnosti obrázku.

Výhodou WGAN, resp. využití Wasserstein loss funkce, je tedy lepší interpretovatelnost než u standardní loss funkce GAN modelu, a lepší stabilita tréninku.

Nevýhodou WGAN je použití weight clippingu pro splnění podmínký aby byl diskriminátor 1-Lipschitz funkcí. Kvalitu WGAN tedy velmi určuje volba hyperparametru c, jenž určuje interval ve kterém se mohou váhy nacházet. Pokud je c příliš velké, může to vést k pomalé konvergenci a naopak pokud je c příliš malé, muže dojít k problému s vanishing gradients. [36]

Řešením výše zmíněných problémů je použití WGAN-GP, což je WGAN, který na splnění Lipschitz podmínky (omezení) místo weigtht clippingu používá tzv. gradient penalty. Tento nový způsob je postavený na výroku, že diferencovatelná funkce f (diskriminátor) je 1-Lipschitz tehdy a jen tehdy když má pro každý vstup normu gradientu maximálně 1. WGAN-GP tedy ještě do Wasserstein loss přidává výraz $\lambda E_{\hat{x}} \left[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2 \right]$ viz 2.7, který penalizuje model tehdy, když se norma gradientu pohne pryč od jeho cílené hodnoty 1. Parametrem λ se určuje síla této penalizace a její doporučená hodnota je 10. [36][37]

$$E_{\hat{x}} \left[D(\hat{x}) \right] - E_x \left[D(x) \right] + \lambda E_{\hat{x}} \left[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2 \right] \qquad [37] \ (2.7)$$

Dále pak je nutné u tohoto rozšíření nepoužívat vrstvu batch normalization, jelikož vytváří korelaci mezi datovými body v batchi, což má neblahý účinek na efektivitu gradient penalty [36]. Jako náhradu autoři doporučují použít vrstvu layer normalization.

Výhody:

- lze generovat obrázky s vyšším rozlišením
- stabilnější trénink
- rychlejší trénink

Nevýhody:

- komplexní měnící se architektura

2.8 StyleGAN2ADA

StyleGAN2ADA je další z řady GAN modelů zaměřující se úzce na obrázky, který se snaží oproti svým předchůdcům zlepšit především architekturu generátoru. Tento typ vychází z PGAN modelu a díky své architektuře generátoru přidává možnost kontrolovat styl na různých úrovních detailu. V abstraktním pojetí se stylem označují nějaké charakteristiky popisující obrázek, příkladem takové charakteristiky na obrázku obličeje je např. barva kůže nebo tvar nosu. Dále pak stejně jako PGAN používá Wasserstein loss funkci s gradient penalty 2.7.

Generátor ke svému běhu ještě navíc používá mapovací síť, která má za úkol vytvořit tzv. intermediate vektor ze vstupního náhodného vektoru, jenž reprezentuje styl. Tato mapovací síť je MLP skládající se z 8 skrytých vrstev a jeho výstup poté putuje do samotného generátoru (syntetické sítě), respektive vstupuje do všech AdaIN komponent v současně vytvořených blocích. Dále pak do generátoru, respektive do AdaIN komponent, ještě vstupuje náhodný šum. [6]

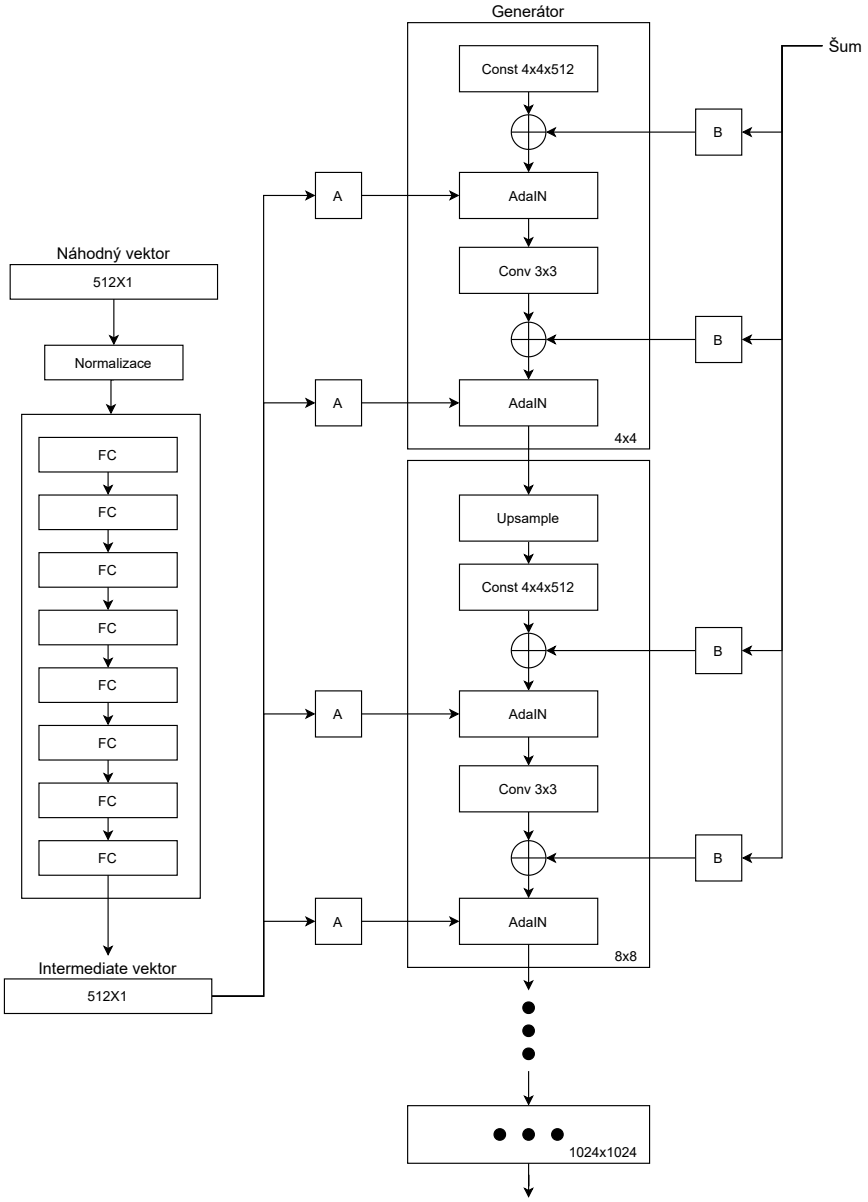
Vylepšením StyleGAN2ADA oproti předchůdci StyleGAN2 je přidaný (použitý) mechanismus adaptive discriminator augmentation, díky kterému je docílena lepší stabilita modelu i při malém množství trénovacích dat, což je velice důležitá vlastnost pro medicínská data, která se tímto problémem s nedostatkem dat vyznačují. Mechanismus spočívá v tom, že v průběhu procesu trénování diskriminátoru probíhá augmentace reálných a vygenerovaných obrázků s určitou pravděpodobností p, kde p se dynamicky mění podle míry overfittingu. [38]

Tento model pak nabízí možnost měnit styly (respektive vložit vlastní intermediate vektory do určitých bloků) daného obrázku ve 3 úrovních detailu: coarse, tj. obrázky od 4x4 až 8x8 px (tvar obličeje, postoj, ...), middle, tj. obrázky od 16x16 až 32x32 px (účes, ...) a fine, tj. obrázky od 64x64 až 1024x1024 px (barva očí, ...). Možná je i kombinace různých stylů, respektive vkládání různých intermediate vektorů do určitých bloků. [39]

Dále pak je možné použít šum pro vytváření podobných obrázků z jednoho intermediate vektoru. Šum stejně jak intermediate vektor se dá použít v různých úrovní detailu a intenzitou. Samotným šum se přidá přímo do obrázku, tj. šum s obrázkem mají stejné rozměry.

Výhody:

- stabilnější a rychlejší trénink
- možnost zvolit styl
- možnost kombinace stylů
- lze trénovat i s malým množstvím dat



Obrázek 2.8: Architektura StyleGAN2 generátoru. [6]

Nevýhody:

- komplexní měnící se architektura

Kapitola 3

Klasifikace dermoskopických obrázků

V této kapitole je popsána tvorba klasifikátoru dermoskopických obrázků, jenž bude použit v kapitole o augmentaci dat. V první části je nejprve popsán daný problém a základní informace o ISIC 2019. Dále tam pak jsou popsány vstupní data pro trénování, jako jsou základní statistiky, třídy a jiné detaily a jejich následné předzpracování.

Následující část se zabývá selekcí vhodné architektury a laděním hyperparametrů, dále pak se tato část věnuje samotné implementaci modelu, kde jsou detailněji popsány komponenty ze kterých se skládá výsledný kód a použité knihovny.

V závěru této kapitoly je popsáno trénování finálních modelů a souhrnná tabulka s dosaženými výsledky.

3.1 ISIC

International Skin Imaging Collaboration neboli ISIC je projekt, který vytváří (navrhuje) technologie, techniky a terminologie používané pro snímkovaní kůže s důrazem na soukromí pacientů, ale i možnost sdílení snímků skrze technologie a lékařské platformy. Dále pak ISIC vytváří veřejně přístupný archiv snímků kůže, který slouží k otestování, resp. validaci zmíněných technologií nebo technik. Tento archiv slouží také k vývoji a testovaní automatických diagnostických modelů nebo pomáhá lékařům ke studiu. [40]

ISIC již řadu let pořádá technologické soutěže (výzvy), které se zaměřují na tvorbu automatických klasifikačních, segmentačních a jiných modelů, jenž pracují se snímky kůže.

Úkolem soutěže ISIC 2019 [41] je rozpoznání nádorového onemocnění u daných dermoskopických obrázků. Jedná se tedy o klasifikační problém s 9 třídami, kde prvních 8 tříd reprezentují nějaké nádorové onemocnění a 9.

třída pak je neznámá, do které spadá např. kůže bez pigmentu nebo úplně jiné nádorové onemocnění. Pro účely této práce je však vytvořen model klasifikující pouze 8 známých nádorových onemocnění a 9. neznámá třída bude ignorována. Vytvářený klasifikátor budu dále označován jako DermKlasifikátor.

3.2 Vstupní data

Vstupní data jsou složena ze tří datasetů HAM10000, BCN20000 a MSK. Trénovací data obsahují 25331 obrázků a metadat s 8 třídami. Testovací data pak obsahují 8238 obrázků a metadat pro testování, jenž obsahuje data s 8 stejnými třídami jako u trénovacích dat a navíc obsahuje i 9. neznámou třídu. Metadata obsahují přidané informace k danému obrázku, jako jsou např. věk nebo pohlaví pacienta. Tyto metadata nejsou dostupná nebo nejsou kompletní pro všechny obrázky. Výsledný model bude na vstupu přijímat pouze obrázky a zmíněná metadata tak nebudou použita pro následné trénování.

Obrázky obsažené v tomto datasetu jsou velmi různorodé. Mají rozdílné rozlišení, viz seznamy níže, různou svítivost a barevnost. Další problém těchto dat je třídní nevyváženost zvýrazněná v grafu 3.1.

HAM10000 dataset:

- počet obrázků je 10015
- rozlišení obrázků je 600x450 px

BCN20000 dataset:

- počet obrázků je 19424
- rozlišení obrázků je 1024x1024 px

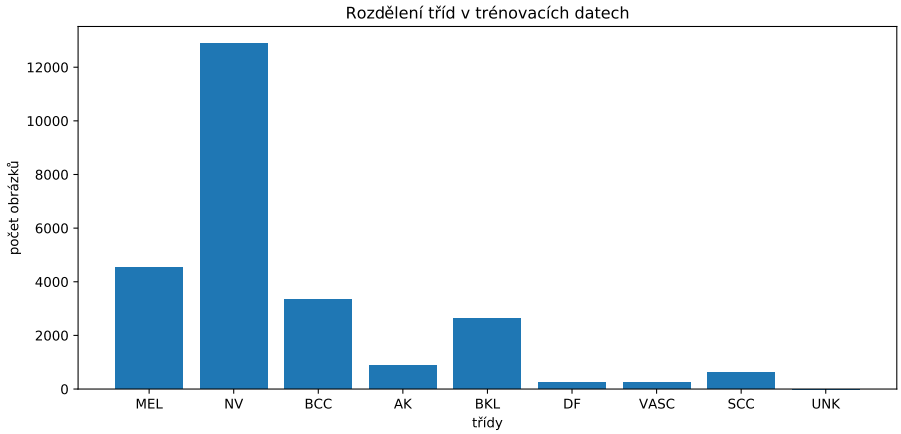
MSK dataset:

- počet obrázků je 19424
- rozlišení obrázků se pohybuje v rozmezí (576 až 1024)x(450 až 1024) px

3.2.1 Třídy

Trénovací data jsou těžce nevyvážená viz Obrázek 3.1 níže. 91,6 % dat jsou ze tříd MEL, NV, BCC nebo BKL, 8,4 % z dalších čtyřech tříd a 0 % dat z neznámé třídy UNK, která reprezentuje žádné nebo jiné nádorové onemocnění.

Pro zmírnění dopadu nevyváženosti tříd byla každé třídě přidělena váha $\frac{N}{N_i}$, kde N je celkový počet dat a N_i je počet dat ve třídě i, jenž bude využita v loss funkci categorical cross-entropy.



Obrázek 3.1: Rozdělení tříd v trénovacích datech

3.2.2 Předzpracování dat

Velikost celého trénovacího datasetu je 9 GB. Trénovací dataset byl předzpracován tak, aby měl výsledný dataset menší velikost a byly na něm provedeny nějaké základní transformace, díky kterým se ušetřil čas při trénování.

Dataset se rozdělil na 3 (velikostně) menší předzpracované datasety. Každý z těchto datasetů byl vytvořen tak, že se změnila velikost obrázků na fixní rozlíšení od 224x224 px (první dataset) po 260x260 px (čtvrtý dataset) se zachováním poměru stran. Nad všemi obrázky pak byla provedena normalizace barevnosti pomocí Shades of Gray algoritmu [42], jelikož se obrázky barevně hodně liší.

Díky změně velikosti obrázků tyto datasety mají velikost menší než 700 MB, což je oproti 9 GB výrazný rozdíl a tyto menší datasety se tak dají lépe přenášet na výsledný trénovací stroj (např. Gooogle Colab).

Při samotném tréninku byla online prováděna normalizace obrázků, tak aby měli pixely průměrnou hodnotu 1 a standardní odchylku 0.

3.3 Modely

Jako modely byly použity EfficientNets [43], což je skupina 8 modelů (od B0 do B7), které jsou svojí strukturou velmi podobné. V každém z jejich modelů dodržují pravidla pro škálování určující šířku, hloubku a rozlišení (velikost vstupních dat), která umožní efektivní trénování na různých rozlišení obrázků viz [44].

Nejmenší vstupní rozlišení dat pro modely je 224x224 px pro EfficientNetB0 a největší pak 260x260 px pro EfficientNetB2. Na konci každého výše

zmíněného modelu byla přidána neuronová síť Multilayer Perceptron neboli MLP.

3.4 Evaluace modelů

Pro ohodnocení modelů byly použity metriky accuracy a multi-class accuracy. Multi-class accuracy (mean sensitivity neboli průměrná senzitivita) je vypočítána jako průměr sensitivit (category-true-positives / category-positives) všech tříd, jenž by měla být dobrým indikátorem, zda se modelu daří klasifikovat rovnoměrně všechny třídy.

3.5 Rozdělení dat

Pro hledání vhodné architektury přidané MLP a vylepšování hyperparametrů byla z trénovacích dat náhodně vybrána menší podmnožina 1496 obrázků. Tato podmnožina byla dále rozdělena na trénovací a validační část obsahující 1201, resp. 295, obrázků, která slouží k ověření kvality architektury a hyperparametrů. Tyto obě části mají stejné rozdělení tříd jako původní kompletní trénovací data.

Na trénování výsledných modelů pak bylo využito 90 % z veškerých trénovacích dat a na validaci (ověření generalizace modelu), bylo využito zbylých 10 % dat.

3.6 Selekce architektury a hyperparametrů

Pro nalezení architektury a hyperparametrů bylo provedeno mnoho experimentů, kde byly vyzkoušeny různé kombinace proměnných a hledala se ta s nejlepší accuracy, popřípadě mean sensitivity. V každém experimentu byl model trénován po dobu 50 epoch. Experiment byl pak třikrát zopakován a byla u něho zvolena maximální dosažená accuracy popřípadě mean sensitivity.

3.6.1 Architektura přidané MLP

Pro nalezení vhodné architektury bylo vyzkoušeno několik kombinací proměnných. Tyto proměnné jsou počet skrytých vrstev a počet neuronů ve vrstvách. U každé kombinace se testovalo, jestli daná architektura dosahuje lepších výsledků než Baseline MLP.

Baseline MLP je složena z jedné skryté vrstvy s 1280 neurony a výstupní vrstvou o 8 neuronech s aktivační funkcí softmax. Ještě před výběrem architektury MLP byly provedeny dva experimenty, které měly zjistit zda je výhodnější vložit jako poslední vrstvu za pretrained síť vrstvu GlobalAveragePooling2D nebo vrstvu GlobalMaximumPooling2D. Z výsledků experimentu vyplynulo, že výhodnější je vložit GlobalAveragePooling2D.

mean sensitivity val	accuracy val	počet skrytých vrstev	počet neuronů ve vrstvě 1	počet neuronů ve vrstvě 2
0.340110	0.661017	1	1280	0
0.336542	0.681356	1	640	0
0.302479	0.661017	1	512	0
0.334969	0.670057	1	320	0
0.328761	0.680226	1	256	0
0.311826	0.681356	1	160	0
0.321610	0.672316	2	256	128
0.339825	0.680226	2	256	64
0.333390	0.672316	2	256	32

Tabulka 3.1: Výsledky experimentů pro hledaní vhodného počtu vrstev a neuronů pro MLP

Z výsledků 3.1 experimentů vyplynulo, že kombinace architektury se 2 skrytými vrstvami s 256 neurony a 64 neurony dosahuje nejlepší accuracy a mírně horší mean sensitivity než Baseline MLP, ale za to je méně komplexní, tj. má méně parametrů. Pro další experimenty tedy bude zvolena tato kombinace architektury.

Dále pak se testovalo, jestli se přidáním Dropout před první skrytou vrstvou MLP, tj. na poslední vrstvu pre-trained sítě, nebo přidáním Dropout na první skrytou vrstvou MLP zlepší generalizace modelu. Byly vyzkoušeny tyto kombinace:

mean sensitivity val	accuracy val	dropout parametr p na vstupu MLP	dropout parametr p na první HL
0.339825	0.680226	0.00	0.0
0.308311	0.685876	0.10	0.0
0.286563	0.684746	0.20	0.0
0.279475	0.674576	0.25	0.0
0.317255	0.693785	0.30	0.0
0.286696	0.673446	0.40	0.0
0.297365	0.679096	0.50	0.0
0.301821	0.663277	0.00	0.1
0.303000	0.693785	0.00	0.2
0.279113	0.679096	0.00	0.3
0.299093	0.657627	0.00	0.4
0.306575	0.681356	0.00	0.5

Tabulka 3.2: Výsledky experimentů pro hledaní vhodného umístění Dropout vrstev a jejich parametru p pro MLP

Z výsledků 3.2 experimentů vyplynulo, že některé kombinace Dropoutu zlepšily accuracy, ale razantně snížily mean sensitivity, takže Dropout ve výsledném modelu nebude použit.

3.6.2 Hledaní vhodného Learning Rate

Po nalezení vhodné architektury se hledala nejvhodnější kombinace proměnných pro learning rate. Testovalo se zda je vhodné použít linear decay nebo step decay a pokud step decay, tak jaký bude decay factor a kdy se použije.

mean sensitivity val	accuracy val	decay factor	drop every	linear decay
0.325843	0.681356	NaN	NaN	False
0.331353	0.724691	0.50	10.0	False
0.321731	0.688136	0.75	10.0	False
0.320379	0.692655	NaN	NaN	True
0.332854	0.700565	0.40	10.0	False
0.322101	0.692655	0.60	10.0	False

Tabulka 3.3: Výsledky experimentů pro hledaní vhodného learning rate pro MLP

Z výsledků 3.3 experimentů vyplynulo, že kombinace step decay prováděná každou 10. epochu s decay factor 0.5 pro snížení learning rate dává razantní zlepšení accuracy při mírném zhoršení mean sensitivity. Ve výsledném modelu tedy bude použita tato kombinace.

3.7 Implementace modelu

Implementace tohoto modelu, DermKlasifikátoru, je napsaná pomocí knihovny Tensorflow (verze 2.x) a nachází se v git repositáři na url adrese `https://gitlab.com/mpribyl/derm-classifier`. Celý kód je pak rozdělen do pěti komponent: train, dataset_utils, augment, preprocess_images a train_summary. Komponenta dataset_utils slouží k přípravě dat do požadované struktury pro klasifikátor, jako je např. přidání tříd do názvů souborů. Pomocí komponenty preprocess_images se data dále předzpracovávají, tj. provádí změnu rozlišení a normalizaci barevnosti. Pomocí kompenty augment se vytváří augmentované obrázky. A Poslední dvě komponenty train a train_summary slouží k trénování, resp. ke zpracování logů z trénování do grafů.

```
python preprocess_images.py \
--images_dir=./dataset/images \
--output=./datasets/isic/256_dataset_color_constancy \
--target_image_resolution=256 \
--with_color_constancy=true \
```

Listing 3.1: Ukázka použití komponenty preprocess_images.

3.8 Trénování finálních modelů

Z předchozího hledání byly zvoleny nejvhodnější hyperparametry do finálních modelů. Tyto modely pak byly trénovány nad celým datasetem. Celý trénovací dataset byl rozdělen do trénovací části, jež tvořila 90 % dat a do testovací části jenž tvořila 10 % dat. U finálních modelů byly vyzkoušeny různé pretrained sítě, jako je EfficientNetB0, EfficientNetB1 a EfficientNetB2, kde se

vyzkoušely různé velikosti rozlišení. Každý model byl trénován po dobu 30 epoch a po každé trénovací epoše byl uložen, aby mohl být v budoucnu použit pro případné další experimenty.

mean_sensitivity_val	accuracy_val	pre_trained_model
72.90 %	80.11 %	EfficientNetB0
70.36 %	81.92 %	EfficientNetB1
71.04 %	82.21 %	EfficientNetB2

Tabulka 3.4: Výsledky finálních modelů

Jak je vidět z Tabulky 3.4, nejlepší accuracy dosáhl model EfficientNetB2, avšak za cenu horší mean sensitivity, která je u tohoto typu problému stěžejní. Nejlepší mean sensitivity dosáhl EfficientNetB0 a dokáže tak nejlépe rozlišovat více druhů nádorových onemocnění a zároveň má nejméně trénovacích parametrů (má menší komplexitu). Jako pretrained síť tak bude použita EfficientNetB0.

Kapitola 4

Trénování GAN modelů

V této kapitole je popsáno trénování tří GAN modelů, jenž budou dále použity pro generování dermoskopických obrázků nádorových onemocnění.

Trénované modely proti sobě budou ohodnoceny pomocí kvantitativní metriky FID, která spolu s vizuální kontrolou obrázků bude určovat kvalitu vygenerovaných obrázků, a pomocí kvantitativní metriky Intra-FID, což je průměrná hodnota vypočítaných FID pro všechny třídy, která bude určovat výslednou diversitu generovaných obrázků.

Budou trénovány tři modely: CDCGAN, Pix2Pix a StyleGAN2ADA, kde CDCGAN zde bude použit jako baseline model. Všechny tři modely budou učeny na celém trénovacím datasetu ISIC 2019, tj. všechna data která používal výše zmíněný DermKlasifikátor.

Nejprve bude všem obrázkům změněno rozlišení vstupních obrázků na požadovanou velikost. Pro CDCGAN baseline bude obrázkům změněno rozlišení na 64 x 64 px, pro zmírnění dříve zmíněných problémů v jeho architektuře a nepříliš stabilnímu trénovacímu procesu. Pro zbylé modely Pix2pix a StyleGAN2ADA bude obrázkům změněno rozlišení na 256 x 256 px.

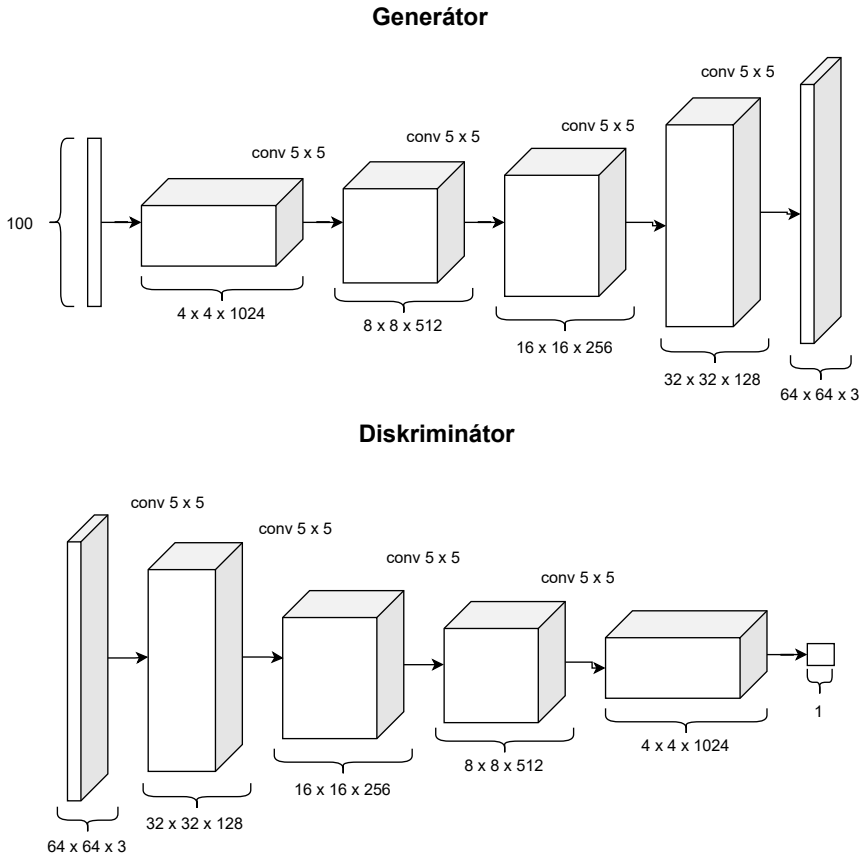
Nad všemi obrázky pak bude po změně rozlišení nejprve provedena normalizace barevnosti pomocí Shades of Gray algoritmu [42], tj. modely tak budou mít na vstupu stejně předzpracovaná data jako klasifikátor a díky samotné normalizaci barevnosti se pak modely budou více soustředit na samotné nádorové onemocnění a méně pak na okolní kůži a to by pak mělo vést modely (generátory) k tvorbě více diverzifikovaných dat.

4.1 CDCGAN

Jako první GAN bude trénován model CDCGAN, který bude následně použit jako baseline model. Architektura pro diskriminátor a generátor vychází z původní studie [45] o DCGAN viz Obrázek 4.1, jenž byla dále upravena.

Jako přidaná informace y je použita třída. Díky této přidané informaci y pak bude možné generovat určitou třídu (nádorové onemocnění) a zároveň se

tím urychlí trénovací proces, jelikož se nebude muset trénovat n GAN modelů, kde n je počet tříd, ale pouze jeden model. Dále pak by měl být trénink takového modelu stabilnější díky vyššímu množství trénovacích dat.



Obrázek 4.1: Archtiktura DCGAN [7]

4.1.1 Architektura generátoru

Vstupem generátoru je latentní vektor s dimenzí 100 spolu se třídou, jenž určuje typ nádorového onemocnění.

Latentní vektor je počátečními vrstvami nejprve transformován do výstupu o velikosti 4 x 4 x 1024. Ve stejné chvíli třída vstupuje do embedding vrstvy, poté do plně propojené vrstvy, která má již výstup o velikost 4 x 4 x 1. Tyto dva výstupy se spojí a vznikne tak společný vstup pro další vrstvy generátoru o velikosti 4 x 4 x 1025.

Dále následují čtyři bloky viz Obrázek 4.1. Blok obsahuje BatchNormalization vrstvu, která standardizuje následující vstup do další vrstvy po mini-batchích, což pomáhá stabilizovat proces trénování [46], další vrstvou v bloku

je LeakyRelu a dále následuje fractionally-stride convolutional (nebo také transpose convolutional) vrstva. Poslední blok pak zkonvertuje data z předchozích bloků do barevného obrázku s rozlišením 64 x 64 px.

4.1.2 Architektura diskriminátoru

Na vstupu diskriminátoru je barevný obrázek spolu se třídou.

Třída nejprve vstupuje do embedding vrstvy, poté do plně propojené vrstvy, která má výstup o velikost 64 x 64 x 1. Takovýto výstup je pak přidán do vstupního obrázku jako nový kanál. Vstupní obrázek pak pokračuje do dalších vrstev.

Dále následují, jako u architektury generátoru, čtyři bloky. Každý blok obsahuje LayerNormalization vrstvu, která je zde použita místo BatchNormalization vrstvy, jenž není pro WGAN-GP (v diskriminátoru) doporučena používat [36], dále následuje aktivační vrstva LeakyRelu a nakonec je použita convolutional vrstva. Výstupem posledního bloku je pak skóre určující míru reálnosti obrázku.

4.1.3 Loss funkce

Jako loss funkce byla použita Wasserstein funkce spolu s gradient penalty viz [36]. Pro použití této funkce musela být pozměněna architektura diskriminátoru, kde byly všechny vrstvy BatchNormalization nahrazeny za LayerNormalization a dále pak musela být odstraněna poslední aktivační vrstva sigmoid (aby vracel diskriminátor skóre a ne pravděpodobnost). Jako λ ve Wasserstein funkci 2.7 byla zvolena doporučená hodnota 10.

4.1.4 Předcházení známých GAN problémů

Pro předejití již známých problémů objevující se při trénování GAN modelů byly použity různé metody (doporučení), které by měly případné problémy snížit či úplně zrušit.

První doporučení, které bylo implementováno, je použití LeakyRelu aktivační funkce s parametrem α s hodnotou 0.3 pro diskriminátor a generátor, jenž by měli zrychlit konvergenci celého modelu [45].

Dále pak byly použity Batch Normalization vrsty v generátoru, které standardizují vstup z předchozí vrstvy, tj. výstup z této vrstvy pak má průměr 0 a rozptyl 1. Díky tomu by pak měl být trénovací proces více stabilní [47]. V diskriminátoru byly místo Batch Normalization použity Layer Normalization vrstvy, kvůli gradient penalty v loss funkci.

Před trénováním GAN modelu musejí být váhy diskriminátoru a generátoru inicializovány nějakým malým náhodným číslem. V modelu byly všechny váhy inicializovány pomocí Normální distribuce s průměrem 0 a rozptylem 0.02, podle článku [48] taková to inicializace pomáhá s problémem exploding a vanishing gradients.

4.1.5 Vstupní a výstupní data

Vstupní data jsou složena z výše zmíněného ISIC trénovacího datasetu obsahující 25331 obrázků s 8 třídami. Všechny vstupní data (obrázky) byla před trénováním barevně normalizována pomocí Shades of Gray algoritmu a poté jim bylo změněno rozlišení na 64 x 64 px. Výstupní data pak budou obrázky s rozlišením 64 x 64 px.

4.1.6 Implementace modelu

Implementace tohoto modelu se nachází v git repositáři na adrese `https://gitlab.com/mpribyl/dcgan`. Celý kód je rozdělen do čtyř komponent: train, evaluation, generate, a summary.

Zdrojový kód je psaný v jazyce Python v knihovně Tensorflow (ve verzi 2.x). Na logování průběhu trénování pak byla využita knihovna Tensorboard.

Komponenta train slouží pro trénování modelu a nachází se v ní tak implementace loss funkce, optimalizačních algoritmů, funkce na předzpracování dat a také implementace generátoru a diskriminátoru.

```
python train.py \
--images_dir ./datasets/64_dataset_color_constancy/ \
--log_dir ./logs \
--batch_size 64\
--epochs 30 \
--saved_models_dir ./saved_models \
--save_model_freq 1  \
--saved_checkpoints_dir ./saved_checkpoints
--generate_images_freq 132 \
--save_checkpoint_freq 132 \
--gen_lr 0.0001 \
--dis_lr 0.0001
```

Listing 4.1: Ukázka použití komponenty train.

Druhou komponentou je evaluation, která slouží k ohodnocení, tj. k výpočtu metrik FID a Intra FID jednotlivých modelů, avšak nejen tohoto modelu, ale i těch zbývajících (Pix2Pix a StyleGAN2ADA). Před výpočtem samotných metrik tato komponenta nejprve z daného datasetu (obrázků) musí získat statistiky (z Inception modelu) a poté již může vypočítat výsledné metriky.

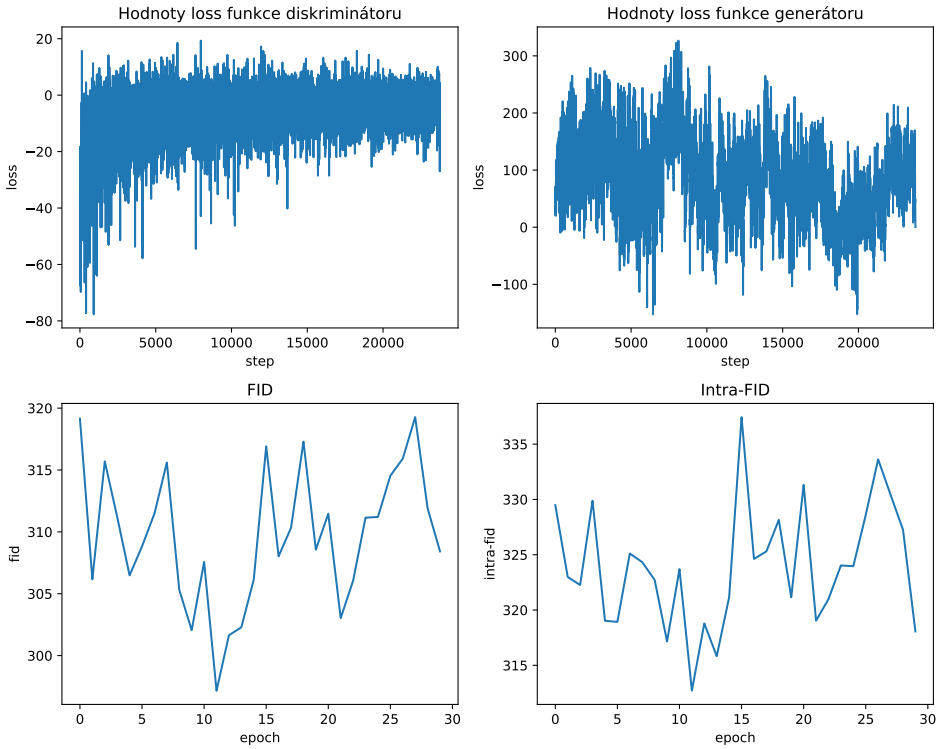
Třetí komponentou je generate, jenž slouží pro vygenerování obrázků pomocí již natrénovaných modelů CDCGAN, Pix2Pix a StyleGAN2ADA.

Poslední komponenta summary pak slouží k extrakci, analýze a transformaci dat z logů (z knihovny Tensorboard). Komponenta tak např. zjišťuje nejlepší dosažené hodnoty nebo vytváří souhrnné grafy.

4.1.7 Trénování - 1. experiment

V 1. experimentu byly zvoleny následující parametry pro model: batch size hodnota 32, learning rate pro Adam optimalizátory byly nastaveny na 0.0002 a dimenze latentního prostoru na 100. Celý tréninkový proces trval 30 epoch, ve kterých proběhlo 23760 učících se kroků.

V průběhu trénování byly ukládány vždy dva poslední modely (diskriminátor i generátor) spolu s aktuálním nastavením optimalizátorů pro případné obnovení tréninku, např. v případě výpadku výpočetního stroje (služby). Dále pak každou epochu byl ukládán aktuální generátor, resp. jeho váhy, pro následné použití, resp. evaluaci.



Obrázek 4.2: Shrnutí trénování modelu CDCGAN v 1. experimentu (pro batch size 32)

Ve shrnutí trénování na Obrázku 4.2 jsou vidět čtyři grafy. V horním řádku je zobrazen vývoj loss funkce diskriminátoru a generátoru pro každý učící se krok (step). V dolním řádku jsou zobrazeny vývoj metrik FID a Intra FID.

Diskriminátor se snaží o to, aby jeho loss funkce byla co nejnižší a generátor snaží mít co největší hodnoty loss funkce. U loss funkce diskriminátoru je tedy nejlepší dosaženou hodnotu -77, kterou dosáhl ve stepu 908, tj. v začátku samotného trénování, kde generátor produkoval velice nekvalitní data a diskri-

minátor pak měl velikou úspěšnost v rozlišování vygenerovaných a reálných obrázků. Loss funkce diskriminátoru se v průběhu trénovacího procesu dostala na stabilní hodnotu okolo -10.

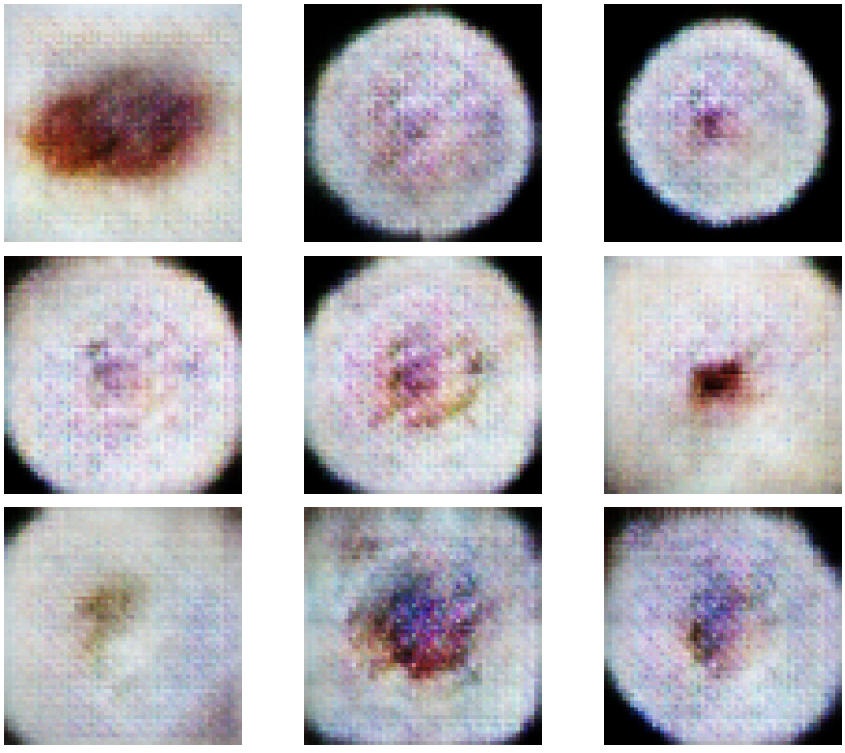
Nejlepší hodnotou loss funkce generátoru byla 326, dosažená v kroku 8122. Dále pak se hodnota loss funkce generátoru pohybovala v průměru okolo hodnoty 100, tj. hodnota byla celkem stabilní.

Jak je vidět na Obrázku 4.2, hodnota metriky FID měla v průběhu trénovaní klesající trend do 12. epochy, ve které dosáhla nejlepší hodnoty 297. Avšak od této epochy se hodnota FID postupně zhoršovala, až se dostala na hodnoty v průměru okolo 310.

Metrika Intra FID má podobný průběh jako FID, tj. její nejlepší hodnota 312 byla dosažena také v epoše 12.

V tomto experimentu byl tedy zvolen nejlepší model (generátor) z epochy 12, ve které dosáhl nejlepších hodnot metrik FID a Intra FID.

Na Obrázku 4.3 je zobrazen výstup generátoru z 12. epochy. Vizuální kvalita těchto obrázků není dostačující, jelikož jenom 2 obrázky z 9 vypadají aspoň trochu přesvědčivě. Na většině vygenerovaných obrázků jsou jasně vidět problémy, jako jsou opakující se artefakty a fialová barva rozprostírající se přes nádorové onemocnění a dokonce také přes kůži.



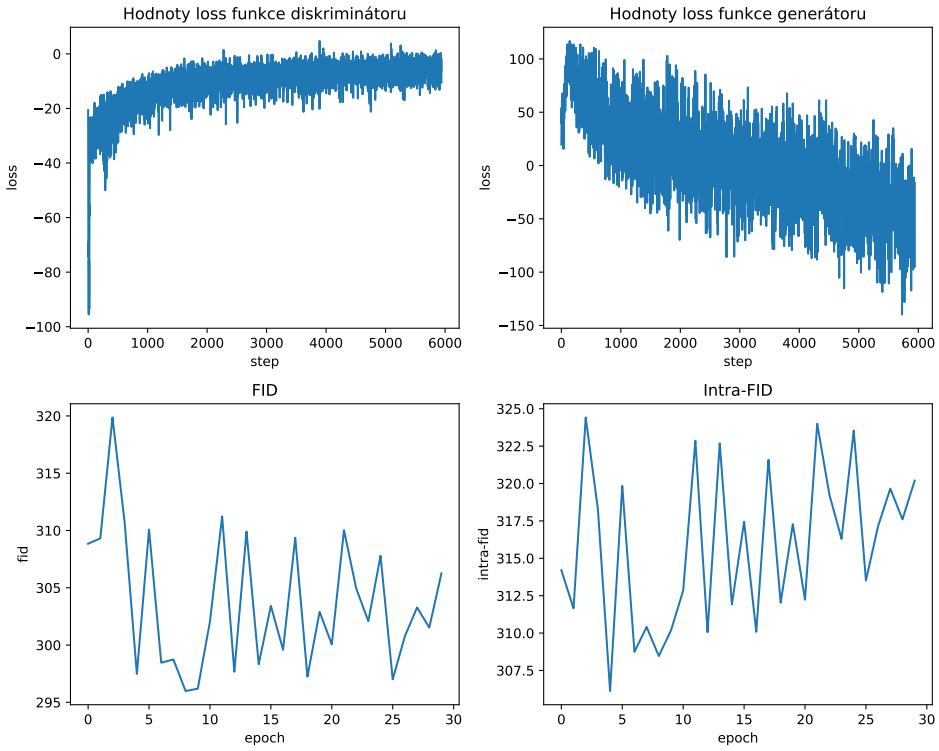
Obrázek 4.3: Výstup modelu CDCGAN z epochy 12 (z 1. experimentu)

4.1.8 Trénování - 2. experiment

Problém trénování z 1. experimentu byl ten, že hodnota metrik FID klesala k hodnotě 297, ale poté se model začal zhoršovat, až se se svou kvalitou vrátil téměř na začátek.

Ve 2. experimentu jsem se pokoušel zbavit problémů z 1. experimentu v hodnotách metrik FID a Intra FID a zároveň u nich dosáhnout lepších hodnot. V tomto experimentu jsem tedy hledal vhodnější parametry modelu.

Parametry modelu jsou stejné jako v předchozím experimentu, až na dále zmíněné věci. Pro dosažení lepší stability tréninku jsem zmenšil learning rate u diskriminátoru a generátoru z 0.0002 na 0.0001. Pro dosažení lepší kvality obrázku, resp. lepších hodnot metrik FID a Intra FID, jsem navýšil batch size z 32 na 64 a dále pak v dalším běhu trénovaní až na 128, což by podle této studie [49] mělo pomoci k lepší kvalitě i diverzitě generovaných obrázků tím, že s větším batch size se zachytí více tříd najednou a tím se poskytne modelu lepší zpětná vazba (vhodnější gradienty) pro učení. V tomto experimentu tedy proběhly 2 trénovací procesy. První pro batch size 64 a druhý pro batch size 128.



Obrázek 4.4: Shrnutí trénování modelu CDCGAN v 2. experimentu (pro batch size 128)

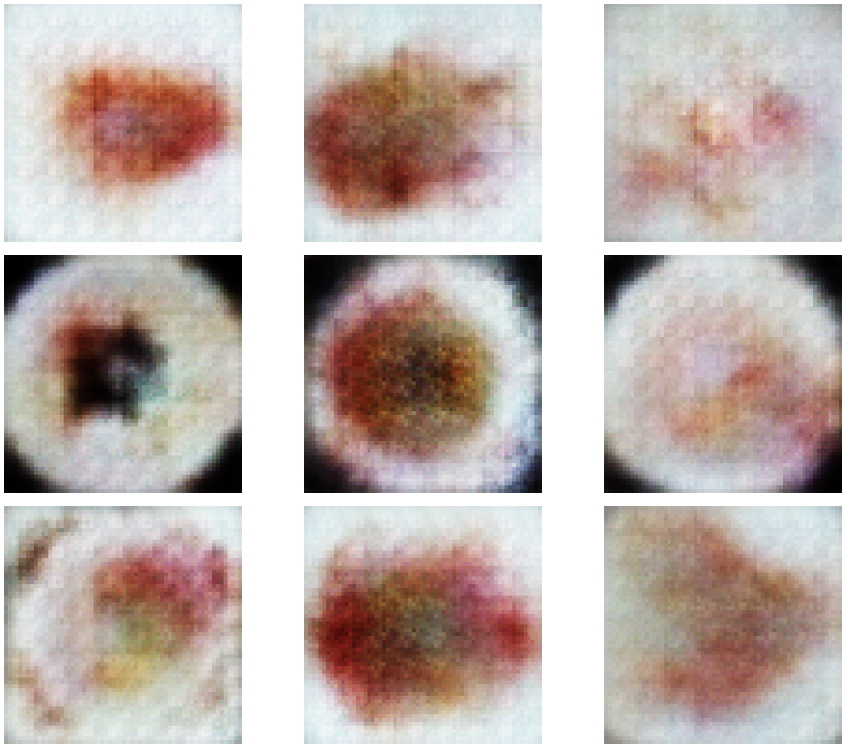
Použít batch size 128 se ukázalo jako nejvhodnější z obou trénovacích procesů. Tato konfigurace dosáhla lepších výsledků než batch size 64.

Loss funkce diskriminátoru u zvolené konfigurace se stejně jako u předchozího experimentu držela stabilně okolo hodnoty -10 a nejlepší dosažená hodnota byla -95, kterou dosáhl ve stepu 10, tj. také na začátku trénování.

Nejlepší hodnota loss funkce dosažená v generátoru byla 116 dosažená ve stepu 140. Dále pak loss generátoru postupně klesala až do konce trénování.

Z Obrázku 4.4 je vidět, že hodnota metriky FID měla v průběhu trénovaní klesající trend do epochy 9 s nejlepší dosaženou hodnotou 293, což je oproti předchozímu experimentu zlepšení o 4. Stejně jako v předchozím experimentu, FID pak začala stoupat, avšak v průměru se držela na hodnotách okolo 303, kdežto v 1. experimentu stále stoupala.

Metrika Intra FID klesala až do epochy 5 na hodnotu 306, což je oproti předchozímu experimentu zlepšení o 6, avšak hodnoty v dalších epochách měli stoupající trend. Generátor z epochy 9 bude brán jako finální model tohoto experimentu.



Obrázek 4.5: Výstup modelu CDCGAN z epochy 9 (z 2. experimentu)

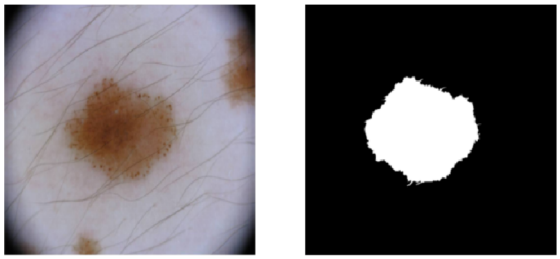
V tomto experimentu se ukázalo, že velikost batch size skutečně pomohla a trénovací proces s batch size 128 dosáhl lepších (nižších) a stabilnějších hodnot. Zároveň pak bylo dosaženo vizuálně kvalitnějších obrázků, na nichž

nejsou tak výrazné opakující se artefakty viz Obrázek 4.5. Tato konfigurace tak bude použita jako finální pro model CDCGAN.

4.2 Pix2Pix

Druhým modelem GAN, který byl trénován je Pix2Pix, jehož architektura vychází z původní práce [34] a při její implementaci nebyla udělána žádná změna.

Generátor zde na vstupu dostává obrázky, jenž slouží jako přidaná informace y. Výstupem generátoru jsou pak také obrázky. V tomto modelu byly jako vstupní obrázky použity segmentační masky, jenž označují na jakém místě se nachází nádorové onemocnění, popřípadě jaké, a kde nikoli, jako je to znázorněno na Obrázku 4.6, každý pixel v segmentační masce tedy označuje určitou třídu.



Obrázek 4.6: Ukázka výstupního a vstupního obrázku (segmentační masky) pro model Pix2Pix. [8]

Pro získání segmentačních masek pro obrázky z datasetu ISIC 2019 byl použit segmentační model BCDU-Net [8][50], který byl trénován na datasetu ISIC 2018.

4.2.1 Vstupní a výstupní data

Vstupní data jsou složena z 25331 párů (*vstupní obrázek*, *výstupní obrázek*), kde vstupním obrázkem je zde segmentační maska a výstupem je pak nějaký obrázek s nádorovým onemocněním.

Každá ze segmentačních masek obsahuje dvě hodnoty pixelů, tou první se určí, kde se nachází nádorové onemocnění a zároveň o jaký typ se jedná, konkrétní hodnota je pak z množiny $\{1, \ldots, 8\}$, a tou druhou hodnotu je pak 0, která se použije pro oblast, kde žádné nádorové onemocnění není, tj. např kůže.

Před trénováním je všem vstupním obrázkům změněno rozlišení na 256 x 256 px. Toto rozlišení pak budou mít i výstupní obrázky.

4.2.2 Implementace modelu

Implementace modelu Pix2Pix se nachází v git repositáři na adrese `https://gitlab.com/mpribyl/pix2pix` a implementace BCDU-Net pro získání segmentačních masek se nachází na adrese `https://github.com/rezazad68/BCDU-Net`.

Zdrojový kód modelu Pix2Pix je napsaný v jazyce Python v knihovně Tensorflow (ve verzi 2.x) a model BCDU-Net pak v knihovně Tensorflow (ve verzi 1.x). Pro logování je pak použita knihovna Tensorboard.

```
python train.py \
--input_images_dir "./dataset/input_images" \
--output_images_dir "./dataset/output_images" \
--batch_size 64 \
--epochs 30 \
--generate_images_freq 396 \
--save_model_freq 1 \
--save_checkpoint_freq 132 \
--log_dir "./logs" \
--saved_models_dir "./logs/saved_models" \
--saved_checkpoints_dir "./logs/saved_checkpoints"
```

Listing 4.2: Ukázka spuštění trénování modelu Pix2Pix.

4.2.3 Trénování

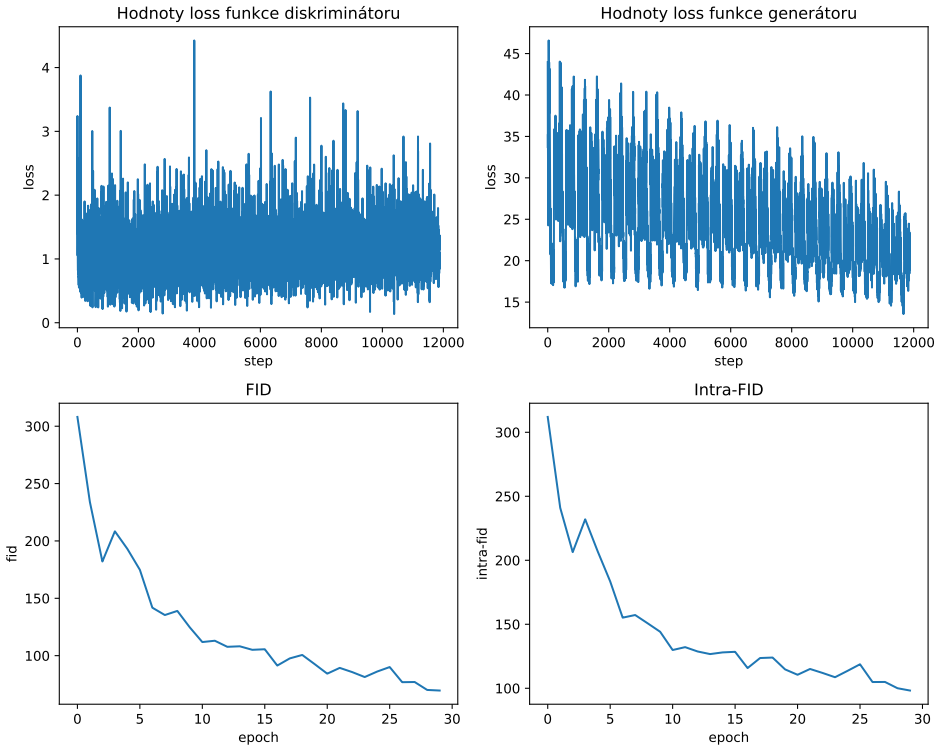
Pro trénování byly zvoleny následující parametry: batch size s hodnotou 64, learning rate diskriminátoru a generátoru nastaven na 0.0002. Celý tréninkový proces trval 30 epoch, jenž obsahoval 11880 učících kroků.

V průběhu trénování byly průběžně, po 396 krocích, ukládány výstupní obrázky generátoru. Po 132 krocích byl vždy uložen poslední model (diskriminátor a generátor) spolu s aktuálním nastavením (parametry, optimalizátorů, ...) pro případnou obnovu trénovacího procesu. Po každé epoše pak byl uložen generátor pro následné vyhodnocení a případně další použití.

Jak je vidět ze shrnutí tréninku, na Obrázku 4.7, loss funkce diskriminátoru je v průběhu celého tréninku stabilní a její hodnota se pak v průměru pohybuje okolo hodnoty 1 a nejnižší dosaženou hodnotou byla 0.13, která byla dosažena v 10386. kroku.

Celková loss funkce generátoru má po celou dobu klesající trend, kde klesne až na hodnotu 10 v kroku 11834. Tato loss funkce je složena ze dvou částí: běžnou soupeřící GAN loss a L1 loss posuzující věrnost rekonstrukce výstupního obrázku. Soupeřící GAN loss se držela stabilně okolo hodnoty 1.2 a L1 loss pak měla klesající trend.

Průběh hodnot metrik FID a Intra FID byl stabilně klesající, tj. generátor se v trénovacím procesu postupně učil generovat stále realističtější obrázky.



Obrázek 4.7: Shrnutí trénování modelu Pix2Pix

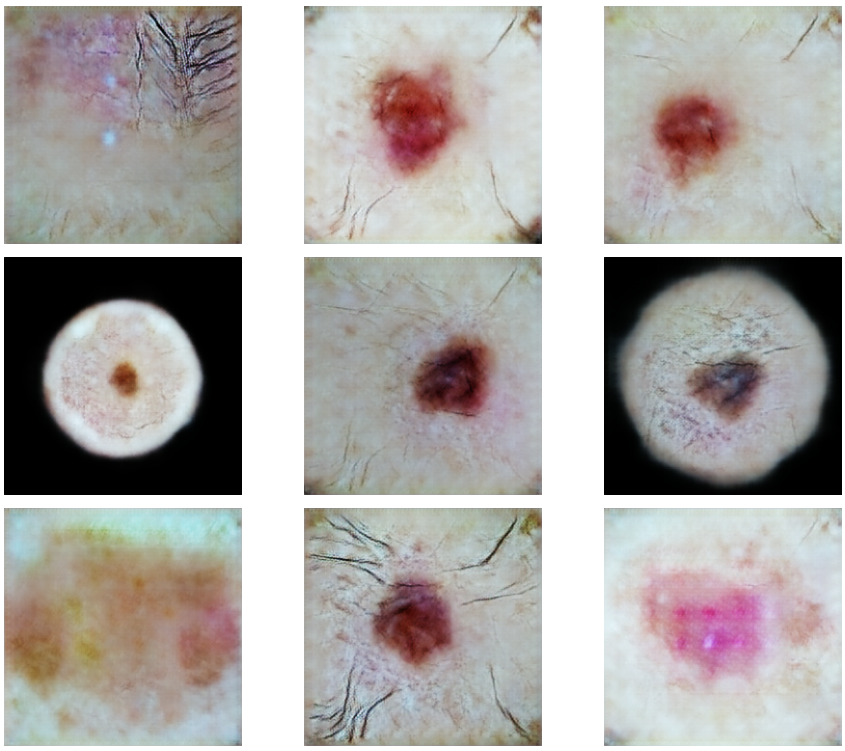
Nejlepší dosaženou hodnotou FID bylo 69 v epoše 30, stejně tak i v této poslední epoše bylo dosaženo nejlepší hodnoty metriky Intra FID 98.

Na Obrázku 4.8 jsou vidět obrázky generátoru z 30. epochy (o rozlišení 256 x 256 px) dosahující znatelně lepší kvality než obrázky (o rozlišení 64 x 64 px) z baseline modelu CDCGAN, na kterých byly viditelné opakující se artefakty.

4.3 StyleGAN2ADA

Posledním modelem, který byl trénován je StyleGAN2ADA [38] používající adaptivní mechanismus pro diskriminátor, díky kterému je docílena lepší stabilita modelu i při malém množství trénovacích dat.

Vstupem tohoto modelu jsou latentní vektory, které jsou následně poslány do mapovací sítě, která vytváří intermediate vektory, teprve ty se pak použijí jako vstup generátoru spolu s typem nádorového onemocnění (třída). Výstupem modelu jsou pak výsledné obrázky daného onemocnění.



Obrázek 4.8: Výstup modelu Pix2Pix z epochy 30

4.3.1 Vstupní a výstupní data

Trénovací vstupní data jsou stejná jako u baseline modelu, tj. jsou složena z 25331 obrázků z datasetu ISIC 2019 obsahující 8 tříd. Všem obrázkům byla před trénováním normalizovaná barevnost pomocí algoritmu Shades of Gray a následně jim bylo změněno rozlišení na 256 x 256 px. Výstupní obrázky generátoru pak mají stejné rozlišením jako ta vstupní, tj. 256 x 256 px.

4.3.2 Implementace modelu

Implementace se použila již stávající od výzkumné laboratoře Nvidia [51]. Výzkumníci z této laboratoře vytvořily dvě implementace tohoto modelu. Ta první byla vytvořena, resp. publikována, v 2. polovině roku 2020 a je napsaná v knihovně Tensorflow ve verzi 1.x (`https://github.com/NVlabs/stylegan2-ada`). Jejich druhá implementace, vytvořena počátkem roku 2021, je pak napsaná v knihovně PyTorch (`https://github.com/NVlabs/stylegan2-ada-pytorch`).

Aby se zachovalo stejné použití knihovny Tensorflow jako u modelů CDC-GAN a Pix2Pix, vybral jsem jejich první implementaci napsanou v Tensorflow.

Po prvním použití se však ukázalo, že při trénování dochází k úniku paměti a trénink tak vždy skončí po nějakém počtu epoch. Jelikož se s tímto problémem setkalo více lidí, existuje na jejich repositáři issue [52], které není ani 2. února 2021 stále vyřešené. Rozhodl jsem se tedy použít jejich druhou implementaci napsanou v PyTorch, ve které se tento problém nenachází.

```
python train.py \
--data=./dataset/isic2019/ \
--outdir=./logs \
--snap=2 \
--metrics=none \
--cond=true \
--kimg=800 \
--allow-tf32=true
```

Listing 4.3: Ukázka spuštění trénování modelu StyleGAN2ADA.

Jejich implementace, stejně jako u CDCGAN a Pix2Pix v průběhu tréninku pravidelně ukládá model pro obnovu nebo pro následné generování obrázků. Avšak pro použití těchto uložených modelů je potřeba mít v python path přístupné moduly torch_utils a dnnlib [53], které nejsou nikde veřejně publikované v žádném package manageru (pip, conda, ...) a tak je tedy nutné tyto moduly ručně přidat z jejich implementace, jako je to např. znázorněno v úryvku kódu 4.4.

```
import sys
sys.path.insert(0, "./stylegan2ada_project_pytorch")
```

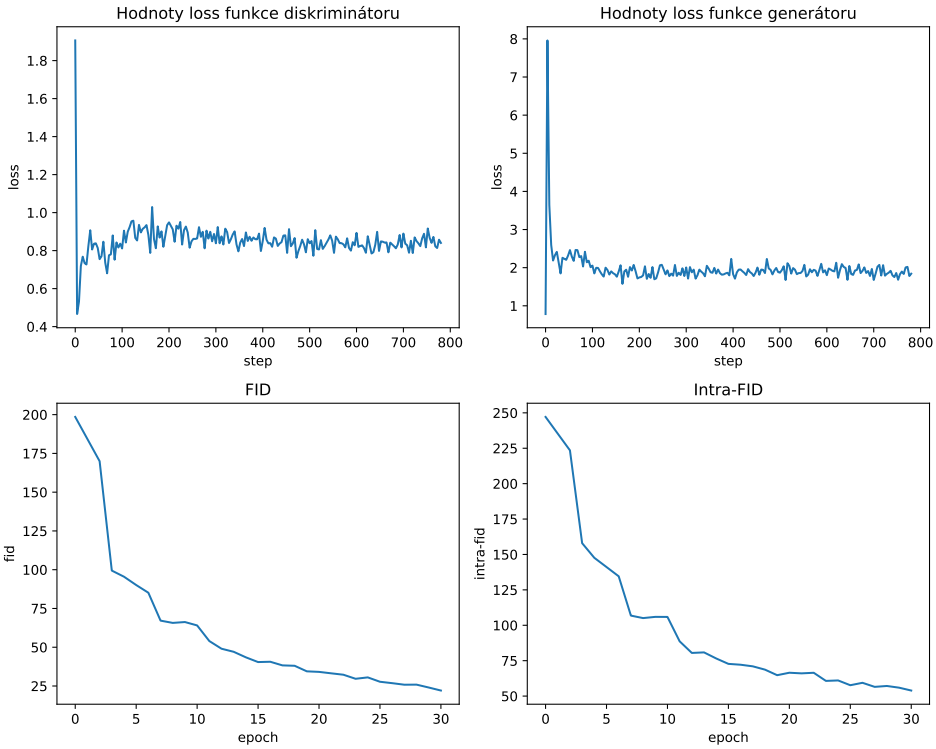
Listing 4.4: Ukázka zpřístupnění modulů torch_utils a dnnlib.

Logy jsou v průběhu trénování ukládány do souboru formátu `JSONL` a také pomocí Tensorboard jako `TFEvents`. Díky Tensorboard pak všechny logy budou moct být zpracovány stejným způsobem jako u předchozích modelů CDCGAN a Pix2Pix.

4.3.3 Trénování

Pro trénování byly zvoleny následující parametry: batch size s hodnotou 16, jenž byl zvolen automaticky, dle dostupného stroje, na kterém probíhal trénink, learning rate optimalizátorů generátoru i diskriminátoru byl nastaven na 0.0025 a dimenze latentního prostoru na 512, resp. velikost vstupních vektorů. Celý tréninkový proces trval 14 epoch, jenž obsahoval 22176 učících kroků. Stejně jako v předchozích modelech byly průběžně ukládány výstupní obrázky a modely pro budoucí ohodnocení, resp. použití.

Z Obrázku 4.9 jsou vidět loss funkce diskriminátoru a generátoru, jenž byly v průběhu trénování stabilní. Hodnota loss funkce disrkiminátoru se většinou pohybovala mezi 0.8 a 1 a hodnota loss funkce generátoru se pohybovala okolo hodnoty 2.

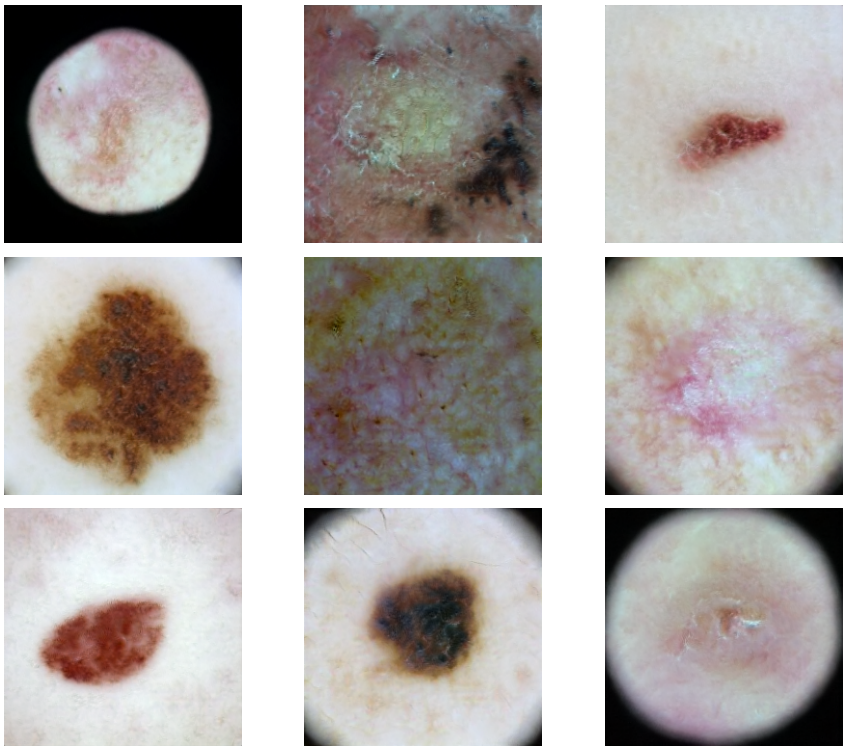


Obrázek 4.9: Shrnutí trénování modelu StyleGAN2ADA

Obě metriky FID i Intra FID mají klesající trend. Nejlepší dosaženou hodnotou FID bylo 22 v poslední epoše 30, stejně tak i v této poslední epoše bylo dosaženo nejlepší hodnoty metriky Intra FID 53.

Na Obrázku 4.10 jsou vidět výstupní obrázky z generátoru ze 30. epochy o rozlišení 256 x 256 px. Stejně jako u Pix2Pix se zde neobjevují žádné opakující se artefakty jako u CDCGAN.

Tento model, StyleGAN2ADA, dosáhl nejlepších hodnot metrik (FID, Intra FID) a vizuální kvality obrázků. Bude tedy použit jako finální pro následné experimentování z augmentací datasetu pro DermKlasifikátor.



Obrázek 4.10: Výstup modelu StyleGAN2ADA z epochy 30

Kapitola 5

Augmentace dat

V této části bude analyzováno jakým způsobem ovlivňuje klasická augmentace a augmentace dat pomocí GAN modelů přesnost klasifikačního modelu DermKlasifikátoru z kapitoly 3 pro daná nádorová onemocnění. Hlavním účelem této analýzy je zodpovězení otázek níže:

1. Jak efektivní je použití klasické augmentace dat na DermKlasifikátoru?

2. Jak efektivní je použití augmentace dat pomocí StyleGAN2ADA na DermKlasifikátoru?

3. Dokáže augmentace dat zvýšit sensitivitu u tříd s malým množstvím dat, tj. zda augmentace dat dokáže pro DermKlasifikátor zmírnit efekt těžce nevyváženého datasetu ISIC 2019?

5.1 Klasická augmentace dat

Nejprve byla testována klasická augmentace dat, kde se pro každou třídu T_i, kde i označuje danou třídu nádorového onemocnění viz Tabulka 5.1, vygenerovaly tři množiny nových augmentovaných dat z těch stávajících. Tyto množiny jsou dále označované jako M_j, kde $j \in \{1, 2, 3\}$.

Tyto množiny mají velikost N, kde $N \in \{500 * 8, 1000 * 8, 2000 * 8\}$, tj. pro každou třídu se vygenerovalo rovnoměrné množství dat. DermKlasifikátor pak byl trénován nad původními daty spolu s přidanými daty z jedné ze zmíněných množin. Pro každou množinu byl model trénován třikrát a výsledné dosažené metriky byly průměrovány. Postupně byly vyzkoušeny čím dál větší množiny, aby se zjistilo ideální množství potřebných dat ke zlepšení monitorovaných metrik accuracy a mean sensitivity.

Všechna data vygenerovaná pomocí klasické augmentace dat byla vytvořena pomocí následujícího postupu sestávajícího z několika kroků. V prvním kroku zmíněného postupu se provedl výběr náhodného vzorku dat (obrázků) z každé

T	MEL	NV	BCC	AK	BKL	DF	VASC	SCC
i	0	1	2	3	4	5	6	7

Tabulka 5.1: Číselné mapování tříd

třídy T_i o velikosti $N/8$. Pokud $N/8$ bylo větší než počet dat pro třídu T_i, pak byl vybrán náhodný vzorek dat s opakováním.

V dalším kroku proběhlo středové oříznutí obrázků, kde velikost vyříznuté oblasti byla vybrána náhodně z intervalu $[0.8, 1)$, kde hodnota z intervalu určovala procentuální velikost z původního obrázku, jenž bude oříznuta. Další kroky proběhly s 50% pravděpodobností. Mezi tyto kroky patří vertikální a horizontální přetočení obrázků, náhodná změna jasu o 0.1 a náhodná změna saturace také o 0.1.

V posledním kroku byly obrázky vráceny zpět do původního rozlišení, aby byly připravené pro vstupní vrstvu DermKlasifikátoru.

Pro uskutečnění tohoto postupu byla využita komponenta augment z implementace DermKlasifikátoru.

Z experimentů vyplynulo, že ideálním množstvím augmentovaných dat ke zlepšení monitorovaných metrik accuracy a mean sensitivity je 8*1000, tj. nejlepších výsledků se dosáhlo přidáním množiny M_2. S touto množinou bylo dosaženo nejlepší mean sensitivity 75.25 a druhou nejlepší accuracy 84.21 srovnatelnou s accuracy 84.10, dosaženou s množinou M_3 .

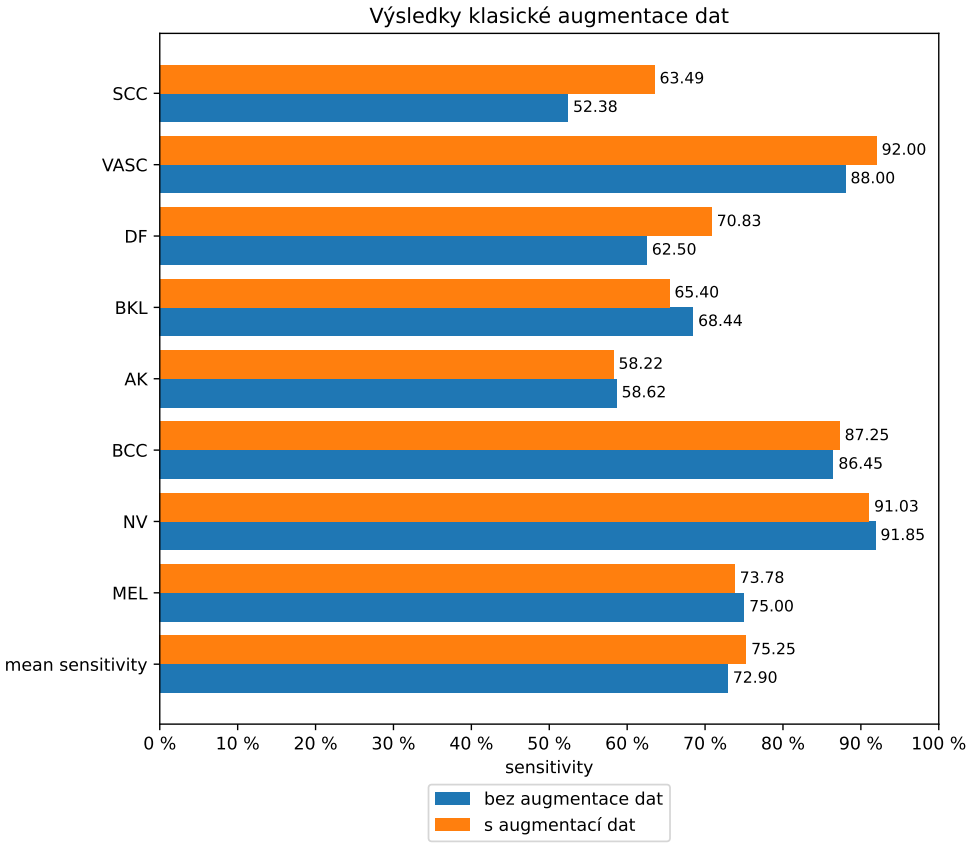
S pomocí augmentovaných dat z množiny M_2 bylo dosaženo zlepšení senzitivit především u tříd obsahující menší množství dat viz shrnutí na Obrázku 5.1, např. u třídy DF to bylo zlepšení o 8 %. Naopak u tříd které v datasetu byly nejvíce zastoupeny se senzitivita po přidání dat téměř nezměnila. Avšak u třídy BKL došlo po přidání dat k 3% zhoršení senzitivity.

5.2 Augmentace dat pomocí GAN modelů

Stejně jako u klasické augmentace byly vytvořeny tři množiny M_j, o velikostech N, kde $j \in \{1, 2, 3\}$ a $N \in \{500 * 8, 1000 * 8, 2000 * 8\}$. Množiny byly vytvořeny pomocí modelu StyleGAN2ADA. V každé množině byly data rovnoměrně rozloženy. tj. každý typ nádorového onemocnění tam měl $N/8$ zástupných obrázků. DermKlasifikátor pak byl třikrát trénován s původními a přidanými daty (pro každou množinu).

Na vytvoření zmíněných množin byla využita komponenta generate z modelu CDCGAN, která slouží ke generování obrázků, z již dříve trénovaných modelů StyleGAN2ADA a Pix2Pix. Všem vygenerovaným obrázkům poté bylo změněno rozlišení na 224 x 224 px.

Z provedených trénovaní vyplynulo, že stejně jako u klasické augmentace je ideálním množstvím dat, ke zlepšení monitorovaných metrik, 8*1000, tj. s množinou M_2 bylo dosaženo nejlepších výsledků accuracy hodnoty 75 a mean



Obrázek 5.1: Výsledky metriky senistivity u klasické augmentace dat

sensitivity 63.21, což je oproti žádné augmentaci zhoršení o 5.11 % u accuracy a 9.69 % u mean sensitivity .

StyleGAN2ADA se tak po 30 epochách tréninku nedokázal naučit generovat typy nádorových onemocnění, ať už v datasetu jsou nebo nejsou příliš zastoupeny, v dostatečné kvalitě. Pro vylepšení kvality vygenerovaných obrázků tak byl tento model trénován po dobu dalších 30 epoch, tj. od 30. do 60. epochy.

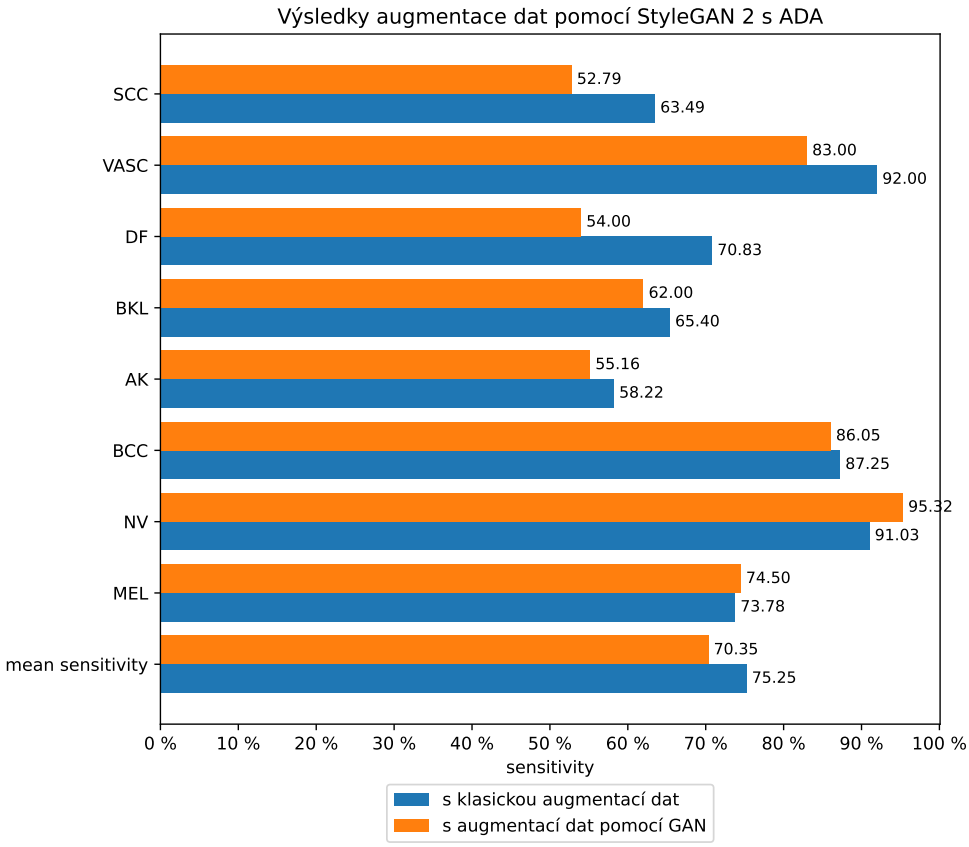
Po tréninku se podařilo zlepšit celkovou FID a Intra FID na hodnoty 12 a 51 z původních 22 a 53, jenž byly dosažena v 59. epoše. Hodnoty se zlepšily především u tříd, které obsahují mnoho dat. U tříd s malým množstvím dat zůstaly hodnoty stejné nebo se mírně zlepšily. S tímto modelem z 59. epochy byl zopakován výše zmíněný postup pro otestování augmentováných dat.

Po znovu provedení experimentu se ukázalo, že s množinou M_2 bylo dosaženo nejlepších výsledků accuracy hodnoty 82.31 a mean sensitivity 70.35, což je oproti klasické augmentaci zhoršení o 1.9 % u accuracy a o 4.9 % u mean sensitivity.

Po bližším porovnání s klasickou augmentací dat, viz shrnutí v grafu na Obrázku 5.2, je vidět, že dosažená sensitivita je značně horší u tříd DF, VASC a SCC, tj. u tříd s malým množstvím dat. Model u těchto tříd stále negeneruje příliš přesvědčivá data. U velké většiny tříd je však jen mírně horší oproti klasické augmentaci. Zlepšení se zaznamenalo u tříd NV a MEL, tj. u tříd s největším množstvím dat.

GAN model dosahoval u zmíněných tříd, kde dosáhl lepších sensitivit, hodnot FID 31 a 27. U těch tříd, s nejhoršími výsledky, model dosáhl hodnot FID z intervalu od 66 do 110. U ostatních tříd pak GAN model měl hodnoty FID v intervalu od 33 do 51. Z výsledků je tedy patrné, že metrika FID dokáže věrně určovat kvalitu výsledných obrázků a je tedy vypovídající.

V budoucích modelech, které budou pracovat s tímto datasetem ISIC 2019, by tedy měla být snaha dosáhnout hodnot FID aspoň 31 u každé ze tříd.



Obrázek 5.2: Výsledky metriky senistivity u augmentace dat pomocí StyleGAN2ADA

5.3 Porovnání výsledků augmentace s modelem DermGAN

V této části je shrnutí a porovnání dosažených výsledků modelu StyleGAN2ADA s DermGAN.

Z experimentů výše, ověřující efektivitu klasické augmentace dat a augmentaci pomocí StyleGAN2ADA, se ukázalo zlepšení sensitivity u DermKlasifikátoru ve dvou třídách s největším množstvím dat a zhoršení u těch s malým množstvím dat (oproti klasické augmentaci dat). Největší zlepšení sensitivit bylo zaznamenáno u tříd NV a MEL, které činilo 4.29 % a 0.72 %.

Výhodou tohoto modelu StyleGAN2ADA je především rychlost a stabilita trénovacího procesu. Další výhodou je pak možnost využití změny stylů při generování, pro větší kontrolu nad výslednými obrázky. Nabízí se tak možnost např. kontrolovaná změna barvy kůže a nebo kombinování obrázků ze stejné třídy. Nevýhodou tohoto modelu se ukázalo, že se nedokázal dostatečně naučit generovat třídy s menším množstvím dat (pod 300).

DermGAN je model vytvořený od skupiny ML4H [54], jehož architektura vychází z Pix2Pix. Na rozdíl od Pix2Pix modelu použitého v této práci dokáže model DermGAN měnit i barvu kůže okolo daného onemocnění. Tento model byl trénován s 40000 obyčejných obrázků kožních onemocnění (nejedná se o dermoskopické obrázky), obsahujících 26 tříd. Poté byl ohodnocen nad 24000 jinými obrázky. Všechny obrázky předzpracovaly a jejich finální rozlišení pak bylo 256 x 256 px. Tento dataset [55] však není veřejně přístupný.

Natrénovaný DermGAN následně otestovali na klasifikátoru MobileNet, jenž byl učen na původním datasetu, spolu s 20000 vygenerovanými obrázky. Z jejich experimentů vyplynulo, že u většiny tříd sensitivita zůstala téměř stejná nebo mírně horší při porovnání s výsledky klasifikátoru trénovaného pouze z původním datasetem. Avšak u tříd BCC a MEL (třídy s malým množstvím dat) dosáhly největšího zlepšení a to o 13.4 % u MEL a o 3 % u BCC.

Hlavní výhodou modelu DermGAN je, že dokáže generovat kvalitní obrázky u tříd s malým množstvím dat a při generování obrázků je možné zvolit barvu (odstín) kůže. Nevýhodou oproti StyleGAN2ADA modelu je, že u tříd s nejvyšším množstvím dat nedokáže generovat kvalitní (nebo diverzní) obrázky natolik, aby mohly být použity pro zlepšení klasifikátoru oproti klasické augmentaci dat. Další nevýhodou oproti StyleGAN2ADA je pomalost učení.

Závěr

První kapitola této práce se věnovala analýze problému, jenž obsahovala nejprve popis medicínských dat, jejich typy a problémy s nimi spojené. Dále zde byla rozebrána augmentace dat, kde byly popsány její dva typy: klasická augmentace a augmentace pomocí hlubokých generativních modelů. Poté se detailněji rozebrala problematika generativních modelů, kde bylo popsáno z jakých komponent se skládá, jak se trénuje, jaké problémy při tom mohou nastat a jak se ohodnocuje. Dále pak byly rozebrány moderní typy generativních modelů.

Ve druhé kapitole byla popsána tvorba klasifikačního modelu DermKlasifikátor, jenž dokáže rozlišovat 8 tříd nádorových onemocnění, tj. byl popsán použitý dataset ISIC 2019 a jakým způsobem byl předzpracován. Dále byl popsán proces výběru architektury a hyperparametrů. V závěru pak bylo shrnutí několika vybraných klasifikačních modelů, ve kterém byl zvolen jako nejlepší ten z pretrained sítí EfficientNetB0. Výstupem této části je tedy implementace modelu a váhy trénovaného modelu. Odkazy na tyto výstupy se nachází v příloze B.

Třetí kapitola byla věnována trénování moderních generativních modelů CDCGAN, který sloužil jako baseline model, Pix2Pix a StyleGAN2ADA. U každého modelu byla popsaná jeho architektura, použitá loss funkce, trénovací proces a jeho vstupní a výstupní data. Prvním trénovaným modelem byl CDCGAN, jenž dosáhl jako baseline nejhorších výsledků metrik FID a Intra FID, jelikož jeho obrázky obsahovaly opakující se artefakty a jejich věrohodnost tak nebyla příliš vysoká. Druhým trénovaným modelem byl Pix2Pix, jenž oproti baseline modelu dosáhl lepších výsledků metrik FID i Intra FID o 224, resp. o 208 a jeho generované obrázky již neměly problém s opakujícími se artefakty. Třetím trénovaným modelem byl StyleGAN2ADA, který dosáhl nejlepších výsledků FID i Intra FID 22 a 53, tj. tento model StyleGAN2ADA byl následně použit v další části při ohodnocovaní generovaných dat pomocí DermKlasifikátoru. Prvním výstupem této časti jsou implementace modelů CDCGAN a Pix2Pix. Dalším výstupem jsou pak finální natrénované modely

CDCGAN, Pix2Pix a StyleGAN2ADA, které dosáhly nejlepších výsledků. Odkazy k finálním modelům a implementaci jsou vypsány v příloze B.

V poslední kapitole byl porovnán efekt klasické augmentace a augmentace pomocí StyleGAN2ADA. Augmentovaná data byly postupně přidávána do původního datasetu, jenž byly následně použity na trénování DermKlasifikátoru. U klasifikátoru pak byla sledována především sensitivita dosažená u jednotlivých tříd.

V prvním experimentu se u klasické augmentace dat zjišťovalo vhodné množství dat, jenž se má přidat do původního datasetu, díky kterému bude docíleno nejlepších výsledků. Tím ideálním množstvím bylo 8000 neboli 1000 nových obrázků pro každou třídu. U porovnání výsledků dosažených sensitivit se ukázalo, že klasická augmentace dat dokázala zlepšit mean sensitivitu o 2.35 oproti použití pouze původního datasetu.

V dalším experimentu se zjišťovalo vhodné množství augmentovaných dat vytvořené pomocí StyleGAN2ADA. Stejně jako u klasické augmentace dat bylo nejlepším množstvím 8000. Avšak sensitivita téměř u všech tříd dosáhla horších výsledků oproti žádné augmentaci. Pro vylepšení kvality generovaných obrázků tak byl model StyleGAN2ADA dále trénován dalších 30 epoch (od 30. do 60. epochy).

Po trénování byl tento experiment opakován s již lepším výsledkem, kde se podařilo dosáhnout u dvou tříd s největším množstvím dat NV a MEL zlepšení sensitivit 4.29 % a 0.72 %, oproti klasické augmentaci dat. U tříd s malým množstvím dat pak tyto výsledky byly značně horší a u ostatních tříd byl výsledek srovnatelný.

Tyto dosažené výsledky StyleGAN2ADA poté byly porovnány s již existujícím modelem DermGAN [54]. S porovnání vyplynulo, že StyleGAN2ADA dokázal generovat kvalitnější obrázky u dvou tříd, které obsahovaly největší množství dat, resp. z jejich pomocí bylo dosaženo lepších sensitivit. Kdežto DermGAN byl úspěšný u dvou tříd, obsahující malé množství dat a lépe tak dokáže minimalizovat problém s nevyváženým datasetem.

V možném rozšíření této práce se tak dají tyto nebo další GAN modely kombinovat pro vygenerování realističtějších obrázků, které budou pomáhat klasifikačním modelům k dosažení lepších výsledků.

Literatura

[1] Prof. MUDr. Jana Hercogová, C.: Co je melanom? 2020. Dostupné z: https://www.dermanet.cz/cs/kozni-choroby/melanomy/co-je-melanom__s536x7215.html

[2] Kazeminia, S.; Baur, C.; Kuijper, A.; aj.: GANs for Medical Image Analysis. 2019, 1809.06222.

[3] Beyondminds: Gradient Penalty. 2020. Dostupné z: https://beyondminds.ai/blog/advances-in-generative-adversarial-networks-gans/

[4] Hui, J.: GAN — Unrolled GAN (how to reduce mode collapse). 2020. Dostupné z: https://jonathan-hui.medium.com/gan-unrolled-gan-how-to-reduce-mode-collapse-af5f2f7b51cd

[5] Karras, T.; Aila, T.; Laine, S.; aj.: Progressive Growing of GANs for Improved Quality, Stability, and Variation. 2018, 1710.10196.

[6] Karras, T.; Laine, S.; Aila, T.: A Style-Based Generator Architecture for Generative Adversarial Networks. 2019, 1812.04948.

[7] Sanjeevi, M.: Ch:14.1 Types of GAN's with Math. leden 2019. Dostupné z: https://medium.com/deep-math-machine-learning-ai/ch-14-1-types-of-gans-with-math-5b0dbc1a491d

[8] rezazad68: Bi-Directional ConvLSTM U-Net with Densely Connected Convolutions. 2020. Dostupné z: https://github.com/rezazad68/BCDU-Net

[9] NZ, D.: Dermoscopy. 2020. Dostupné z: https://dermnetnz.org/topics/dermoscopy/

[10] Soni, J.; Ansari, U.; Sharma, D.; aj.: Predictive data mining for medical diagnosis: An overview of heart disease prediction. *International Journal of Computer Applications*, ročník 17, č. 8, 2011: s. 43–48.

[11] Cios, K. J.; William Moore, G.: Uniqueness of medical data mining. *Artificial Intelligence in Medicine*, ročník 26, č. 1, 2002: s. 1–24, ISSN 0933-3657, doi:https://doi.org/10.1016/S0933-3657(02)00049-0, medical Data Mining and Knowledge Discovery. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0933365702000490

[12] Djib2011: Standard data augmentation techniques. únor 2020. Dostupné z: https://datascience.stackexchange.com/questions/65311/could-gans-be-used-to-augment-data

[13] Antoniou, A.; Storkey, A.; Edwards, H.: Data Augmentation Generative Adversarial Networks. 2018, 1711.04340.

[14] Frid-Adar, M.; Diamant, I.; Klang, E.; aj.: GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification. *Neurocomputing*, ročník 321, Dec 2018: str. 321–331, ISSN 0925-2312, doi:10.1016/j.neucom.2018.09.013. Dostupné z: http://dx.doi.org/10.1016/j.neucom.2018.09.013

[15] Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; aj.: Generative Adversarial Networks. 2014, 1406.2661.

[16] Odena, A.: Open Questions about Generative Adversarial Networks. *Distill*, 2019, doi:10.23915/distill.00018, https://distill.pub/2019/gan-open-problems.

[17] Candido, R.: Discriminative vs Generative Models. červenec 2020. Dostupné z: https://realpython.com/generative-adversarial-networks/

[18] Google: Loss Functions. červenec 2020. Dostupné z: https://developers.google.com/machine-learning/gan/loss

[19] Borji, A.: Pros and Cons of GAN Evaluation Measures. 2018, 1802.03446.

[20] Alqahtani, H.; Kavakli-Thorne, M.; Kumar Ahuja, D. G.: AN ANALYSIS OF EVALUATION METRICS OF GANS. 07 2019.

[21] Brownlee, J.: How to Explore the GAN Latent Space When Generating Faces. září 2019. Dostupné z: https://machinelearningmastery.com/how-to-interpolate-and-perform-vector-arithmetic-with-faces-using-a-generative-adversarial-network/

[22] Shmelkov, K.; Schmid, C.; Alahari, K.: How good is my GAN? 2018, 1807.09499.

[23] Mack, D.: A simple explanation of the Inception Score. březen 2019. Dostupné z: https://medium.com/octavian-ai/a-simple-explanation-of-the-inception-score-372dff6a8c7a

[24] Brownlee, J.: How to Implement the Frechet Inception Distance (FID) for Evaluating GANs. srpen 2019. Dostupné z: https://machinelearningmastery.com/how-to-implement-the-frechet-inception-distance-fid-from-scratch/

[25] DeVries, T.; Romero, A.; Pineda, L.; aj.: On the Evaluation of Conditional GANs. 2019, 1907.08175.

[26] Saxena, D.; Cao, J.: Generative Adversarial Networks (GANs): Challenges, Solutions, and Future Directions. 2020, 2005.00065.

[27] Google: Mode Collapse. 2020. Dostupné z: https://developers.google.com/machine-learning/gan/problems

[28] Weng, L.: Vanishing gradient. 2020. Dostupné z: https://lilianweng.github.io/lil-log/2017/08/20/from-GAN-to-WGAN.html

[29] Google: Convergence. 2020. Dostupné z: https://developers.google.com/machine-learning/gan/training

[30] Google: Mode Collapse. 2020. Dostupné z: https://developers.google.com/machine-learning/gan/problems#mode-collapse

[31] Arjovsky, M.; Bottou, L.: Towards Principled Methods for Training Generative Adversarial Networks. 2017, 1701.04862.

[32] Roth, K.; Lucchi, A.; Nowozin, S.; aj.: Stabilizing Training of Generative Adversarial Networks through Regularization. 2017, 1705.09367.

[33] Brownlee, J.: How to Develop a Conditional GAN (cGAN) From Scratch. 2020. Dostupné z: https://machinelearningmastery.com/how-to-develop-a-conditional-generative-adversarial-network-from-scratch/

[34] Isola, P.; Zhu, J.-Y.; Zhou, T.; aj.: Image-to-Image Translation with Conditional Adversarial Networks. 2018, 1611.07004.

[35] Brownlee, J.: How to Develop a Pix2Pix GAN for Image-to-Image Translation. 2020. Dostupné z: https://machinelearningmastery.com/how-to-develop-a-pix2pix-gan-for-image-to-image-translation/

[36] Gulrajani, I.; Ahmed, F.; Arjovsky, M.; aj.: Improved Training of Wasserstein GANs. 2017, 1704.00028.

[37] Wolf, S.: ProGAN: How NVIDIA Generated Images of Unprecedented Quality. prosinec 2018. Dostupné z: https://towardsdatascience.com/progan-how-nvidia-generated-images-of-unprecedented-quality-51c98ec2cbd2

[38] Karras, T.; Aittala, M.; Hellsten, J.; aj.: Training Generative Adversarial Networks with Limited Data. 2020, 2006.06676.

[39] Arasanipalai, A. U.: How to Generate Game of Thrones Characters Using StyleGAN. 2019. Dostupné z: https://nanonets.com/blog/stylegan-got/

[40] Collaboration, I. S. I.: Organization of the isic melanoma project. 2020. Dostupné z: https://www.isic-archive.com/#!/topWithHeader/tightContentTop/about/isicHistory

[41] Collaboration, I. S. I.: Skin Lesion Analysis Towards Melanoma Detection. 2020. Dostupné z: https://challenge2019.isic-archive.com/

[42] Gessert, N.; Nielsen, M.; Shaikh, M.; aj.: Skin Lesion Classification Using Ensembles of Multi-Resolution EfficientNets with Meta Data. 2019, 1910.03910.

[43] Tan, M.; Le, Q. V.: EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. 2020, 1905.11946.

[44] Mingxing Tan, S. S. E.; Quoc V. Le, G. A., Principal Scientist: EfficientNet: Improving Accuracy and Efficiency through AutoML and Model Scaling. 2020. Dostupné z: https://ai.googleblog.com/2019/05/efficientnet-improving-accuracy-and.html

[45] Radford, A.; Metz, L.; Chintala, S.: Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. 2016, 1511.06434.

[46] Brownlee, J.: A Gentle Introduction to Batch Normalization for Deep Neural Networks. 2020. Dostupné z: https://machinelearningmastery.com/batch-normalization-for-training-of-deep-neural-networks/

[47] Brownlee, J.: Use Batch Normalization. 2020. Dostupné z: https://machinelearningmastery.com/how-to-code-generative-adversarial-network-hacks/

[48] Dellinger, J.: Weight Initialization in Neural Networks: A Journey From the Basics to Kaiming. 2020. Dostupné z: https://towardsdatascience.com/weight-initialization-in-neural-networks-a-journey-from-the-basics-to-kaiming-954fb9b47c79

[49] Brock, A.; Donahue, J.; Simonyan, K.: Large Scale GAN Training for High Fidelity Natural Image Synthesis. 2019, 1809.11096.

[50] Azad, R.; Asadi-Aghbolaghi, M.; Fathy, M.; aj.: Bi-Directional ConvLSTM U-Net with Densley Connected Convolutions. 2019, 1909.00166.

[51] Nvidia: RESEARCH AT NVIDIA. 2020. Dostupné z: https://www.nvidia.com/en-us/research/

[52] aiXander: Memory Leak. 2020. Dostupné z: https://github.com/NVlabs/stylegan2-ada/issues/9

[53] Nvidia: Using networks from Python. 2020. Dostupné z: https://github.com/NVlabs/stylegan2-ada-pytorch

[54] Ghorbani, A.; Natarajan, V.; Coz, D.; aj.: DermGAN: Synthetic Generation of Clinical Skin Images with Pathology. 2019, 1911.08716.

[55] Liu, Y.; Jain, A.; Eng, C.; aj.: A deep learning system for differential diagnosis of skin diseases. *Nature Medicine*, ročník 26, č. 6, May 2020: str. 900–908, ISSN 1546-170X, doi:10.1038/s41591-020-0842-3. Dostupné z: http://dx.doi.org/10.1038/s41591-020-0842-3

Příloha **A**

Seznam použitých zkratek

GAN Generative Adversarial Network

ADA Adaptive Discriminative Augmentation

ISIC International Skin Imaging Collaboration

FID Frechet Inception Distance

Příloha B

Odkazy na finální modely a repositáře

DermKlasifikátor

- Odkaz na repositář: `https://gitlab.com/mpribyl/derm-classifier`
- Odkaz na váhy modelu natrénovaného bez augmentace dat: `https://drive.google.com/file/d/1OSqDF_CtZo3uYjjSJUi26x03i0VE8CN_/view?usp=sharing`

CDCGAN

- Odkaz na repositář: `https://gitlab.com/mpribyl/dcgan`
- Odkaz na model z 9. epochy: `https://drive.google.com/file/d/1tgdkezcUtAObq0GojXiV_yWTpxw0BZTA/view?usp=sharing`

Pix2Pix

- Odkaz na repositář: `https://gitlab.com/mpribyl/pix2pix`
- Odkaz na model z 30. epochy: `https://drive.google.com/file/d/1V--kluHcqlkfOrul53qFpnW1I6JrIzQf/view?usp=sharing`

StyleGAN2ADA

- Odkaz na repositář: `https://github.com/NVlabs/stylegan2-ada-pytorch`
- Odkaz na model z 59. epochy: `https://drive.google.com/file/d/132sLjQtp152tkZxCR4iqiwyssg-tJQHZ/view?usp=sharing`

Příloha C

Obsah přiloženého USB flash disku

- `readme.txt` ..... stručný popis obsahu USB flash disku
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `thesis.pdf` ..... text práce ve formátu PDF