Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

DIPLOMOVÁ PRÁCE

Plzeň, 2024 Bc. Jan Burian

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2023/2024

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Jan BURIAN**
Osobní číslo:	**A22N0106P**
Studijní program:	**N0714A150011 Kybernetika a řídicí technika**
Specializace:	**Umělá inteligence a automatizace**
Téma práce:	**Extrakce mezibuněčné hmoty z histologických mikroskopických snímků jater pomocí metod strojového učení**
Zadávající katedra:	**Katedra kybernetiky**

Zásady pro vypracování

1. Nastudujte metody přípravy a hodnocení histologických snímků. Seznamte se s aplikacemi strojového učení v histologických mikroskopických obrazech. Zvláště se zaměřte na metody filtrace, detekce a segmentace.
2. Navrhněte metodu pro filtraci histologického snímku s cílem potlačit buněčnou hmotu a zvýraznit strukturu mezibuněčné hmoty. Navrženou metodu naimplementujte.
3. Ověřte funkčnost navrženého řešení a vhodným způsobem jej vyhodnoťte. Identifikujte problematické vlastnosti navržené metody a diskutujte možnosti dalšího vývoje.

Rozsah diplomové práce: **40-50 stránek A4**
Rozsah grafických prací:
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

- Šonka, M., Hlaváč, V., & Boyle, R. (2014). Image Processing, Analysis, and Machine Vision Second Edition. Thomson-Engineering.
- Čihák, R. (2002). Anatomie 2. Praha: Grada Publishing.
- Liška, V., Baxa, J., Beneš, J., Brůha, J., Ferda, J., Hošek, P., Jansová, M., Jiřík, M., Jonášová, A., Králíčková, M., Křečková, J., Křen, J., Lobovský, L., Lukeš, V., Mírka, H., Pálek, R., Pešta, M., Pitule, P., Rohan, E., ... Vyčítal, O. (2016). Experimental surgery (V. Liška (ed.); 1st ed.). NAVA, s. r. o.
- Petr Ferczadi (2019). Učební text k předmětu histologická technika. SZŠ Plzeň.
- Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2021). Dive into Deep Learning.
- Polák, Š., Varga, I. & kol. (2010). Úvod do histológie a histologickej techniky. Univerzita Komenského Bratislava.
- VACEK, Zdeněk. Histologie a histologická technika. Brno: Institut pro další vzdělávání pracovníků ve zdravotnictví, 1995. ISBN 80-7013-201-9.

Vedoucí diplomové práce: **Ing. Miroslav Jiřík, Ph.D.**
Výzkumný program 1

Datum zadání diplomové práce: **2. října 2023**
Termín odevzdání diplomové práce: **20. května 2024**

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
L.S.

Doc. Ing. Miloš Železný, Ph.D.
děkan

Doc. Dr. Ing. Vlasta Radová
vedoucí katedry

V Plzni dne 2. října 2023

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

Extrakce mezibuněčné hmoty z histologických mikroskopických snímků jater pomocí metod strojového učení

Diplomová práce

Autor práce: Bc. Jan Burian

Vedoucí práce: Ing. Miroslav Jiřík, Ph.D.

Plzeň, 2024

Prohlášení

Předkládám tímto k posouzení a obhajobě diplomovou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne ....

Podpis autora

Poděkování

Na tomto místě bych rád vyjádřil upřímné poděkování:

- Ing. Miroslavu Jiříkovi, Ph.D.: za jeho odborné vedení mé diplomové práce, za cenné rady a připomínky, a za možnost častých konzultací, které mi pomohly lépe se orientovat v řešené problematice.
- RNDr. Vladimíře Moulisové, Ph.D.: za podrobný popis procesu získání biologických vzorků, který byl pro mou práci klíčový.

Anotace

Hlavním tématem této diplomové práce je navržení a následná implementace metody pro extrakci jaterní mezibuněčné hmoty (extracelulárního matrixu) z mikroskopických histologických snímků. Navrženou metodu lze rozdělit do několika na sebe navazujících kroků. Nejprve dojde k segmentaci buněčných jader, následně dojde k jejich filtraci (metoda inpaintingu) s využitím binární masky získané ze segmentace. V závěrečné části se obraz s filtrovanými buněčnými jádry opět segmentuje, tentokrát však již segmentujeme mezibuněčnou hmotu. Pro potřebu zpracování mikroskopických obrazů jsme vyvinuli modul v jazyce Python, který umožňuje dané paměťově náročné obrazy zpracovávat po jednotlivých dlaždicích. V závěru práce jsme porovnali výsledky naší metodu s metodou založenou na generativních neuronových sítích, tzv. GANech a ověřili funkčnost navrženého řešení.

Klíčová slova: umělá inteligence, lékařství, mezibuněčná hmota, histologie, filtrace, segmentace

Annotation

The main aim of this thesis is to design and implement a method for the extraction of extracellular matrix from microscopic histological images, also called whole slide images. The proposed method can be divided into several consecutive steps. First, the cell nuclei are segmented, then they are filtered (inpainting method) using the binary mask already obtained from the segmentation step. In the final part, the image with filtered cell nuclei is segmented again, but this time we segment the extracellular matrix. Because of the high memory requirements of microscopic images, we have developed a Python module that allows the given image to be processed one tile at a time. At the end, we focused on a comparison of the results of our method with a method based on generative neural networks, the so-called GAN and verified the functionality of the proposed method.

Keywords: artifical intelligence, medicine, extracellular matrix, histology, filtration, segmentation

Obsah

Úvod

Téma této diplomové práce vyvstalo na základě spolupráce mezi Fakultou aplikovaných věd a Biomedicínským centrem v Plzni, které je součástí Lékařské fakulty UK. Plzeňské biomedicínské centrum se zaměřuje především na studium regenerace jaterní tkáně. Hlavním cílem je výzkum možností léčby pacientů s nádorovým onemocněním jater, a to s využitím stimulace regenerace zdravé jaterní tkáně poškozené např. onkologickou léčbou. Samotný výzkum probíhá s využitím experimentálních modelů na velkých zvířatech, zejména na přeštických černostrakatých prasatech. Jednotlivé experimentální modely mohou reprezentovat např. steatohepatitidy asociované s chemoterapií, toxické alkoholické postižení jater, biliární cirhózu, apod. Prasečí a lidská játra sice vypadají na první pohled rozdílně, jelikož lidská játra se skládají ze 2 laloků a játra již zmíněných přeštických prasat mají 4 až 6 laloků. Avšak z pohledu struktury jsou si vzájemně velice podobná.

Je-li pacient postižen např. onkologickým onemocněním jater, existují v zásadě dvě řešení vedoucí k vyléčení. Prvním řešením je provedení chirurgického zákroku, během kterého dojde k odstranění postižené části jaterní tkáně, tzv. resekce. Tímto způsobem je možné odstranit až 70 % jaterní tkáně. Tento chirurgický zákrok je možné provést díky schopnosti regenerace jaterní tkáně. Obecně je však operovatelnost závislá na velikosti a umístění nádoru. Druhým řešením je transplantace jater od lidského dárce. Lidských dárců je ovšem v dnešní době nedostatek a právě z tohoto důvodu se v Biomedicínském centru rozhodli realizovat již zmíněný výzkum. Vize výzkumu spočívá v tom, že se do prasečího decelularizovaného jaterního skeletu (scaffoldu) umístí pacientovy zdravé jaterní buňky. Tímto způsobem by tedy teoreticky bylo možné pacientovi „vypěstovat“ játra nová. Během tohoto procesu je třeba sledovat, zda dochází ke správnému osídlení scaffoldu pacientovými jaterními buňkami.

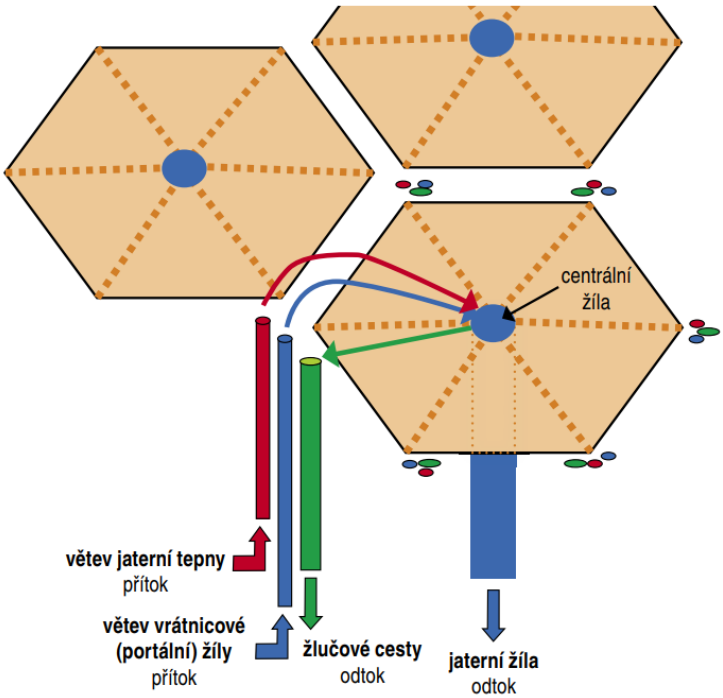
Hlavním cílem této diplomové práce je vytvoření a implementace metody pro filtraci histologického snímku s cílem potlačit buněčnou hmotu (extracelulární matrix) a zvýraznit strukturu v mikroskopických histologických obrazech. Hlavní podstatou je aplikace technologií umělé inteligence, zejména strojového učení, do lékařství. Jinými slovy, chceme na základě vstupního mikroskopického histologického obrazu, v našem případě zejména obarveném pomocí kitu na barvení retikulinových vláken získat co nejvěrnější nabarvený scaffold.

1. Játra jako orgán

Játra [1, 2] jsou součástí trávicí soustavy. Jedná se o největší žlázu v lidském těle (u dospělého člověka váží kolem 1,5 kg). Jsou umístěny v pravé brániční klenbě. Játra jsou tvořena, v případě člověka, dvěma laloky - větším pravým lalokem a menším levým lalokem. Skládají se z podlouhlých vícehranných lalůčků, jejichž velikost je v rozmezí od 1 do 2 mm. Lalůčky jsou tvořeny jaterními buňkami tzv. hepatocyty. Obecně je jaterní tkáň tvořena řadou buněk, jejichž počet se odhaduje až na několik stovek miliard. Z buněk jsou nejvíce zastoupeny hepatocyty (60-70 % buněčné hmoty jater).

Dále se v jaterní tkáni nachází cholangiocyty, což jsou epitelové buňky žlučových vývodů, endoteliální buňky či Kupferovy buňky. Poslední dvě zmíněné skupiny buněk však tvoří pouze jednotky procent buněčné hmoty. Zásobování jater probíhá přívodem okysličené krve do jednotlivých lalůčků (obr. 1.1) prostřednictvím větví jaterní tepny a vrátnicové žíly. Větvemi jaterní tepnou projde cca 25 % objemu přiváděné krve, naopak vrátnicovými žílami projde cca 75 % objemu přiváděné krve. Odkysličená krev je poté odváděna do centrální žíly jaterního lalůčku, jež se nachází ve střední části lalůčku.

Naopak ve směru proti toku krve proudí jaterním lalůčkem ve žlučových kanálcích žluč, která byla vytvořena jaterními buňkami. Žluč je následně svedena až do žlučníku. Mimo jiné má jaterní tkáň schopnost regenerace. Po resekci 50–60 % jaterní tkáně dorostou lidská játra do původní předoperační velikosti během několika měsíců. Přesný mechanizmus, však nebyl do dnešní doby zcela objasněn.



Obrázek 1.1: Schéma jaterních lalůčků. Převzato z [2].

1.1 Funkce jater

Játra [1] lze považovat jako jeden z nejvýznamnějších a nejvýkonnějších metabolických orgánů v lidském těle [2]. Účastní se metabolismu, což je látková a energetická přeměna, sacharidů, lipidů, proteinů i látek, obsahujících dusík. Pomáhají ukládat nadbytečnou glukózu ve formě glykogenu. Probíhá zde tvorba tuků ze sacharidů či dochází k převodu nadbytečného dusíku na močovinu. Rovněž se v nich vytváří žluč, která je potřebná pro trávení a vstřebávání lipidů, stejně jako vitamínů rozpustných v tucích. Dále jsou v játrech uchovány vitamíny rozpustné v tucích (A, D, K) a vitamín B_{12}. Játra rovněž disponují detoxikační funkcí. Během procesu detoxikace dochází ke zbavování se škodlivých látek (toxinů). Mimo jiné játra slouží jako rezervoár krve, k tvorbě velkého množství tělesného tepla či k syntéze látek potřebných pro normální srážlivost krve (např. protrombin).

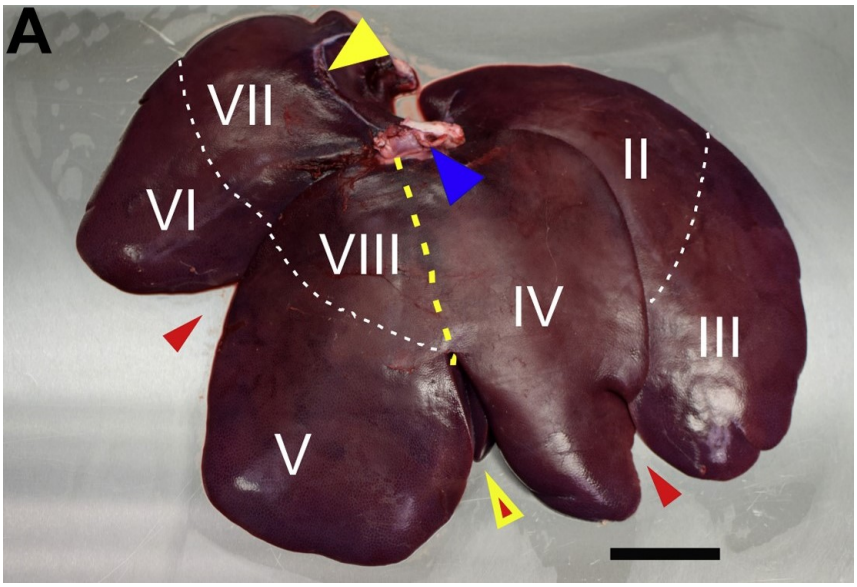
1.2 Onemocnění jater

Mezi nejčastější onemocnění jater patří hepatitidy či žlučové kameny [1]. Hepatitidy rozdělujeme do 2 základních skupin - *hepatitida typu A* a *hepatitida typu B*. Virová hepatitida typu A se svými příznaky podobá chřipce. K šíření viru dochází při nedodržování hygienických zásad. Průběh virové hepatitidy B je podobný jako u hepatitidy typu A. Rozdíl je však v typu šíření, přičemž hepatitida typu B se šíří krví. Nadměrná konzumace tvrdého alkoholu může vést k cirhóze [3] jater. Během cirhózy dochází k postupnému nahrazení jaterních buněk zjizvenou (vazivovou) nefunkční tkání. Hlavní příčinou onemocnění cirhózou nemusí být pouze nadměrné pití alkoholu, může být i virového původu. Dále sem patří nádorové onemocnění jater. Odhaduje se, že v roce 2020 byla rakovina jater celosvětově 6. nejvíce diagnostikovaným typem rakoviny a 3. nejčastější příčinou úmrtí na rakovinu [4]. Existují rovněž odhady, že do roku 2040 počty diagnóz a úmrtí budou dále růst, a to až o 55 % [5]. Nádorové onemocnění jater lze léčit buď transplantací, anebo resekcí [6]. Nemocní po transplantaci jater žijí dalších 5 let až v 75 % případů. Resekce je vhodná pouze při raném nálezu nádoru [7], navíc nesmí být játra postižena cirhózou, která sníží schopnost regenerace jater. Udává se, že resekci lze provést pouze u 5-15 % pacientů.

1.3 Srovnání prasečích a lidských jater

Za největší rozdíl [8, 9] mezi lidskými a prasečími játry (obr. 1.2) lze považovat počet jaterních laloků. Lidská játra jsou tvořena dvěma laloky, zatímco prasečí játra jsou rozděleny do 4 až 6 laloků. Počet laloků, v případě prasete, se může lišit v závislosti na plemeni. V případě prasete přeštického černostrakatého jsou játra nejčastěji rozděleny do 5 laloků (right lateral, right medial, left lateral, left medial a caudate). Oba orgány lze rozdělit do 8 segmentů. Prasečí játra mají, ve srovnání s lidskými játry, nižší hmotnost

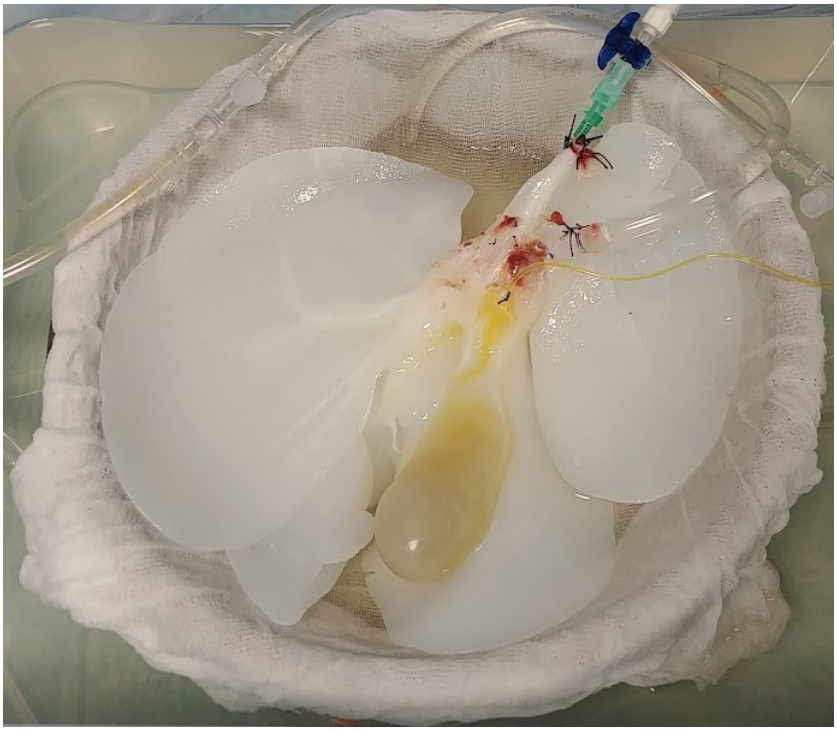
a s tím i odpovídající nižší objem. Co se týče arteriální vaskulatury, nebyly nalezeny žádné konkrétní rozdíly. I přes výše zmíněné rozdíly jsou prasečí játra vhodná pro použití v experimentální chirurgii.



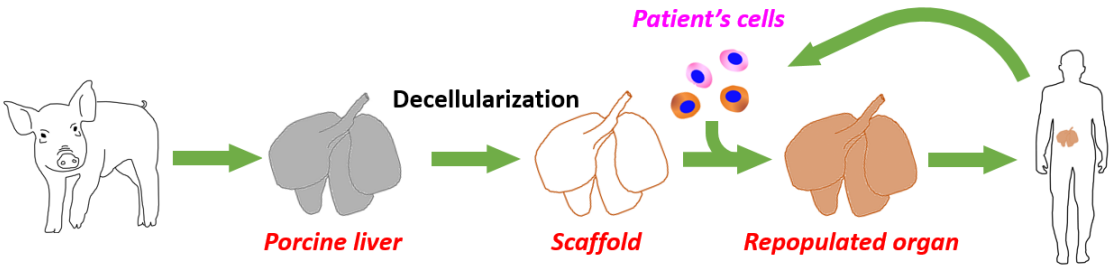
Obrázek 1.2: Játra černostrakatého přeštického prasete s viditelným rozdělením do jednotlivých segmentů (římské číslice II-VIII). Měřítko 5 cm. Převzato z [8].

1.4 Jaterní tkáňové inženýrství

Jednou z hlavních myšlenek jaterního tkáňového inženýrství je využití decelularizovaného skeletu (scaffoldu) (obr. 1.3), na který jsou aplikovány pacientovy zdravé jaterní buňky [10]. Proces znovuosídlení se nazývá recelularizace či repopulace [11]. Celý proces recelularizace je popsán na obr. 1.4. Za jednu z hlavních výhod tohoto přístupu lze považovat, že v případě použití pacientových kmenových buněk (autologní buňky) by se bylo možné kompletně vyhnout procesu imunosuprese. Imunosuprese je proces, který se používá u pacientů, kteří absolvovali transplantaci orgánu. Pomocí tohoto procesu dochází ke zmírnění přirozené imunitní reakce organismu. Po vyjmutí jater z těla dárce (prase černostrakaté přeštické) jsou játra uložena do sterilního kontejneru, kde jsou zmrazeny na teplotu -80 °C. Následně jsou játra pomalu rozmrazena až na teplotu 4 °C. Následně jsou opláchnuty fyziologickým roztokem. Poté může proběhnout proces decelularizace jater skládající se např. z 18 hodinového průtoku chemické látky (detergentu) Triton X-100 skrz játra. Obecná vize je taková, že v ideálním případě by měl být nově osídlený skelet nakonec implantován do těla pacienta a tím se vyhnout transplantaci orgánu od lidského dárce, kterých je nedostatek.



Obrázek 1.3: Výsledný decelularizovaný skelet (scaffold) přeštického černostrakatého prasete.



Obrázek 1.4: Schéma znázorňující proces decelularizace a následné recelularizace/repopulace deculularizovaného skeletu (scaffoldu) pacientovými buňkami. Převzato z [10].

2. Histologie

Histologie [12, 13, 14] je vědní disciplína, která se zabývá studiem živočišných tkání a orgánových struktur na mikroskopické úrovni. Histologii řadíme mezi biologické vědy. Počátky histologie jsou úzce spjaty s vynálezem optického mikroskopu na počátku 17. století. Není úplně jasné, kdo jako první zkonstruoval mikroskop, v literatuře [15] se objevují lidé jako Hans Jannsen se svým synem Zachariasem, Galileo Galilei či Talian Stelluci. Prvním, kdo použil mikroskop v biologických vědách, byl Antoni van Leeuwenhoek, což lze považovat za důležitý milník pro vznik histologie. Leeuwenhoekovi se podařilo sestrojit mikroskop se zhruba 200-násobným zvetšením, díky čemuž mohl vůbec poprvé pozorovat červené krvinky (erytrocyty).

2.1 Zpracování tkáňového vzorku

V první řadě byl do zvířete implantován buňkami osídlený decelularizovaný skelet (scaffold) prasečích jater. Scaffold byl implantován do tukové tkáně v břišní dutině, tzv. předstěry (lat. *omentum*) pokusného zvířete. V našem případě bylo pokusným zvířetem přeštické černostrakaté prase [16]. Po nějakém čase byl vzorek ze zvířete opět chirurgicky extrahován k dalšímu výzkumu.

2.2 Příprava tkáňového vzorku

Obecně přípravu tkáňového vzorku [12, 14, 15] dělíme do 3 po sobě jdoucích hlavních fází - *fixace* (2.2.1), *zalití do parafínu* (2.2.2) a *barvení* (2.2.3). Stručně řečeno, během fixace dochází, s využitím chemických látek, k zabránění rozkladu a zachování struktury tkání. Následně jsou zafixované tkáně zality do parafínu, aby z nich bylo možné zhotovit tenké řezy pomocí mikrotomu. Na závěr jsou tenké řezy barveny histologickými barvivy, které pomohou zvýraznit různé struktury ve tkáních.

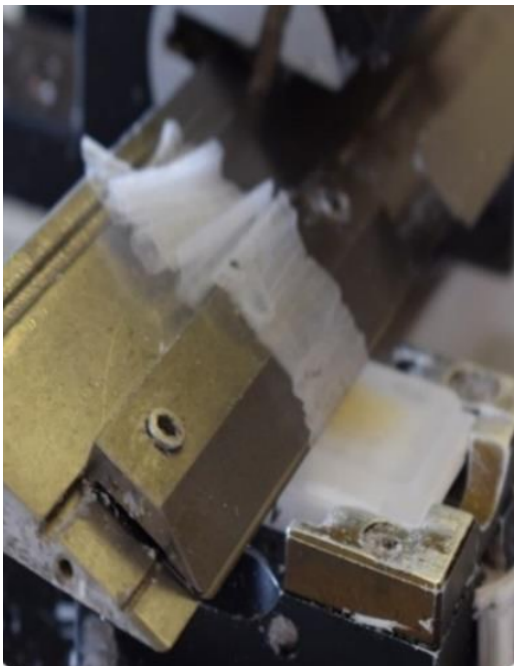
2.2.1 Fixace

V prvním kroku přípravy tkáňového vzorku musí být vzorek fixován. Často se k tomuto účelu používá formalín, což je asi 37% vodný roztok formaldehydu, společně s ostatními chemickými látkami. Možné je i použití pufrovaného formolu (4%), jehož výhodou je udržení optimální hodnoty pH, díky čemuž nedochází k poškození DNA. Formol se používá především u genetických metod. Na fixační účely lze aplikovat i tekutiny s kyselinou pikrovou. Kyselina pikrová se vyznačuje jako látka výbušné povahy. V případě, že tato látka přijde do kontaktu s kovem, který působí v tomto případě jako katalyzátor, může dojít k výbuchu. Příklady tekutin obsahující kyselinu pikro-

vou jsou např. Bouinova tekutina či Pasteelsova tekutina. Z ostatních tekutin je možné na fixaci použít ještě bezvodý 100% ethanol nebo oxid osmičelý. Tkáň je však možné fixovat i fyzikálně, a to pomocí suchého tepla (v bakteriologii), varem či rychlým zmrazením (s využitím vlastností kapalného dusíku). Mezi hlavní cíle fixace tedy patří: zastavení metabolismu buňky; usmrcení plísní, virů a bakterií a v neposlední řadě zabránění procesu autolýzy [14], což je jev, kterému podléhají tkáně a orgány po zástavě přísunu kyslíku. Během autolýzy dochází k rozkladu tkáně způsobeným vlastními enzymy. V obecném případě je třeba, aby fixační prostředky co nejlépe zachovávaly strukturu a barvitelnost tkáně. Dalším požadavkem je, aby rychle prostupovaly tkání. Optimální délka fixace činí 24-48 hodin.

2.2.2 Zalití do parafínu

V následující fázi je implantát po fixaci nejdříve zbaven nečistot a dehydratován v alkoholových roztocích za účelem zbavení vody. V dalším kroku je vzorek zbaven alkoholu za pomoci organických rozpouštědel jako xylen či toluen. Následně je vzorek zalit rozehřátým tekutým parafínem o teplotě 56 až 58 °C. Po ztuhnutí parafínu, který vyplnil všechny mikroskopické štěrbiny ve tkáni, je konzervovaný vzorek vytvarován do bloku. Mimo jiné lze vzorek zalít do želatiny, celodalu (polymer močoviny a formolu) a dále do vosků rozpustných ve vodě. Poté se vzorek vloží do mikrotomu a pomocí nože jsou z parafínového bloku krájené velice tenké řezy o tloušťce několika tisícin milimetru (v řádu několika jednotek mikrometru) (obr. 2.1).



Obrázek 2.1: Krájení parafínového bloku pomocí mikrotomové žiletky. Na žiletce dochází ke hromadění pásky řezů. Převzato z [14].

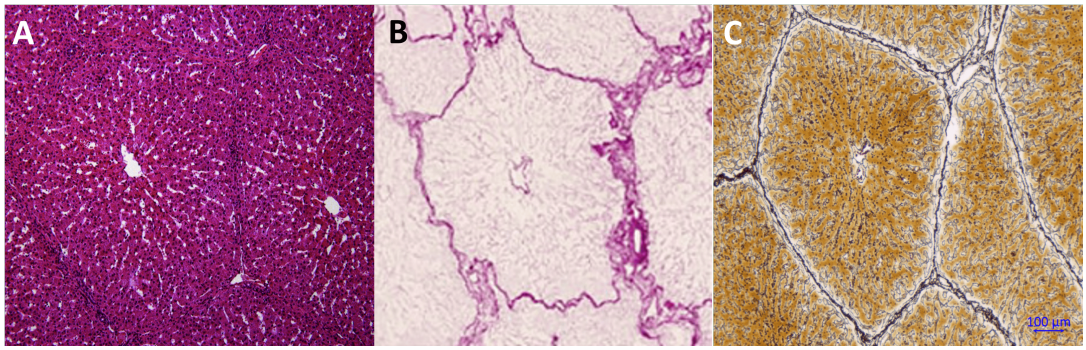
Pro správný účinek musí být parafínové bloky vychlazené v mrazničce. V dnešní době jsou jednotlivé vzorky krájeny pomocí žiletek, které mají

různé složení slitiny, tvrdost a broušení. Žiletka se volí v závislosti na tvrdosti tkání. Rozlišujeme několik druhů mikrotomů: sáňkový, rotační, zmrazovací (kryostat) a ultramikrotom. Jednotlivé získané řezy jsou následně umístěny do velké Petriho misky, obsahující studenou destilovanou vodu. Po oddělování či rovnání jednotlivých řezů pomocí preparačních jehel či podložních skel je řez přenesen, s využitím podložního skla, do vodní lázně o teplotě 47-49 °C. Dojde ke konečnému vypnutí řezu, který je následně nalepen, s využitím preparačních jehel, na podložní sklo.

2.2.3 Barvení

Na závěr jsou jednotlivé řezy, na podložních sklech, obarveny a to z důvodu bezbarvých parafínových sekcí. Je třeba rozpustit parafín, k čemuž opět použijeme organická rozpouštědla xylen či toluen. Parafínu je třeba se zbavit ve všech případech, nezávisle na zvolené barvicí metodě. Poté musí dojít k rehydrataci pomocí sestupné řady alkoholových roztoků z důvodu nerozpuštění rozpouštědla ve vodě. Následně je ve vodě vzorek na sklíčku nabarven hematoxylinem. Z důvodu kontrastního pozadí nebo jasnějšího rozlišení mezi různými druhy tkání se použije eosin, avšak tentokrát dojde k obarvení v alkoholu. Předtím opět došlo k rehydrataci pomocí sestupných alkoholových roztoků.

V případě H&E (hematoxylin a eosin) barvení jsou buněčná jádra tmavě modrá, cytoplazma buněk růžovo-modrá (do fialova) a decelularizovaný skelet čistě růžový. V našem případě však byl však použit kit na barvení (Gomori reticulin). Tento druh barvení je založen na impregnaci retikulinových vláken stříbrem. Retikulinová vlákna totiž nelze detekovat prostřednictvím H&E barvení. Jádra jsou v tomto případě tmavá a extracelulární matrix má oranžovo-hnědé odstíny. Zmíněné typy barvení jsou zobrazeny na obr. 2.2. Dále lze použít např. imunohistochemické barvení či Gomori trichrome. Imunohistochemické barvení [17, 18] funguje na bázi detekce tkáňových antigenů s využitím imunologické vazby (princip vazby antigenu a protilátky). Gomori trichrome [19] barvení probíhá v postupném barvení v barvivech hematoxylin, chromotrop a fuxin.



Obrázek 2.2: Ukázky jednotlivých barvení. Na (A) je zobrazeno H&E barvení (převzato z [8]), (B) zobrazuje scaffold obarvený pomocí H&E barvení a (C) znázorňuje kit na barvení retikulinových vláken.

2.3 Hodnocení tkáňového vzorku

Pro hodnocení získaného tkáňového vzorku či mikroskopických obrazů se používá stereologie [20]. Stereologie, jež je inspirována geometrií, zkoumá vlastnosti trojrozměrných objektů na základě jejich dvourozměrných rovinných řezů. S využitím statistických metod umožňuje kvantifikovat vlastnosti struktur a objektů v materiálech, ať už jde o jejich velikost, tvar, uspořádání, nebo vnitřní strukturu. Stereologie představuje přístup pro analýzu mikroskopických snímků a řezů, ať už pochází z optické mikroskopie, elektronové mikroskopie, tomografie, nebo jiných technik. Počátky moderní stereologie se datují do počátku 60. let 20. století, její vývoj však probíhal už od 17. století.

Pokud spolu porovnáme obrazovou analýzu a stereologii, pak mezi hlavní výhody analýzy obrazu patří např. rychlé a efektivní zpracování velkého množství preparátů, možnost automatizace kroků analýzy či interakce s uživatelem pro kontrolu a korekci. Naopak mezi nevýhody řadíme nižší efektivitu, která může nastat u preparátů s neuniformním vzhledem a artefakty či časovou náročnost v případě výskytu chyb (jejich identifikace a oprava). Mezi kladné vlastnosti stereologických metod patří vysoká reprodukovatelnost většiny metod díky binárnímu hodnocení (ano/ne), robustnost vůči odchylkám v barvení a jiným artefaktům (např. praskliny či prach), které by komplikovaly obrazovou analýzu, či možnost zahrnutí všech preparátů do hodnocení. Za nevýhodu lze považovat omezenou automatizaci stereologických metod. Použití velké většiny řezů rovněž umožňuje efektivní vzorkování bločků a zároveň snižuje variabilitu výsledků.

Správný stereologický postup pro kvantifikaci čtveřice parametrů prvního řádu, které se v literatuře označují jako V, A, L, N (V = objem, A = plocha, L = délka a N = počet) vyžaduje tento vztah: „Součet dimenzí veličiny a stereologické sondy musí být vždy roven třem.“ Součet dimenzí v jednotlivých řádcích musí být tedy v následující tabulce (tab. 2.1) roven 3.

Tabulka 2.1: Tabulka znázorňující příslušné dimenze geometrických sond kvantifikovaných veličin. Předěláno podle [20].

Dimenze geometrické sondy	Dimenze kvantifikované veličiny
Bod (L^0)	Objem (L^3)
Linie (L^1)	Plocha (L^2)
Rovina (L^2)	Délka (L^1)
Objem (L^3)	Počet (L^0)

Uveďme příklad, kdy bychom chtěli spočítat počet buněk v tkáni. V případě, že bychom zhodnotili pouze počet profilů těchto buněk (0-D) v rovině řezu tkání (řez je idealizovaně 2D), činil by součet $0 + 2 = 2$, což je menší číslo než 3 dimenze skutečné tkáně zahrnující buňky. Z toho vyplývá, že nelze zanedbat trojrozměrnost buněk.

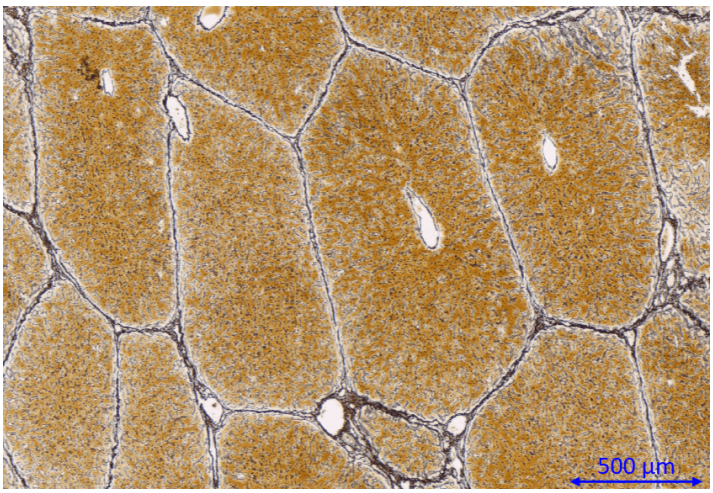
Další důležitou vlastností je náhodnost aplikace stereologických sond na hodnocené vzorky. Náhodnost musí být splněna z hlediska vzájemné pozice

i orientace. Kvantitativní přístupy zahrnují kromě technik počítání objektů, délek, ploch a objemů také statistické hodnocení výsledků, analýzu správnosti a přesnosti odhadů či design vzorkování. Vzorkování zahrnuje výběr částí k analýze z makroskopické úrovně až na úroveň mikroskopickou. Je rovněž třeba mezi sebou rozlišovat pojmy přesnost a správnost. Přesnost lze popsat jako míru shody mezi předešlými a následujícími měřeními. Správnost odkazuje na to, jak moc se naše odhady parametrů blíží očekávané hodnotě.

2.4 Digitalizace tkáňového vzorku

Nabarvený tkáňový vzorek položený na sklíčku je naskenován pomocí skeneru ZEISS Axioscan, čímž získáme digitalizovaný tkáňový vzorek (mikroskopický obraz) (obr. 2.3). Obraz uložíme v počítači, díky čemuž je s ním možné dále pracovat. Rozlišení získaných obrazů je zpravidla velmi vysoké, jedná se tedy o velice kvalitní obrazy, co se detailů týče. Na druhou stranu, počítačové zpracování těchto obrazů je paměťově velice náročné. Velikost těchto obrazů se může pohybovat v řádu stovek MB až jednotek GB. Skener navíc ukládá digitalizované vzorky ve speciálním formátu .czi, což je proprietární formát společnosti ZEISS určený právě pro ukládání mikroskopických obrazů ve vysokém rozlišení. Soubory s příponou .czi je možné otevřít buď ve specializovaném programu, určeném přímo pro práci s nimi, jako např. ZEISS ZEN [21], anebo prostřednictvím např. programu ImageJ [22].

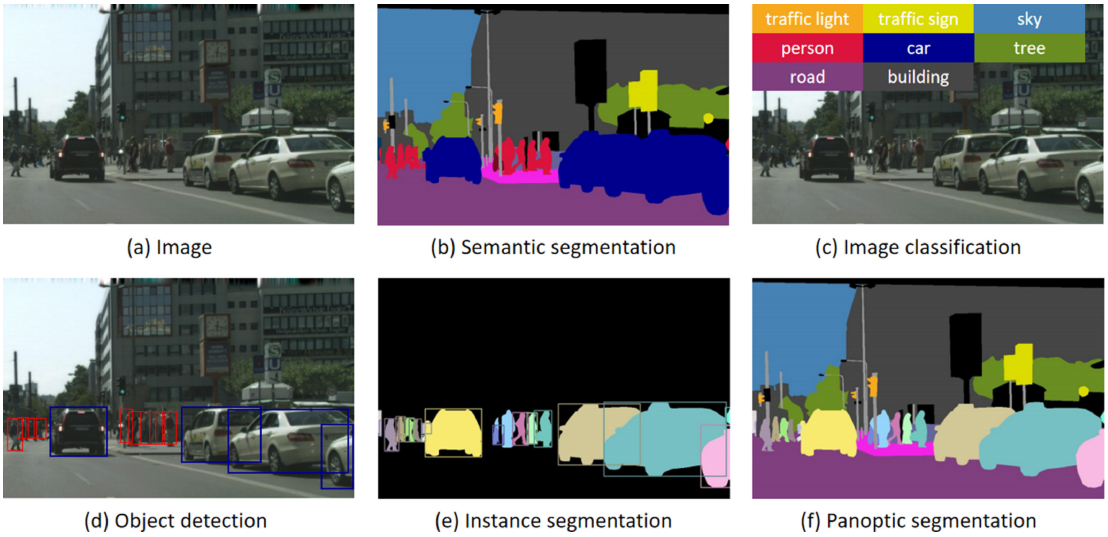
Na každém získaném mikroskopickém snímku jsou zřetelné jaterní lalůčky (lat. *lobuli hepatis*), což jsou mikroskopické útvary ve tvaru mnohoúhelníků. Jak již bylo zmíněno dříve, jedná se o základní stavební prvek jaterní tkáně. Uprostřed jaterního lalůčku se zpravidla nachází centrální žíla. Mikroskopické snímky tkáňových vzorků jsou v anglické literatuře často označovány jako „whole slide images“ (WSI).



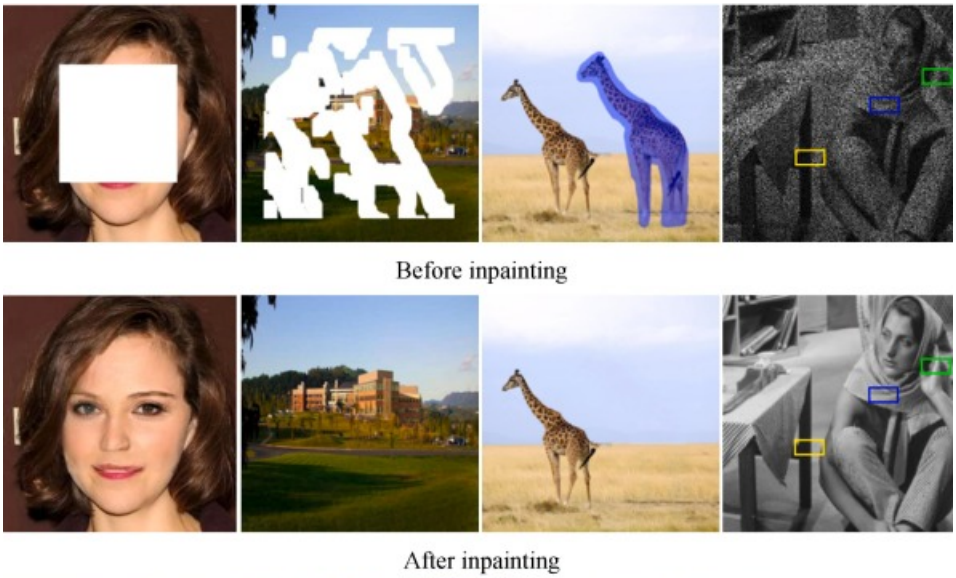
Obrázek 2.3: Mikroskopický detail jaterní tkáně. Na obrázku jsou viditelné mnohoúhelníkové jaterní lalůčky. Zobrazená jaterní tkáň byla nabarvena pomocí kitu na barvení retikulinových vláken.

3. Metody počítačového vidění

Mezi obecné cíle metod počítačového vidění patří klasifikace, detekce objektů a segmentace obrazu. V případě klasifikace pouze určíme do jaké třídy náleží objekt v obraze. V případě detekce označíme objekty pomocí tzv. bounding boxů. Cílem segmentace je pak rozdělení vstupního obrazu na jednotlivé segmenty a objekty. Segmentaci dělíme do několika skupin - semantic segmentation [23], instance segmentation [24] a panoptic segmentation [25]. U semantic segmentation se jedná o per-pixel segmentaci, tzn. každý pixel je přiřazen do třídy nebo objektu. V případě instance segmentation jsou jednotlivé objekty v obraze detekovány a poté per-pixel segmentovány. Panoptic segmentation spojuje oba dříve zmíněné přístupy segmentací dohromady, zajímají nás instance i semantic segmentace zároveň. Jednotlivé úlohy počítačového vidění jsou zobrazeny na obr. 3.1. Za další úlohu počítačového vidění můžeme považovat úlohu filtrace, tj. odstranění objektu z obrazu a vyplnění prázdného místa příslušnou texturou (z angl. *inpainting* [26]). Navíc tento přístup může sloužit např. k odstranění šumu z obrazu. Použití inpaintingu je zobrazeno na obr. 3.2.



Obrázek 3.1: Porovnání jednotlivých typů úloh spojených s počítačovým viděním - (a) vstupní obrázek, (b) semantic segmentation (per-pixel segmentace), (c) klasifikace objektů v obraze do jednotlivých tříd, (d) detekce objektů pomocí bounding boxů, (e) instance segmentation a (f) panoptic segmentation. Převzato z [23].



Obrázek 3.2: Příklady aplikací filtrace/inpaintingu. Převzato z [26].

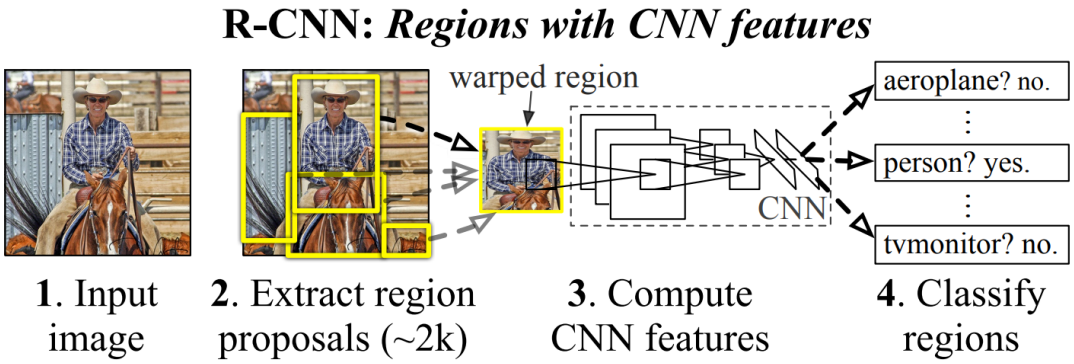
3.1 Detekce

Hlavním cílem detekce [27] je lokalizace jednoho či více objektů v obraze. Detekované objekty jsou označeny pomocí bounding boxů. Pomocí bounding boxu je možné popsat prostorové umístění objektu v obraze. V případě 2D obrazů jsou bounding boxy reprezentovány obdélníky. Jsou často popsány pomocí 2 párů souřadnic: (x_1, y_1) a (x_2, y_2). První pár souřadnic značí levý horní roh a druhý pár souřadnic označuje pravý dolní roh obdélníku/boxu. Pro popis bounding boxu je rovněž možné použít souřadnice středu (x_c, y_c) společně s jeho výškou a šířkou. Tyto souřadnice popisují levý horní roh a pravý dolní roh obdélníku. V praxi se detekce vyskytuje např. v úloze autonomního řízení (je třeba detekovat okolní vozidla, chodce, atp.), v úloze pohybu robotů či v bezpečnostních systémech.

3.1.1 R-CNN

Detekční neuronová síť R-CNN [27, 28] (Regions with CNN features) byla představena v roce 2013. R-CNN se skládá ze 3 modulů. Úkolem prvního modulu je generování návrhů oblastí nezávislých na třídách. Proces generování oblastí si lze představit jako přikládání pravoúhlých čtyřúhelníků o různých velikostech na vstupní obraz. Pro generování navržených oblastí autoři použili algoritmus Selective Search [29]. Pomocí těchto návrhů je definována množina kandidátních detekcí pro detektor. Druhým modulem je konvoluční neuronová síť, která se stará o extrakci příznakového vektoru z každé navržené oblasti. Příznakový vektor má fixní délku. Na základě příznakového vektoru se následně určí třída navržené oblasti a příslušný bounding box. Třetí modul je reprezentován množinou SVM klasifikátorů. SVM klasifikátory přísluší konkrétním třídám, které chceme detekovat v obraze. Jinými slovy, počet SVM klasifikátorů odpovídá počtu detekovaných tříd. Po-

mocí nich určíme, zda navržená oblast obsahuje danou třídu. Dále pomocí extrahovaného příznakového vektoru a určeného bounding boxu navržené oblasti natrénujeme lineární regresní model k predikci skutečného (ground truth) boxu. Celý proces R-CNN je popsán na obr. 3.3.

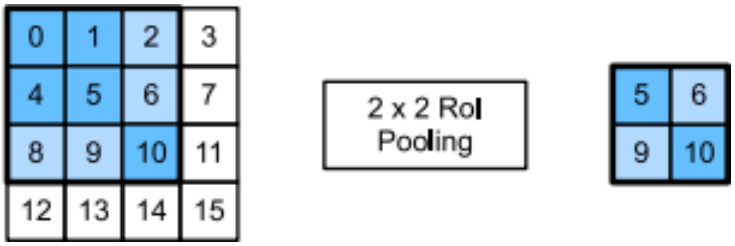


Obrázek 3.3: Schéma znázorňující jednotlivé kroky detekční neuronové sítě R-CNN. Převzato z [28].

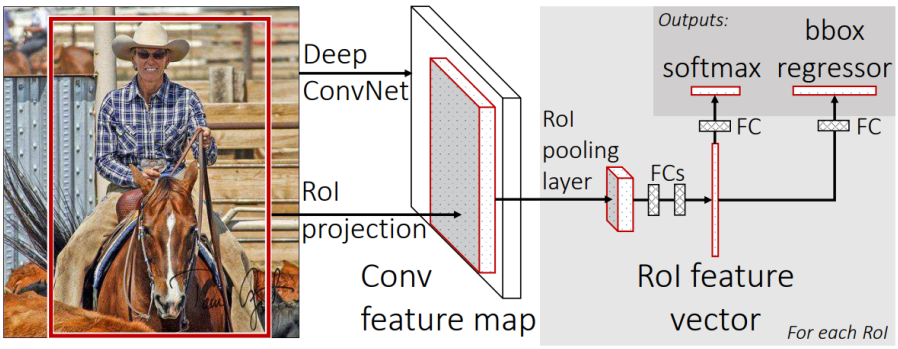
Za nevýhodu tohoto algoritmu lze považovat velkou výpočetní složitost. Vezmeme-li v potaz, že se v prvním modulu generuje i řádově několik tisíc návrhů oblastí z jednoho obrazu. V dalším kroku tyto navržené oblasti vstupují do konvoluční sítě, kde dochází k tisícům dopředných propagací. Kvůli této velké výpočetní zátěži je znemožněno široké využití tohoto přístupu v reálných úlohách.

3.1.2 Fast R-CNN

Fast R-CNN [27, 30] je přístup, který byl uvedený v roce 2015. Autoři udávají, že tato síť je rychlejší během trénování i inference ve srovnání s R-CNN. Rovněž s ní dosáhli vyšší hodnoty mAP (mean Average Precision) na PASCAL VOC 2012 datasetu (66 % vs 62 %) opět ve srovnání s R-CNN. Na rozdíl od R-CNN je tato síť trénována end-to-end. Vstup Fast R-CNN je tvořen celým vstupním obrazem společně s množinou navržených oblastí. Nejprve je vstupní obraz převeden na feature mapu prostřednictvím několika konvolučních a max-pooling vrstev. Poté je pomocí RoI pooling vrstvy (obr. 3.4) pro každou navrženou oblast (= RoI) vyextrahována malá feature mapa o velikosti např. 7×7. Každá RoI je definována pomocí tuplu o velikosti 4 (r, c, h, w). První 2 čísla r a c specifikují levý horní roh a čísla h a w zase označují výšku a šířku. RoI max-pooling vrstva poté rozdělí $h \times w$ okno RoI do mřížky $H \times W$ podoken o přibližné velikosti $h/H \times w/W$. V dalším kroku je na každé podokno aplikován max-pooling. Následně malé featury mapy reprezentují vstupy do několika fully connected vrstev, odkud následně postupuje do 2 výstupních větví. V první větvi dojde k určení softmax pravděpodobnosti, tím získáme třídu. Z druhé větve získáme 4 body definující bounding box. Jednotlivé kroky Fast R-CNN jsou popsány na obr. 3.5.



Obrázek 3.4: Ilustrace znázorňující fungování RoI pooling vrstvy. Převzato z [27].



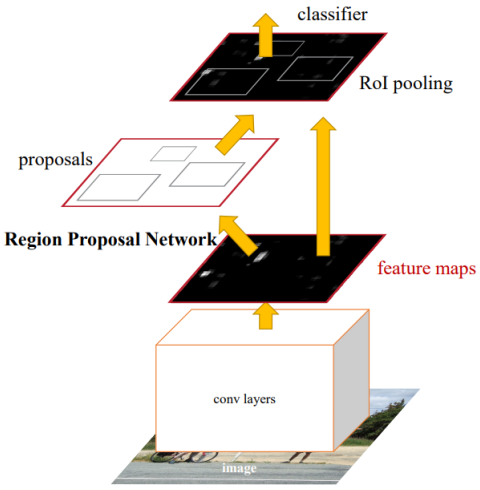
Obrázek 3.5: Schéma znázorňující jednotlivé kroky detekční neuronové sítě Fast R-CNN. Převzato z [30].

3.1.3 YOLO

YOLO (You Only Look Once) [31] je neuronová síť určená pro real-time úlohy. YOLO zvládne zpracovat 45 snímků za 1 sekundu, odvozená menší verze Fast YOLO pak dokonce 155 snímků za sekundu. Autoři transformovali detekční úlohu na regresní úlohu. V porovnání s ostatními detekčními přístupy se YOLO dopouští více chyb spojenými s lokalizací, na druhou stranu je zde menší pravděpodobnost predikce falešně pozitivních vzorků na pozadí. Proces detekce nejprve probíhá zmenšením vstupního obrazu na velikost 448×448. V konvoluční síti je vstupní obraz nejprve rozdělen do mřížky o velikosti $N \times N$. V dalším kroku se rozhoduje, jaké konkrétní buňky v mřížce obsahují nějaký objekt. Poté dojde k sestrojení kandidátních bounding boxů ohraničující jednotlivé objekty na základě několika parametrů. Každý bounding box je reprezentován vektorem (regresní úloha). Následně je na bounding boxy aplikována IoU metoda s nějakým konkrétním definovaným prahem. Jelikož IoU metoda nemusí být dostatečná např. z důvodu většího množství bounding boxů ohraničujících jeden objekt, použijeme metodu non-max suppresion. Architektura YOLO se skládá z 24 konvolučních vrstev, 4 max-pooling vrstev a 2 fully connected vrstev.

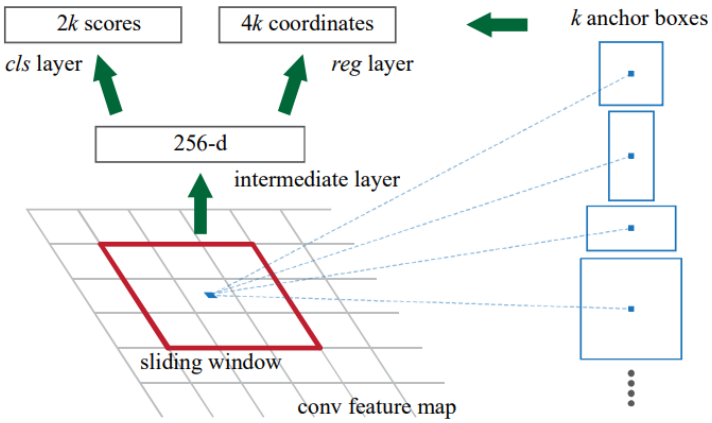
3.1.4 Faster R-CNN

V první řadě je neuronová síť Faster R-CNN [27, 32] vylepšením předešlého přístupu Fast R-CNN. Tato síť byla představena jen několik málo měsíců po již zmíněné Fast R-CNN. Opět došlo ke zlepšení mAP na PASCAL VOC 2012 datasetu a to na 70.4 %. Tato síť jako první využívá konvoluční neuronovou síť RPN (Region Proposal Network) k návrhu oblastí pro detekci, což je jediný rozdíl oproti Fast R-CNN (Fast R-CNN zůstává jako detekční síť). Zjednodušené schéma Faster R-CNN je na obr. 3.6.



Obrázek 3.6: Schéma znázorňující jednotlivé kroky detekční neuronové sítě Faster R-CNN. Převzato z [32].

RPN jsou trénovány end-to-end k tomu, aby generovaly návrhy oblastí ve vysoké kvalitě. Na vstupu RPN je obraz jakékoliv velikosti a na výstupu jsou navržené oblasti. Každá oblast má obdélníkový tvar a tzv. objectness score. Návrhy oblastí jsou generovány pomocí posouvání malé CNN po výstupní konvoluční feature mapě, která je sdílená poslední konvoluční vrstvou. Malá CNN je reprezentována konvoluční vrstvou (velikost jádra $n \times n$, např. 3×3), následovaná fully connected vrstvou. Díky konvoluci došlo k namapování na vektor nižší dimenze (např. 256-d). Výstup malé CNN pokračuje do dvou fully connected vrstev ($4k$ neuronů pro polohu k boxů a $2k$ neuronů pro pravděpodobnost, zda se jedná či nejedná o objekt pro každou navrženou oblast). Písmeno k v tomto kontextu reprezentuje počet tzv. anchor boxů. Anchor boxy mají rozdílné velikosti a různý poměr stran. Pro každé umístění pohyblivého jádra predikujeme právě k návrhů oblastí. Výhodou přístupu s využitím anchor boxů je vlastnost invariance vůči translaci. V případě vzájemného překrytí anchor boxů použijeme metodu non-maximum suppression. Proces fungování RPN je popsán na obr. 3.7.



Obrázek 3.7: RPN (Region Proposal Network). Převzato z [32].

Trénování RPN probíhá s využitím ground truth boxů (anotované bounding boxy v trénovacích datech) a anchor boxů. Vyhodnocujeme vzájemný překryv boxů pomocí metriky IoU (Intersection over Union). V případě, že má anchor box hodnotu IoU > 0.7 je považován za pozitivní vzorek. Pozice a velikost anchor boxu je vypočtena z ground truth oblasti. Za negativní vzorky jsou považovány anchor boxy s hodnotou IoU < 0.3. Definujeme následující funkci ztráty (rov. 3.1):

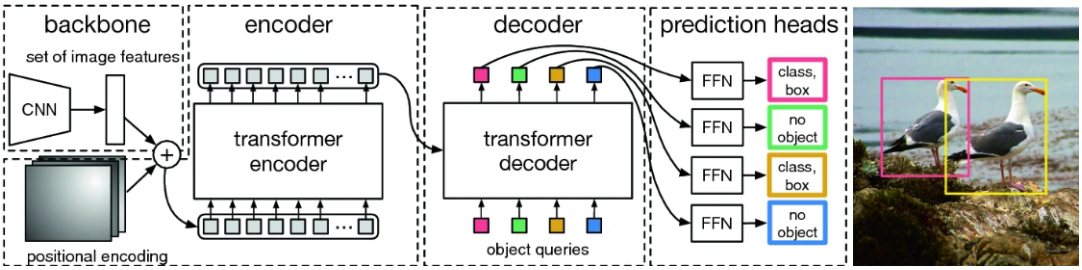
$$L(\{p_i\},\{t_i\}) = \underbrace{\frac{1}{N_{cls}}\sum_i L_{cls}(p_i,p_i^*)}_{\text{klasifikační část = SOFTMAX}} + \lambda \underbrace{\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i,t_i^*)}_{\text{regresní část = } L^1 \text{ norma}}, \tag{3.1}$$

kde N_{cls} je počet tříd, do kterých klasifikujeme, N_{reg} je počet regresních parametrů, $L_{cls}(p_i, p_i^*)$ je ztrátová funkce pro klasifikaci i-tého objektu, p_i je předpovězená pravděpodobnost, že i-tý objekt patří do dané třídy, p_i^* je skutečná binární hodnota, která značí, zda i-tý objekt patří do dané kategorie, λ je hyperparametr, $L_{reg}(t_i, t_i^*)$ je ztrátová funkce pro regresi i-tého objektu, t_i je předpovězená poloha i-tého objektu a t_i^* je skutečná poloha i-tého objektu.

3.1.5 DETR

V roce 2020 byla pro řešení úloh detekce objektů představena neuronová síť DETR (DEtection TRansformer) [33]. Architektura Transformer [34] je v dnešní době známá především díky chatbotům jako např. ChatGPT, které jsou na dané architektuře postaveny. Navržená metoda nahlíží na úlohu detekce objektů jako na množinu predikcí. Tento přístup byl do té doby již aplikován v úlohách rozpoznání řeči či strojového překladu. DETR využívá architekturu typu enkodér-dekodér převzatou z původních Transformerů. DETR predikuje všechny objekty ve scéně najednou. Detekce je zjednodušena díky zbavení se spatial anchorů a non-maximal supression. Trénování probíhá end-to-end společně se ztrátovou funkcí, která aplikuje metodu tzv. bipartite matching mezi predikovanými a ground truth objekty. Schéma DETRu je na 3.8.

Backbone DETRu je tvořena konvoluční neuronovou sítí jako např. ResNet [35], která slouží jako enkodér. Pomocí konvoluční sítě získáme high-level příznaky ze vstupního obrazu, získáme pomocí nich prostorové informace o jednotlivých objektech na snímku. K výstupu z enkodéru navíc přidáme positional encodings, které slouží k informování modelu o různých prostorových vztazích v obraze. Positional encodings jsou pro Transformer podstatné, aby mohlo dojít ke správné interpretaci pozic jednotlivých objektů v obraze. DETR mimo jiné zavádí tzv. „object queries“, „keys“ a „values“. Počet object queries je obvykle předdefinován. Keys a values odpovídají extrahovaným příznakům z výstupu konvoluční sítě. Keys reprezentují prostorové pozice v obraze. Ve values jsou obsaženy informace o příznacích. Keys a values jsou použité pro self-attention mechanismus, pomocí něhož je možné vyhodnotit důležitost jednotlivých oblastí v obraze. Celé jádro DETRu tkví v použití tzv. multi-head self-attention mechanismu. Pomocí tohoto mechanismu je DETR schopen zachytit složité závislosti a vztahy mezi jednotlivými objekty v obraze. Každý attention head může klást důraz na různé aspekty a oblasti obrazu současně. Attention heads lze metaforicky popsat jako „experty“ na různé části obrazu.



Obrázek 3.8: Schéma znázorňující architekturu DETRu. DETR se skládá z backbone, transformer enkodéru, transformer dekodéru a 4 hlav určených pro predikci. Převzato z [33].

DETR, v porovnání s Faster-RCNN, dosahuje lepších výsledků v detekci velkých objektů, pravděpodobně díky zpracování globální informace pomocí mechanismu self-attention. Naopak u menších objektů bylo dosaženo horších výsledků, než v případě Faster R-CNN. Za nevýhodu DETRu lze považovat velkou výpočetní složitost. Rovněž se začíná objevovat použití Transformerů i v případě úloh pracujícími s medicínskými daty [36].

3.2 Segmentace

Hlavním úkolem segmentace je rozdělení vstupního obrazu na jednotlivé části, které souvisejí s objekty reálného světa. Na tuto úlohu lze aplikovat tradiční segmentační techniky založené např. na jasových vlastnostech (prahování) či techniky založené na strojovém učení (především neuronové sítě). Jak již bylo zmíněno na začátku kapitoly 3, existují 3 základní typy segmentace - semantic segmentation, instance segmentation a panoptic segmentation (obr. 3.1).

3.2.1 Prahování

Jednou ze základních a nejjednodušších metod segmentace je prahování [37]. Prahování využívá vlastností odrazivosti a absorpce světla povrchů jednotlivých objektů či oblastí nalézajících se v obraze. Tím pádem lze definovat tzv. práh (často označován jako T), který slouží k segmentaci objektů a pozadí. Základní úloha prahování je tedy definována pomocí následujícího vztahu (rov. 3.2):

$$g(i,j) = \begin{cases} 1 & \text{pro } f(i,j) > T, \\ 0 & \text{pro } f(i,j) < T, \end{cases} \tag{3.2}$$

kde T je práh (předem zvolená konstanta), $g(i,j) = 1$ zvýrazní objekty a $g(i,j) = 0$ rozdělí pozadí obrazu od objektu.

Výhody prahování spočívají především ve výpočetní rychlosti (lze ji provést v reálném čase) a nenáročné hardwarové realizaci. Na druhou stranu tuto metodu lze použít pouze na obrazy s jasně rozlišitelnými objekty od pozadí. Navíc může nastat problém se zvolením správné prahové hodnoty T. Především automatické zvolení hodnoty prahu může být velice obtížné. Hodnotu prahu lze určit pomocí tzv. bimodálního histogramu, v tomto případě je práh roven hodnotě nalézající se mezi dvěma „kopci“. Pro automatické určení hodnoty prahu lze rovněž použít Otsuovu metodu [38]. Dále existují různé modifikace základní úlohy prahování, např. prahování s množinou známých jasů, poloprahování či prahování více prahy.

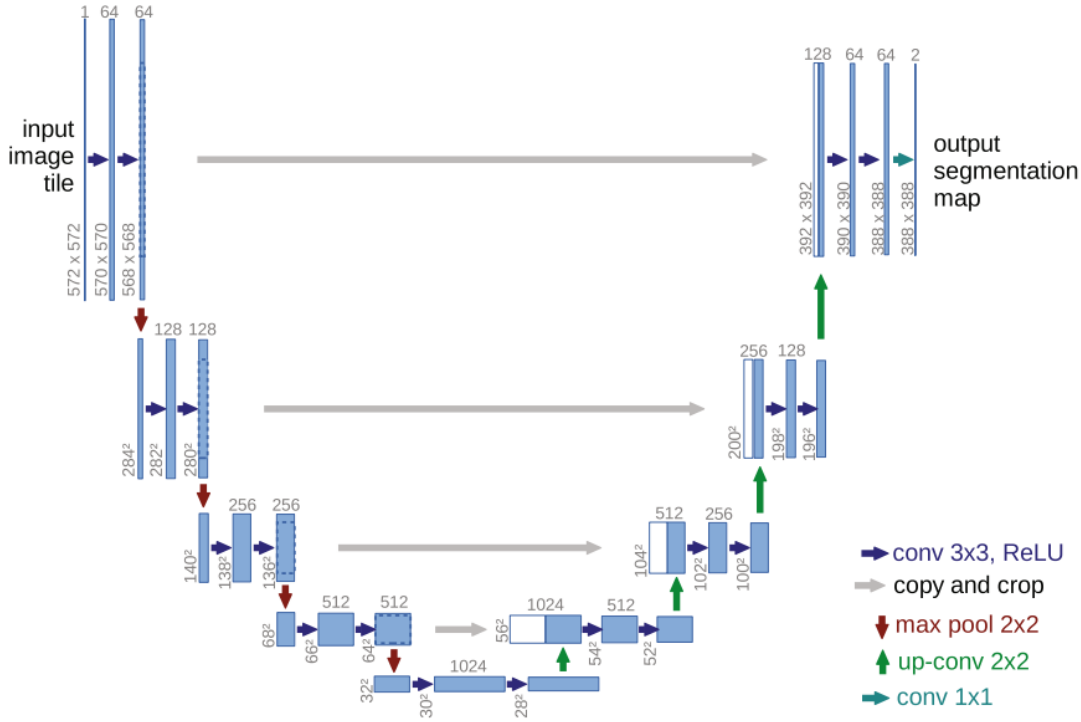
3.2.2 U-Net

Neuronová síť U-Net [39] byla představena v roce 2015 a je přímo zaměřená na segmentaci biomedicínských obrazů. Je založena na architektuře „fully convolutional network“, využívá tedy v tomto případě konvolučních vrstev. Autoři se během návrhu museli popasovat s nedostatkem trénovacích dat, což řešili pomocí augmentací. K tomuto účelu použili elastické deformace, a to důvodu, že tkáně jsou často různě deformovány.

Další výzvou pro autory bylo rozdělení vzájemně se dotýkajících objektů stejné třídy, proto představili použití tzv. „weighted lossu“, pomocí kterého pozadí mezi dotýkajícími se buňkami obdrží velkou váhu ve ztrátové funkci. Z důvodu limitované GPU paměti použili autoři strategii, pomocí níž rozdělili vstupní, často velký obraz, do jednotlivých dlaždic. Kvůli nepoužití fully connected vrstev v architektuře, obsahuje segmentační mapa pouze pixely, pro které je celý kontext dostupný pouze ve vstupním obrazu, proto byl u dlaždic navíc aplikován overlap, aby bylo možné predikovat pixely na hraničních částech obrazu. Chybějící kontext je v tomto případě extrapolován pomocí zrcadlení vstupního obrazu.

Architekturu sítě (obr. 3.9) lze rozdělit do 2 cest - *contracting* (levá část) a *expansive* (pravá část). V levé části dochází k opakované aplikaci vždy dvou 3×3 konvolucí následovanými ReLU aktivační funkcí a 2×2 max-poolingem se stridem 2 pro účel downsamplingu. V každém kroku

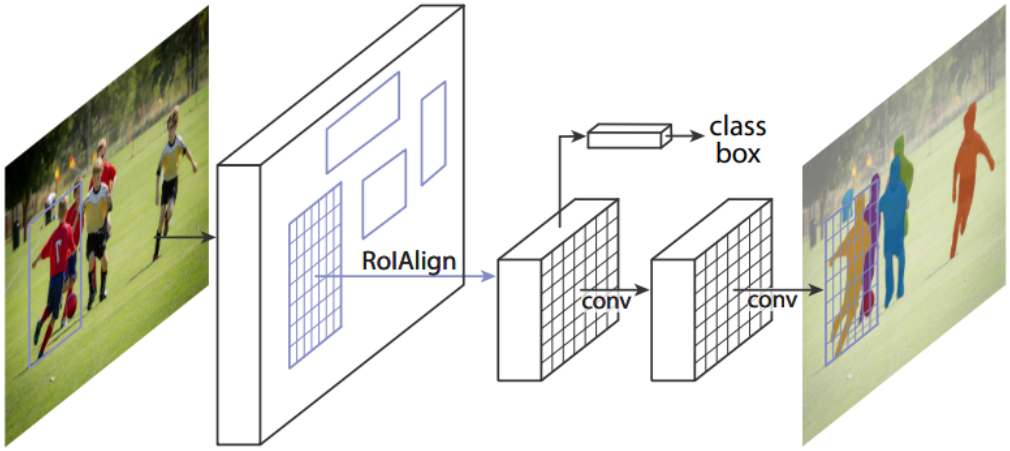
downsamlingu dojde ke zdvojnásobení feature channelů. V pravé části dochází k upsamplingu feature map, po kterém je následně aplikována 2×2 konvoluce, která rozpůlí počet feature channelů. Následuje konkatenace s odpovídajícím způsobem ořízntutou feature mapou z levé části. Dále dojde k aplikaci dvou konvolucí 3×3, z nichž každá je následovaná opět ReLU aktivační funkcí. V poslední vrstvě dojde pomocí konvoluce 1×1 k namapování feature vektoru o velikosti 64 na požadovaný počet tříd. Celkem architektura sítě obsahuje 23 konvolučních vrstev. Pro natrénování sítě použili autoři vstupní obrazy a jejich odpovídající segmentační mapy. Pro správné naučení sítě bylo klíčové využití augmentací dat. Autoři během trénování použili rovněž drop-out.



Obrázek 3.9: Schéma původní architektury neuronové sítě U-Net. Šipky označují různé aplikované operace. Převzato z [39].

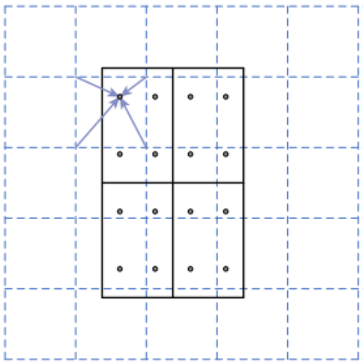
3.2.3 Mask R-CNN

Mask R-CNN [40] bylo poprvé představeno v roce 2018. Hlavní doménou Mask R-CNN je instance segmentation, tj. nejprve správně detekovat objekty v obraze (klasifikace a poté lokalizace pomocí bounding boxů) a následně provést jejich sémantickou segmentaci, což znamená, že pro každý pixel v ohraničeném objektu je určena jeho kategorie, čímž dojde k vytvoření masky daného objektu. Získané masky reprezentují jednotlivé instance objektů. Mask R-CNN dále rozšiřuje detekční neuronovou síť Faster R-CNN a to přidáním větve za účelem predikce segmentačních masek v každém Region of Interest (RoI). Zjednodušené schéma Mask R-CNN je na obr. 3.10.



Obrázek 3.10: Schéma neuronové sítě Mask R-CNN. Převzato z [40].

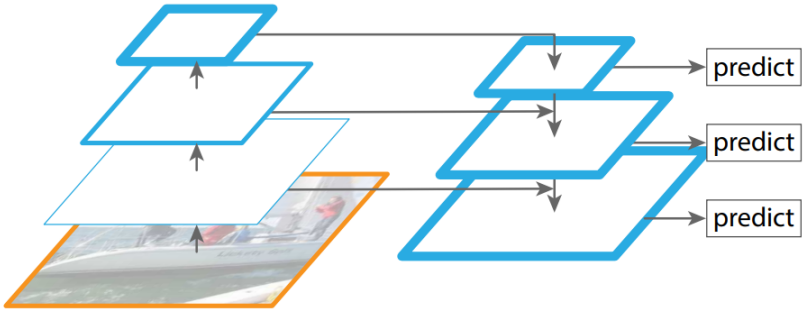
Větev určená pro segmentaci pracuje paralelně s větví určenou pro klasifikaci a regresi pomocí bounding boxu. Segmentační větev si lze představit jako malou konvoluční neuronovou síť. Při vytváření Mask R-CNN autoři představili vrstvu ROIAlign (obr. 3.11), pomocí které se vyhnuli kvantizaci, která způsobovala v předešlých přístupech nepřesnosti mezi RoI a extrahovanými příznaky. Tyto nepřesnosti měly především vliv na predikci masek. V případě RoIAlign vrstvy použili autoři bilineární interpolaci.



Obrázek 3.11: RoIAlign. Tečkovaná mřížka představuje feature mapu, černé linky reprezentují RoI a modré tečky jsou transformované body. S využitím váženého průměru nejbližších bodů z feature mapy spočteme nové souřadnice transformovaných bodů. Převzato z [40].

Co se architektury Mask R-CNN týče jako konvoluční backbone pro extrakci příznaků v celém obraze lze použít např. ResNet či ResNeXt [41] o hloubce 50 či 100 vrstev. Autoři rovněž poznamenávají, že se jim podařilo dosáhnout výborných výsledků ve smyslu přesnosti a rychlosti, a to s využitím Feature Pyramid Network (FPN) [42] a ResNetu (ResNet-FPN backbone). Pomocí FPN je možné vstupní obraz reprezentovat v různých měřítkách. Upsampling probíhá v závislosti na nejbližším sousedovi. V případě rozpoznávání bounding boxu (úlohy klasifikace a regrese) a predikci

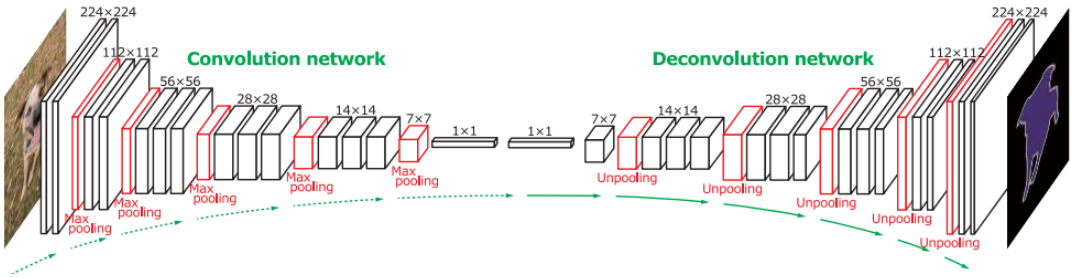
masky, autoři volí Faster R-CNN s přidáním fully convolutional mask prediction větve. Zjednodušené schéma FPN je na obr. 3.12.



Obrázek 3.12: Feature Pyramid Network (FPN). Převzato z [42].

3.2.4 Enkodér-dekodér

Na úlohu semantic segmentation lze aplikovat architekturu typu enkodér-dekodér. Enkodér si lze představit jako konvoluční neuronovou síť, naopak dekodér jako dekonvoluční neuronovou síť. Cílem enkodéru je převedení vstupního obrázku na n-dimenzionální příznakový vektor. Hlavním úkolem dekodéru je pak vytvořit ze vstupního příznakového vektoru segmentační masku. Jako první byl představen v roce 2015 DeconvNet [43]. Dekodér DeconvNetu používá vrstvy z neuronové sítě VGG16 [44], pouze došlo k odstranění poslední klasifikační vrstvy. Konvoluční síť celkem obsahuje 13 konvolučních vrstev a 2 fully connected vrstvy. Dekonvoluční síť je zrcadlovou verzí konvoluční sítě a probíhají v ní operace unpoolingu a dekonvoluce. V případě dekonvoluční sítě dochází ke zvýšení počtu aktivací, a to prostřednictvím již zmíněných operací unpoolingu a dekonvoluce. Architektura DeconvNetu je znázorněna na obr. 3.13.



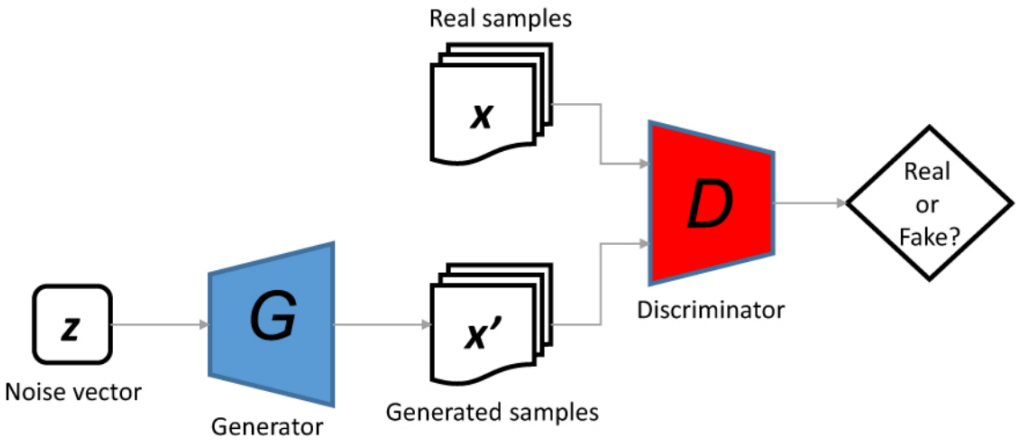
Obrázek 3.13: Schéma architektury DeconvNetu. V levé části se nachází enkodér (konvoluční neuronová síť) a v pravé části se nachází dekodér (dekonvoluční neuronová síť). Vstupem konvoluční sítě je obrázek, naopak výstupem dekonvoluční sítě je semantic segmentation (maska). Převzato z [43].

Dále lze do této části zařadit i neuronovou síť SegNet [45]. V případě SegNetu došlo k odstranění fully connected vrstev, čímž došlo k velkému snížení parametrů ve srovnání s [43] (ze 134 na 14.7 M). Autoři SegNetu se zaměřili především na úlohu segmentace spojenou s provozem na silnici (segmentace silnice, chodců, automobilů, atp.), z toho vyplývá velký důraz na rychlost a efektivnost toho přístupu. Rovněž byl představen přístup určený přímo na segmentaci medicínských obrazů - HMEDN (High-Resolution Multi-Scale Encoder-Decoder Network) [46]. Tento přístup zvládá pracovat s CT obrazy, které mají nízký kontrast a navíc jsou hranice v těchto obrazech často rozmazané až mizející.

3.2.5 GANy

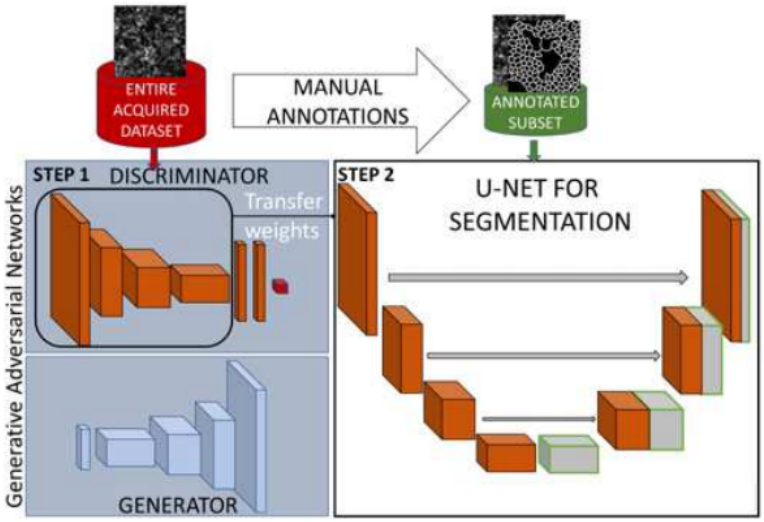
K segmentaci lze využít i tzv. GANy [47] (General Adversarial Networks). GAN se skládá ze 2 neuronových sítí - generátoru G a diskriminátoru D. Obě neuronové sítě „soupeří“ proti sobě. Hlavním úkolem generátoru je vytvářet falešné obrazy na základě n-dimenzionálního vektoru z, který má na vstupu. Vstup klasifikátoru je tvořen buď reálným obrazem, anebo falešným obrazem vytvořeným generátorem. Hlavním cílem diskriminátoru je tedy rozhodnout, zda se jedná o reálný či falešný obraz. Proces lze připodobnit ke hře minimax mezi G a D. Generátor nikdy „nevidí“ reálné obrazy. Snahou diskriminátoru je minimalizace chyby klasifikace mezi reálnými a falešnými obrazy, proto maximalizuje ztrátovou funkci. Naopak generátor se snaží o maximalizaci chyby klasifikace diskriminátoru, a proto minimalizuje ztrátovou funkci. Cíle je dosáhnout Nashovy rovnováhy, tzn. stavu, ve kterém diskriminátor určuje, zda se jedná o falešný či pravý obraz s pravděpodobností $p = 0.5$ pro jakýkoliv vzorek. Popis fungování GANu je popsán na obr. 3.14. Ztrátová funkce GANu má následující tvar:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}\left[\log D(x)\right] + \mathbb{E}_{z\sim p_z(z)}\left[\log\left(1 - D(G(z))\right)\right] \quad (3.3)$$



Obrázek 3.14: Schéma fungování GANu. Převzato z [48].

GANy byly v roce 2019 využity jako součást procesu segmentace hranic iRPE (induced Retinal Pigment Epithelial) buněk v mikroskopických obrazech [49]. Navržený přístup lze rozdělit do 2 modulů: využití GANů a následná segmentace pomocí U-Netu. Tento přístup je zobrazen na obr. 3.15. Autoři použili architekturu GANu pro získání abstraktní reprezentace ze všech vstupních neanotovaných dat (transfer learning). Abstraktní informace následně postupuje do segmentační neuronové sítě, v tomto případě se jedná o U-Net. Co se týče vnitřní architektury GANu, autoři použili dekodér a enkodér z U-Netu jako generátor, respektive diskriminátor. Na vstupu generátoru v tomto případě není n-dimenzionální vektor z, nýbrž obraz. Naučené váhy diskriminátoru byly využity v U-Netu, který byl rovněž dotrénován na malém počtu ručně anotovaných dat.



Obrázek 3.15: Proces použití GANu a U-Netu. Převzato z [49].

GANy rovněž našly své využití i v případě úlohy semantic segmentation [50]. Autoři v tomto případě implementovali diskriminátor jako fully convolutional klasifikátor určený pro klasifikaci pixelů v obraze do jednotlivých tříd. V tomto přístupu bylo využito semi-supervised trénování, tj. z trénovací množiny byly anotovány pouze některé vzorky. Do diskriminátoru vstupují falešné obrazy, neanotovaná data a anotovaná data. Generátor v tomto případě sloužil k vytvoření dalších trénovacích dat pro klasifikátor. Výstupem diskriminátoru jsou konfidenční mapy pro každou třídu a label pro falešná data.

3.3 Filtrace

V případě filtrace (inpaintingu) řešíme úlohu, kdy na vstupu je obraz obsahující díry, tj. místa která chceme „vyplnit“ tak, aby nebylo rozpoznatelné, že se v obrazu původně nějaké díry nacházely. K tomuto účelu potřebujeme masku se stejnými dimenzemi jako má vstupní obraz. Maska je často reprezentována binárním černobílým obrazem, tj. pixely mohou v tomto případě

nabývat pouze boolovských hodnot 0 = false (černá) nebo 1 = true (bílá). Pomocí masky určujeme místa chybějících pixelů v obraze, která chceme zaplnit. V našem případě značí bílá barva nacházející se v masce oblasti, které se mají vyplnit. V poslední řadě aplikujeme na vstupní obraz získanou masku a s využitím různých přístupů obdržíme výsledek.

3.3.1 Exemplar-based inpainting

Hlavním cílem exemplar-based inpaintingu [51] je odstranění velkých objektů z digitálních obrazů a jejich nahrazení vhodným pozadím. Algoritmus je založen na využití tzv. isofot. Isofota je křivka, spojující obrazové body s ekvivalentním jasem. Nejprve je třeba ručně označit tzv. target region Ω (binární maska), což je místo v obraze, které chceme odstranit a následně vyplnit. Dále definujeme tzv. source region Φ, který lze vypočítat podle následujícího vzorce (rov. 3.4):

$$\Phi = I - \Omega, \tag{3.4}$$

kde I je vstupní obrázek a Ω je target region/maska.

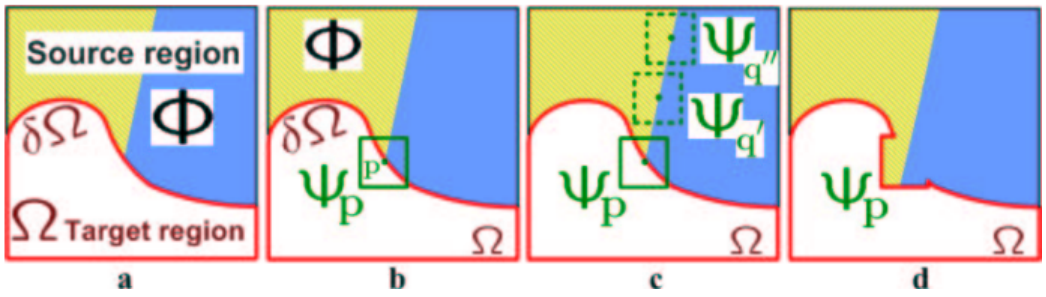
V dalším kroku definujeme template window/patch Ψ, často o velikosti 9×9 pixelů, je však doporučeno velikost template window volit v závislosti na největším rozpoznatelném texturním elementu. Další část tohoto algoritmu už probíhá automaticky. Následující 3 kroky algoritmu jsou iterativně opakovány, dokud není dosaženo nejlepšího výsledku.

V *prvním kroku* je nejdříve spočtena priorita pro každý patch na hranici chybějící části. Priorita závisí na dvou faktorech: důvěryhodnosti okolních pixelů a síle isofot směřujících do této oblasti. Čím je důvěryhodnost okolí vyšší a čím silnější je linie, tím vyšší je priorita daného patche. Ve *druhém kroku* algoritmu je určen patch s nejvyšší prioritou. Pro daný patch je v source regionu Φ objeven ten nejpodobnější patch. Pro tento účel se využívá vzorec (rov. 3.5):

$$\Psi_{\hat{\mathbf{q}}} = \arg\min_{\Psi_{\mathbf{q}} \in \Phi} d(\Psi_{\hat{\mathbf{p}}}, \Psi_{\hat{\mathbf{q}}}), \tag{3.5}$$

kde $\Psi_{\hat{\mathbf{q}}} \in \Phi$ je source exemplar, $\Psi_{\hat{\mathbf{p}}}$ je patch s nejvyšší prioritou, vzdálenost $d(\Psi_{\hat{\mathbf{p}}}, \Psi_{\hat{\mathbf{q}}})$ je definovaná pomocí sum of squared differences (SSD) na základě pixelů mezi dvěma patchy.

Z nejpodobnějšího patche jsou následně zkopírovány informace o pixelech. Pomocí tohoto způsobu se šíří textura i struktura původního obrazu do chybějící části, aniž by docházelo k rozmazání. Nakonec ve *třetím kroku*, po vyplnění patche $\Psi_{\hat{\mathbf{p}}}$ novými hodnotami pixelů, dojde k aktualizaci důvěryhodnosti okolních pixelů. Platí, že čím blíže středu oblasti chybějící části, tím nižší je důvěryhodnost. Celý proces této metody je popsán na obr. 3.16.



Obrázek 3.16: Proces exemplar-based inpaintingu. (a) Vstupní obraz s vyznačeným source regionem Φ, hranicí $\delta\Omega$ a target regionem Ω (oblast, kterou chceme zaplnit). (b) Zvolíme patch $\Psi_{\mathbf{p}}$ se středovým bodem $\mathbf{p} \in \delta\Phi$. (c) Vhodní kandidáti ($\Psi_{\mathbf{q}'}$ a $\Psi_{\mathbf{q}''}$) se nalézají na hranici mezi dvěma texturami v source regionu. (d) Z množiny vhodných kandidátů je vybrán právě ten nejvhodnější, který je poté kopírován na pozici definovanou $\Psi_{\mathbf{p}}$, čímž dojde k částečnému zaplnění a zároveň změně tvaru target regionu Ω. Převzato z [51].

3.3.2 Biharmonické funkce

Tento přístup filtrace (inpaintingu) je založen na využití tzv. biharmonických funkcí [52]. Princip tohoto přístupu spočívá ve dvojité aplikaci operátoru Laplacián (odtud plyne pojem biharmonická funkce) na vstupní obraz, který je reprezentován pomocí funkce u. Pro sestavení biharmonických funkcí jsou použity hodnoty Laplaciánových a normálových derivací funkcí na hranici. Nyní se pojd'me zaměřit na vyřešení rovnice pomocí numerické metody. Nejprve definujeme následující vztah (rov. 3.6):

$$\begin{aligned} \Delta^2 u &= 0, \\ u_N|_S &= f, \\ u|_S &= g, \end{aligned} \tag{3.6}$$

kde $\Delta^2 u = 0$ je biharmonická rovnice, $u_N|_S = f$ a $u|_S = g$ jsou podmínky spojené s hranicemi. Zjednodušeně řečeno, na základě rovnice (3.6) hledáme náležitou funkci u, která je co nejvíce hladká na všech místech.

Dále definujeme vztah pro diskrétní aproximaci bi-Laplaciánu, využijeme 13-bodové konečné diferenční schéma (rov. 3.7):

$$\Delta^2_{13} u = \frac{1}{h^4} \begin{pmatrix} & & 1 & & \\ & 2 & -8 & 2 & \\ 1 & -8 & 20 & -8 & 1 \\ & 2 & -8 & 2 & \\ & & 1 & & \end{pmatrix} u = \Delta^2 u + O(h^2) \tag{3.7}$$

Pomocí matice s centrálním koeficientem 20 a okolními koeficienty -8 aproximujeme 4. derivaci každého pixelu. Pomocí tohoto schématu dochází k šíření hodnot pixelů od hranice směrem dovnitř díry v obraze. Použitím diferenčního schématu (3.7) rovněž převedeme rovnici (3.6) na soustavu

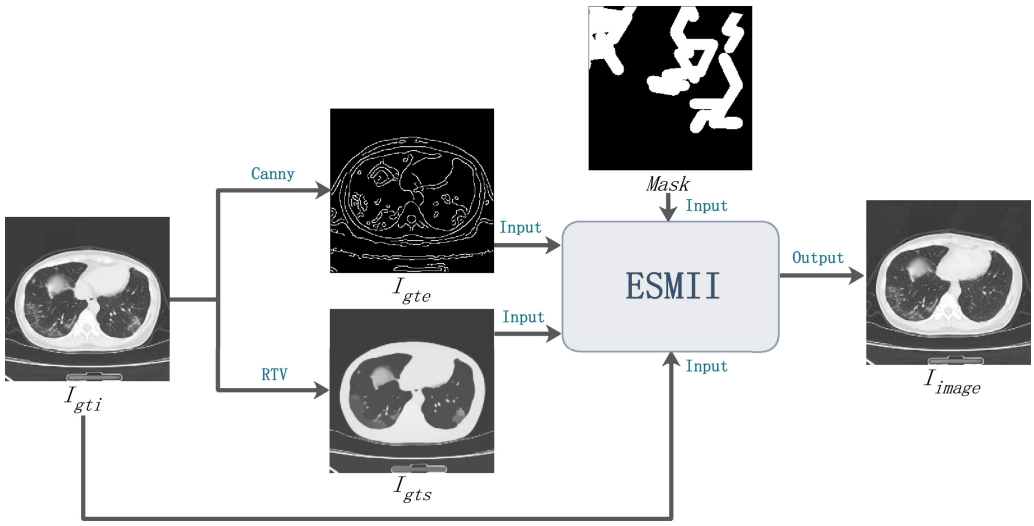
lineárních algebraických rovnic ve formě $Au = b$. Algoritmus inpaintingu pomocí biharmonických funkcí je rovněž součástí knihovny programovacího jazyka Python *scikit-image* [53].

3.3.3 GANy

Úlohu inpaintingu je rovněž možné řešit pomocí GANů. Jako příklad uveďme [54]. Autoři v tomto článku přišli s myšlenou tzv. contextual attention, pomocí které je možné použít informace ze vzdálených prostorových míst pro rekonstrukci chybějících pixelů. Dále bylo řešení s využitím GANů využito i v rámci inpaintingu medicínských obrazů [55]. Autoři k tomuto účelu využili tzv. cGany [56] (conditional GANs). Jako vstupní obrazy byly v tomto případě použity skeny z magnetické rezonance. Tyto obrazy však v praxi mohou obsahovat lokální poruchy způsobené např. kovovými implantáty. Celá myšlenka spočívá ve využití 3 hlavních komponent: CasNet [57] generátoru G (generátor umožňující převádět medicínské obrazy z PET domény do MRI domény), globálního diskriminátoru D a lokálního diskriminátoru DL. Na vstupu generátoru G je obraz obsahující chybějící oblast. Generátor se poté snaží vygenerovat kompletní obraz se zaplněnou chybějící oblastí. Globální diskriminátor D rozhoduje, zda je kompletní vygenerovaný obraz reálný či falešný. Lokální diskriminátor DL se pak zaměřuje pouze na vyplněnou oblast, kontroluje zda detaily vyplněné oblasti odpovídají okolí. Autoři pojmenovali vytvořený framework ip-MedGAN.

3.3.4 Metoda ESMII

Metoda ESMII [58] (Edge and Structure information for Medical Image Inpainting) využívá hranovou a strukturální informaci k inpaintingu medicínského obrazu. Pro získání hranové reprezentace obrazu byl autory použit Cannyho hranový detektor. Pro získání strukturální reprezentace byla použita vyhlazovací metoda RTV [59], která zároveň zachovává hrany. Do klíčového modulu ESMII tedy vstupuje ground truth obraz, hranová reprezentace obrazu, strukturální reprezentace obrazu a binární maska. Uvnitř ESMII modulu se skrývá enkodér, dekodér a diskriminátor. Pomocí enkodéru jsou vyextrahovány příznaky ze 3 již zmíněných vstupních reprezentací. Tyto příznaky následně vstupují do dekodéru, který na základě nich vytvoří opět 3 reprezentace (ground truth obraz, hranová reprezentace a strukturální reprezentace) ve 3 paralelních větvích. Nakonec diskriminátor posuzuje, zda se jedná o reálný či falešný obraz. Během trénování se používá několik ztrátových funkcí: L_{re} (reconstruction loss), L_{perc} (perceptual loss), L_{style} (style loss) či L_{adv} (adversarial loss). Zjednodušené schéma metody ESMII je na obr. 3.17.

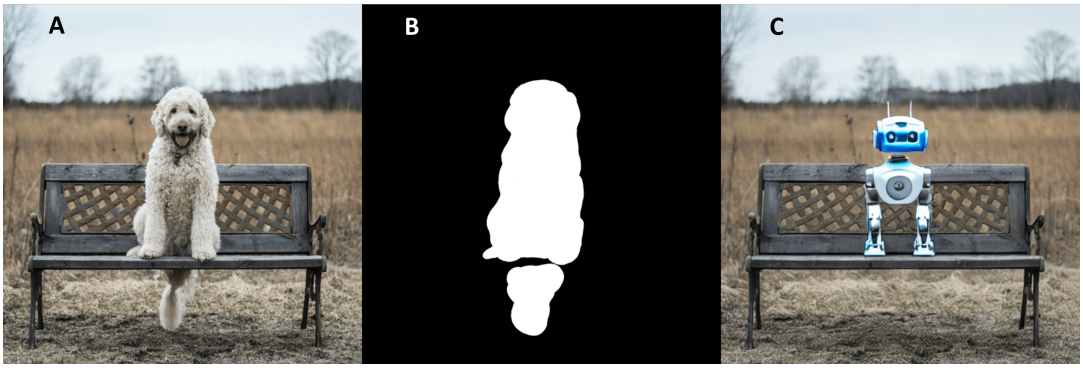


Obrázek 3.17: Schéma metody ESMII. Do modulu ESMII vstupuje ground-truth obraz, hranová reprezentace vytvořená pomocí Cannyho hranového detektoru, strukturální reprezentace a maska. Na výstupu modulu ESMII je vygenerovaný obraz. Převzato z [58].

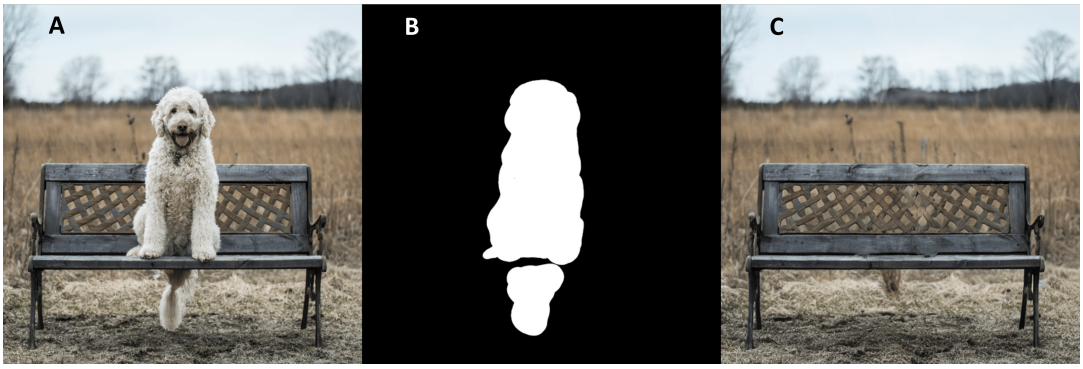
3.3.5 Stable difussion

Pro filtrační účely je možné použít i model stable diffusion [60, 61]. Jedná se o algoritmus strojového učení, který je založený na tzv. difuzních modelech [62]. Difuzní modely lze zařadit do skupiny generativní umělé inteligence, podobně jako třeba GANy. Hlavní myšlenka spočívá v použití Markovského řetězce, na základě kterého je do obrazu v jednotlivých difuzních krocích postupně přidáván náhodný šum. Poté nastane opačný proces, a to opětovné vygenerování vstupního obrazu ze šumu. Proces trénování difuzních modelů se tedy skládá ze 2 kroků - forward difussion a denoising difussion.

V případě použití stable difussion na úlohu filtrace, celý proces zjednodušeně probíhá tak, že na vstupu je obrázek, na kterém leží objekt, který chceme z obrazu odstranit. Dále potřebujeme masku daného objektu. V dalším kroku ovšem nastává rozdíl oproti předešlé metodě založené na biharmonických funkcích. Stable diffusion totiž využívá navíc textový vstup, do kterého uživatel napíše, čím chce vyplnit místo odstraněného objektu. Algoritmus už pak sám může vytvořit až několik možných výstupů, které respektují vstupní obrázek, masku a textový vstup uživatele (obr. 3.18). Pokud uživatel do textového vstupu nic nenapíše (zanechá prázdný string), pak se model stable difussion snaží vyplnit dané místo na základě okolí (obr. 3.19). Obecně lze tento přístup aplikovat zejména díky natrénování stable difussion na obrovském množství textů a obrázků, což mu umožňuje porozumět vztahu mezi slovy a vizuálními prvky.



Obrázek 3.18: Obrázek (A) reprezentuje vstupní obrázek, obrázek (B) je maska objektu, který chceme odstranit a výstupní obrázek (C). V tomto případě bylo jako textový vstup použito slovní spojení: *a robot sitting on a bench*. Vytvořeno prostřednictvím Colab notebooku [63].

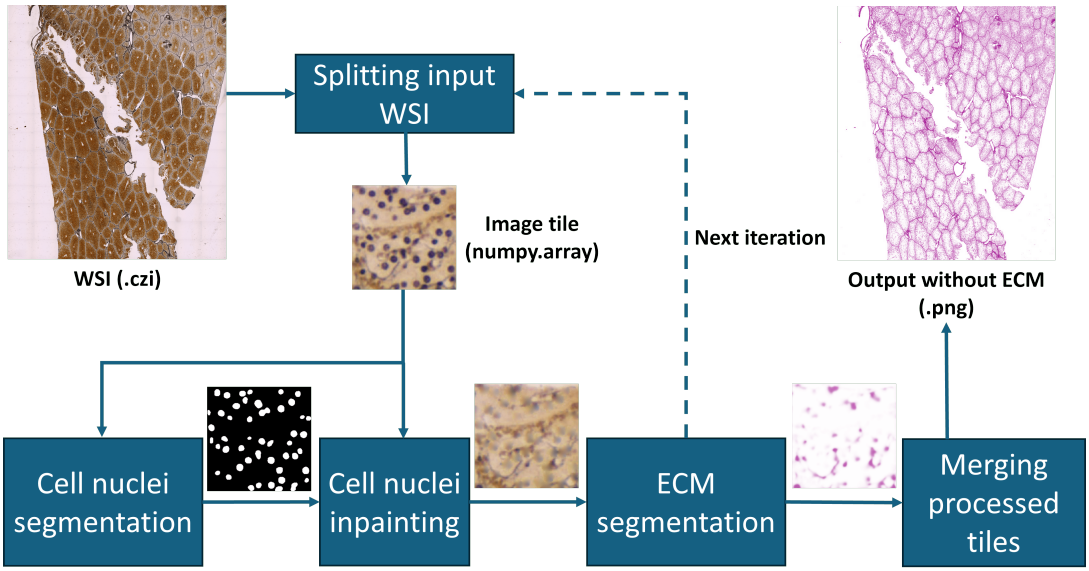


Obrázek 3.19: Obrázek (A) reprezentuje vstupní obrázek, obrázek (B) je maska objektu, který chceme odstranit a obrázek (C) je výstupním obrazem, na kterém je patrné úplné odstranění objektu. V tomto případě byl jako textový vstup použit prázdný textový řetězec. Vytvořeno prostřednictvím Colab notebooku [63].

4. Návrh metody pro filtraci histologického snímku

Hlavním cílem navržené metody (obr. 4.1) bylo vytvoření přístupu pro filtraci mikroskopického histologického obrazu. Zjednodušeně řečeno, vstup metody je tvořen histologickým obrazem (WSI) ve formátu .czi a na výstupu chceme získat obraz s vyextrahovanou mezibuněčnou hmotou. Zajímá nás pouze struktura vstupního obrazu. Naším hlavním cílem bylo se co nejvíce, s výstupem námi navržené metody, vizuálně přiblížit scaffoldu obarvenému pomocí H&E barvení (2.2B).

Z důvodu paměťové náročnosti vstupního .czi souboru byl tento soubor zpracováván po jednotlivých čtvercových „dlaždicích“. V každé dlaždici byla nejprve segmentována buněčná jádra buď pomocí Mask R-CNN (instance segmentation), anebo pomocí U-Netu (semantic segmentation). Výstupem segmentace byla vždy binární maska, jejíž bílé pixely značily segmentovaná buněčná jádra, pokud je dlaždice obsahovala. Tuto masku jsme společně s původní dlaždicí použili pro filtraci/inpainting jader pomocí biharmonické funkce. Následně jsme provedli segmentaci mezibuněčné hmoty s použitím U-Netu. Zpracovanou dlaždici jsme uložili do seznamu a pokračovali zpracováním další dlaždice. Po zpracování všech dlaždic, jsme z jednotlivých dlaždic obsažených v seznamu poskládali finální obraz.



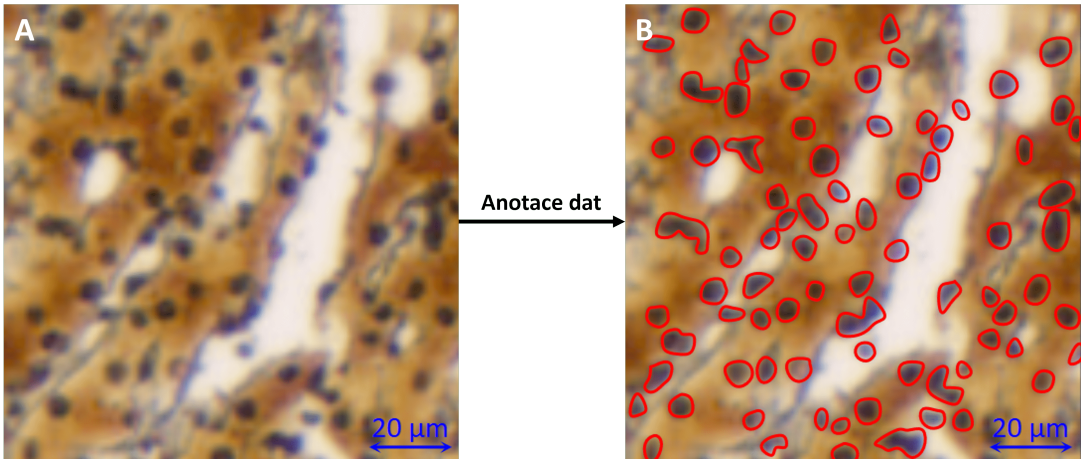
Obrázek 4.1: Schéma navržené metody. Vstup metody je tvořen mikroskopickým obrazem. Vstupní obraz je následně zpracováván po jednotlivých dlaždicích. V každé dlaždici jsou nejprve segmentována buněčná jádra buď pomocí Mask R-CNN, anebo U-Netu. V dalším kroku dojde k filtraci buněčných jader s využitím biharmonické funkce. Následně se uskuteční segmentace mezibuněčné hmoty pomocí U-Netu. Na závěr, po zpracování všech dlaždic, proběhne jejich poskládání do finálního obrazu.

4.1 Anotace dat

V našem případě lze anotovaná data rozdělit do dvou skupin: buněčná jádra a mezibuněčná hmota (extracelulární matrix). Anotace dat byla klíčová pro natrénování či fine-tuning modelů strojového učení, ať už se jednalo o Mask R-CNN či U-Net, jelikož jsme zvolili strategii učení s učitelem.

4.1.1 Buněčná jádra

Anotování jader jaterních buněk probíhalo ručně v softwaru ZEISS ZEN (obr. 4.2). Volba tohoto programu byla dána proprietárním formátem .czi, ve kterém byly mikroskopické snímky poskytnuty. ZEISS ZEN umožňuje uživateli anotovat data pomocí obdélníků, kruhů a beziérových křivek. Pro anotaci buněčných jader byly zvoleny beziérovy křivky, a to s ohledem na další zpracování anotovaných dat. Celkem bylo tímto způsobem anotováno 100 vyříznutých obrazů z několika velkých poskytnutých mikroskopických histologických obrazů.

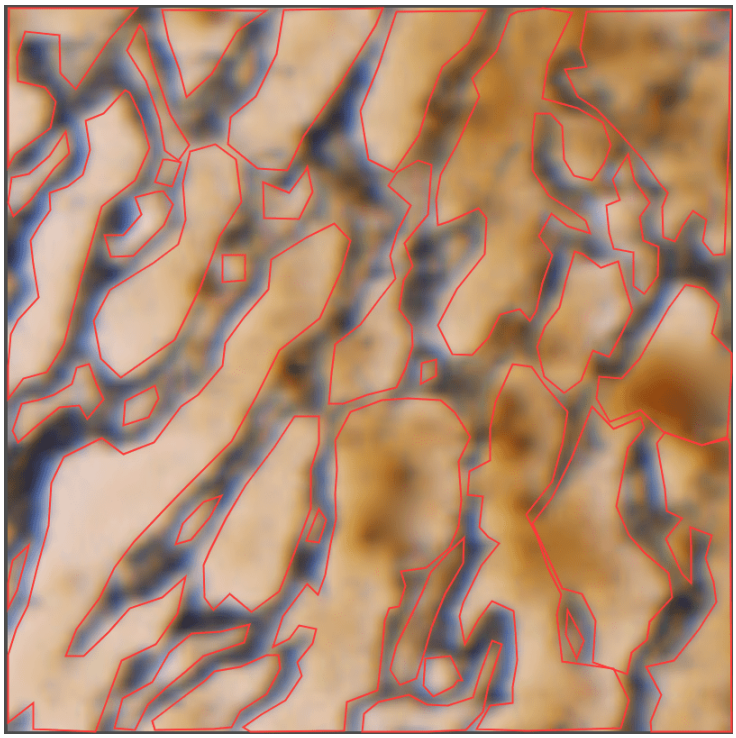


Obrázek 4.2: Proces anotace jader buněk v programu ZEISS ZEN. Na obr. (A) jsou vstupní neanotovaná jádra buněk, na obr. (B) jsou výstupní anotovaná jádra buněk.

4.1.2 Mezibuněčná hmota

Pro získání anotací mezibuněčné hmoty (obr. 4.3) jsme využili webový nástroj Make Sense [64]. Jako výchozí data jsme použili obrazová data ve formátu .png s již odstraněnými buněčnými jádry. Pomocí polygonů jsme následně vyznačili oblasti představující mezibuněčnou hmotu, kterou jsme chtěli potenciálně extrahovat. Na závěr jsme vytvořené anotace exportovali ve formátu COCO .json [65]. Tato funkcionalita je přímo implementována v nástroji Make Sense. Formát se skládá ze souboru ve formátu .json (získaný exportovaný soubor) a obrázků ve formátu .jpg (vstupní obrazy). V .json souboru jsou obsaženy informace týkající se datasetu např. základní

informace o datasetu (informace o datasetu, kdo je autorem, datum, atd.), seznam příslušných obrázků (jejich přiřazené IDs, rozměry a názvy). V poslední řadě lze v tomto souboru nalézt údaje spojené s jednotlivými anotacemi. Zmiňme např. IDs anotací, k jakému obrázku patří daná anotace či kde v obraze se anotace nachází. Nemůže chybět ani informace o třídě jakou daná anotace reprezentuje. V případě naší úlohy však anotace reprezentuje pouze jednu třídu (*ECM*). Exportovaný .json soubor společně s obrázky nám následně posloužil pro natrénování U-Net modelu pro segmentaci mezibuněčné hmoty.



Obrázek 4.3: Anotace mezibuněčné hmoty.

4.2 Převedení anotací jader na COCO formát

Na základě anotovaných buněčných jader byl následně vytvořen dataset ve formátu COCO. K převodu anotovaných buněčných jader na COCO formát byl použit námi vytvořený nástroj, který je možné prakticky používat prostřednictvím příkazové řádky. Vlastní nástroj jsme museli vytvořit zejména z důvodu formátu .czi, ve kterém jsou uložena vstupní data. Pro spuštění nástroje je třeba spustit pomocí příkazu *python* soubor `main.py` a společně s tím doplnit 2 vstupní parametry. Prvním vstupním parametrem je cesta k adresáři obsahující soubory ve formátu .czi, druhým vstupním parametrem je cesta, kam chceme uložit získaný COCO formát. Repozitář nástroje pro převod .czi souborů na COCO formát je veřejně dostupný na GitHubu [66]. Námi navržený nástroj využívá i některé moduly z knihovny ScaffAn [10]. Repozitář k této knihovně je taktéž veřejný [67].

4.3 wsitools

Whole-slide imaging tools (*wsitools*) je námi vytvořená knihovna programovacího jazyka Python sloužící pro práci s mikroskopickými histologickými obrazy. Knihovna *wsitools* používá několik modulů z aplikace ScaffAn, tyto moduly se starají především o převod z .czi formátu na standardní Python NumPy formát. Dále lze pomocí těchto modulů vybírat určité oblasti z mikroskopických obrazů.

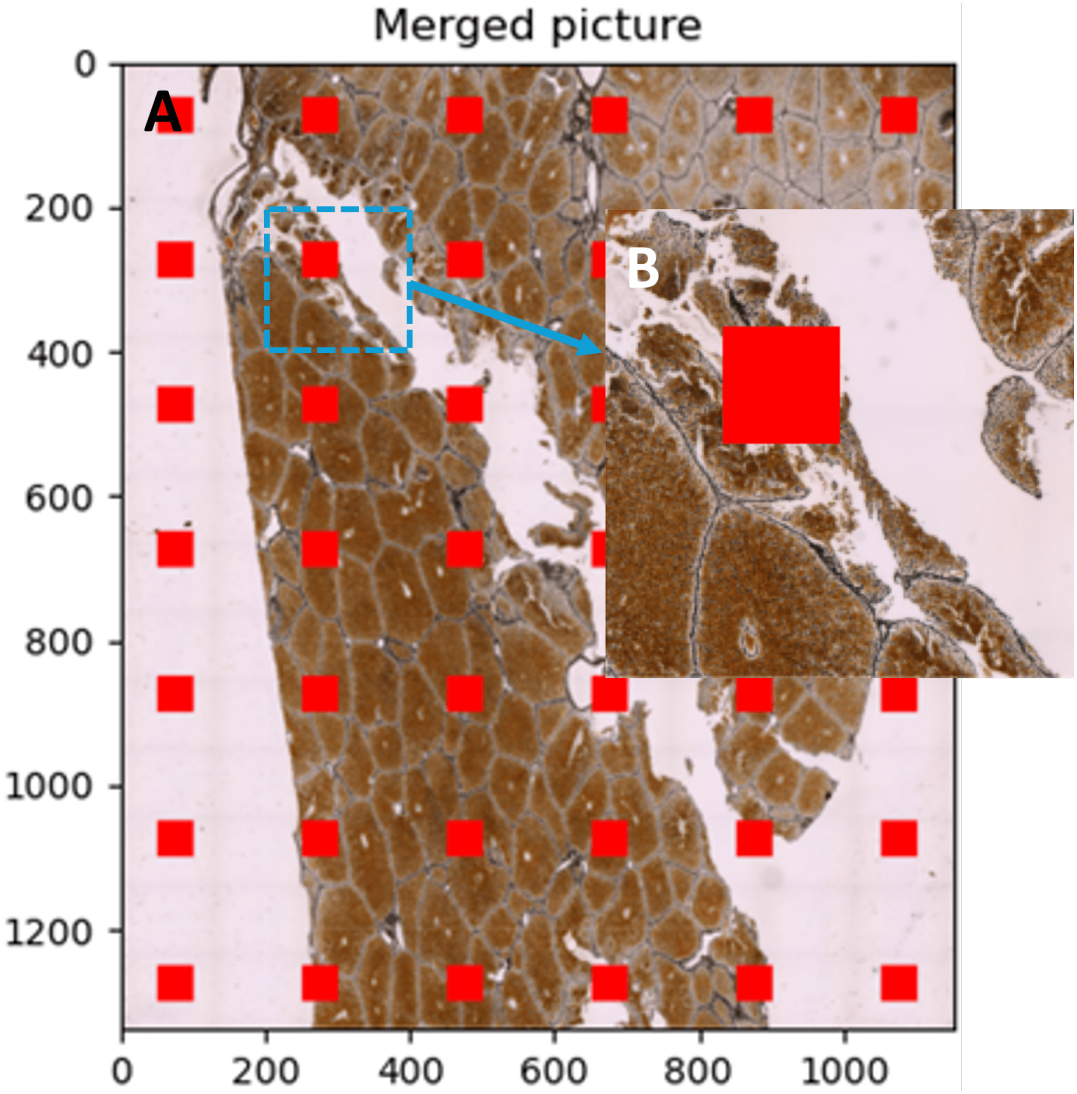
Knihovna *wsitools* vznikla především kvůli paměťové náročnosti práce s mikroskopickými histologickými obrazy. Součástí této knihovny je námi vytvořený modul `tile_image.py`, který obsahuje třídu ImageSplitterMerger. S využitím metod z této třídy je možné rozdělit soubor ve formátu .czi na jednotlivé dlaždice. Výhodou tohoto přístupu je postupné individuální zpracování dlaždic a tím pádem i snížení paměťové náročnosti, v porovnání s přímým zpracováním celého vstupního .czi obrazu. Soubor ve formátu .czi může mít totiž velikost až několik GB. Zpravidla totiž obsahuje medicínský obraz s velmi vysokým rozlišením, který nelze celý načíst do paměti RAM. Na každou dlaždici mohou být aplikovány metody např. z oblasti počítačového vidění. Po aplikaci metody či metod na jednotlivé dlaždice, jsou tyto dlaždice opět poskládány, tak aby tvořily jeden obraz. Složený obraz je již ve standardním Python NumPy formátu, který lze převést např. na formát .png. Co se týče reálného použití této funkcionality, je třeba nejprve vytvořit instanci třídy ImageSplitterMerger (4.1). Při definování instance třídy je třeba zadat následující atributy:

- `path_to_czi` – cesta k souboru ve formátu .czi
- `tilesize_px` – velikost dlaždice v pixelech
- `overlap_px` – velikost překrytí dlaždice v pixelech na všech stranách, např. v případě dlaždice o velikosti 200×200 a překryvu o velikosti 10 pixelů je na pozadí ve skutečnosti pracováno s dlaždicí o rozměrech 220×220, což je v našem případě výhodné především pro segmentaci mezibuněčné hmoty pomocí U-Netu; jinými slovy řečeno, kladná hodnota překrytí umožní neuronové síti nastavit správné hodnoty krajních pixelů v původní dlaždici na základě sousedních pixelů ze zvětšené dlaždice
- `pixelsize_mm` – velikost pixelu v mm, např. [0.01, 0.01]
- `fcn` - metoda, kterou si sám uživatel definuje; tato metoda slouží ke zpracování jedné dlaždice

Listing 4.1: Vytvoření instance třídy ImageSplitterMerger

```
image = ImageSplitterMerger(path_to_czi, tilesize_px=200,
    overlap_px=0, pixelsize_mm=[0.01, 0.01], fcn=process_tile)
```

Příklad implementace metody *process_tile()* je součástí příloh (A.1). Metoda *process_tile()* v tomto případě nakreslí červený čtverec do určitého místa do každé zpracovávané dlaždice (obr. 4.4B). Celý obraz vzniklý spojením všech dlaždic je pak zobrazen na obr. 4.4A. Tohoto dlaždicového zpracování jsme následně využívali v další části. Repozitář se zdrojovými kódy je veřejně dostupný na GitHubu [68].

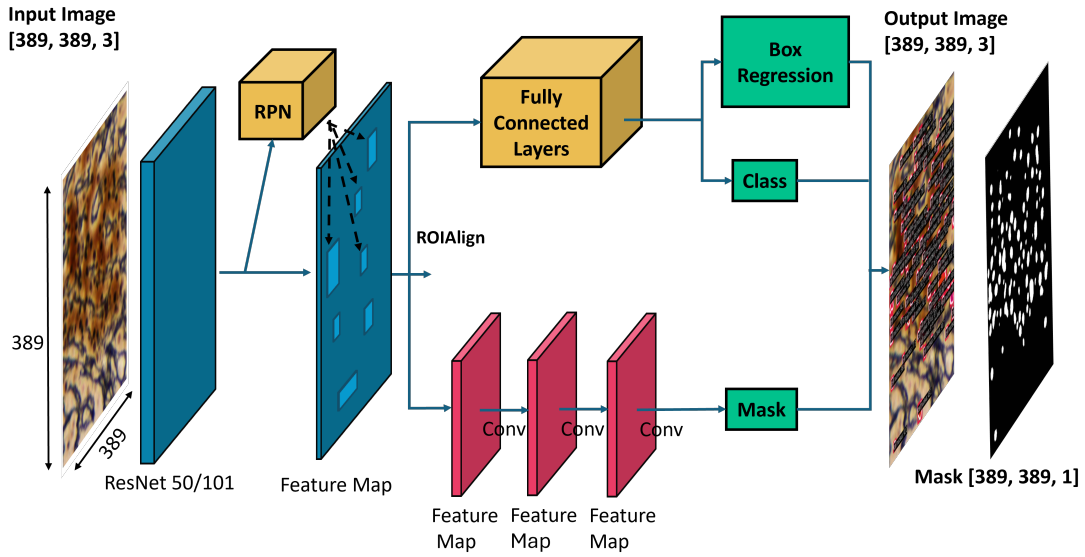


Obrázek 4.4: Obrázek A znázorňuje obraz vzniklý spojením individuálně zpracovaných dlaždic. V obrázku A je navíc modrým čtvercem zvýrazněna dlaždice, která je zobrazena na obr. B.

4.4 Segmentace buněčných jader pomocí Mask R-CNN

Pro segmentaci buněčných jader jsme nejprve vyzkoušeli segmentační neuronovou síť Mask R-CNN. V tomto případě se jednalo o úlohu instance segmentation. Výstup této sítě byl tvořen bounding boxem, třídou (v našem případě pouze jedna třída - *cell nucleus*) a maskou. Následně jsme původní výstup neuronové sítě převedli na binární masku. Pro snazší práci s touto segmentační neuronovou sítí jsme použili framework Detectron2 [69]. Zjednodušená architektura sítě je zobrazena na obrázku 4.5. V prvním kroku jsme dataset, skládající se ze 100 obrazů, ve formátu COCO rozdělili do 3 následujících podmnožin:

- Trénovací podmnožina (80 % datasetu)
- Validační podmnožina (10 % datasetu)
- Testovací podmnožina (10 % datasetu)



Obrázek 4.5: Zjednodušené schéma architektury modelu Mask R-CNN. Vytvořeno podle schématu obsaženého v [70].

4.4.1 Fine-tuning modelů

K fine-tuningu jsme vybrali již předtrénované modely Mask R-CNN z Detectron2 Model ZOO [71]. Vybrané modely jsou obsaženy v tabulce 4.1. Čísla 50 a 101 v tomto kontextu značí počet vrstev u ResNetu (popřípadě ResNeXtu), který byl použit jako backbone. Dále modely využívají feature pyramid network (FPN). Tyto modely byly vybrány z důvodu vysokých dosažených hodnot AP (average precision) u bounding boxu a masky. Kombinaci ResNetu společně s FPN jsme zvolili z důvodu nejlepšího kompromisu mezi

rychlostí a přesností. V trénovací podmnožině se celkem nacházelo 2 790 instancí buněčných jader.

Tabulka 4.1: Tabulka obsahující vybrané předtrénované Mask R-CNN modely. Jedna epocha se v tomto případě skládá ze 118 000 obrázků z COCO datasetu. Tabulka vytvořena podle [71].

Name	COCO epochs	box AP	mask AP
R50-FPN	37	38.6	35.2
R101-FPN	37	42.9	38.6
X101-FPN	37	44.3	39.5

Pro fine-tuning obou modelů jsme zvolili následující parametry trénování:

- `cfg.DATALOADER.NUM_WORKERS = 8`
- `cfg.SOLVER.IMS_PER_BATCH = 8`
- `cfg.SOLVER.BASE_LR = 0.0025`
- `cfg.SOLVER.MAX_ITER = 500`
- `cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 128`
- `cfg.MODEL.ROI_HEADS.NUM_CLASSES = 1`

4.4.2 Vyhodnocení modelů

Pro vyhodnocení modelů jsme použili metriku IoU (Intersection over Union), známou také jako Jaccardův koeficient, spočtenou pro jednotlivé bounding boxy a masky. Hodnotu IoU lze obecně vypočítat pomocí následujícího vzorce (4.1):

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}. \tag{4.1}$$

Vyhodnocení proběhlo na testovací podmnožině, která obsahovala 469 instancí. Dosažené výsledky jsou obsažené v tabulkách 4.2 a 4.3. Výsledky IoU jednotlivých modelů byly získány s využitím třídy COCOEvaluator, která je součástí frameworku Detectron2. AP (Average Precision) byla v tomto případě vypočítána pro různé hodnoty IoU v rozmezí od 0.5 do 0.95 s krokem 0.05. Jinými slovy, pro výpočet AP bylo použito celkem 10 různých prahových hodnot IoU (0.50; 0.55; 0.60; ..., 0.90; 0.95). Pro každou prahovou hodnotu IoU byla vypočítána přesnost a z těchto 10 hodnot byla následně vypočtena průměrná hodnota (= finální hodnota AP). V případě výpočtu AP50 či AP75 byla použita pouze jedna prahová hodnota 0.5, respektive 0.75. Na obrázku 4.6 je možné porovnat ground truth anotace s výsledným segmentovaným obrazem.

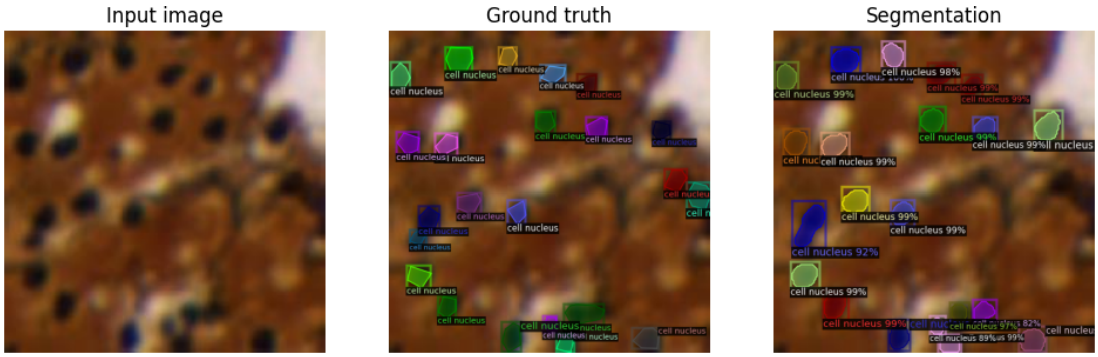
Tabulka 4.2: Tabulka znázorňující získané výsledky na testovacích datech (pro bounding boxy).

Name	AP	AP50	AP75
R50-FPN	30.034	58.684	24.523
R101-FPN	**33.080**	**66.832**	**25.636**
X101-FPN	29.711	66.432	20.249

Tabulka 4.3: Tabulka znázorňující získané výsledky na testovacích datech (pro masky).

Name	AP	AP50	AP75
R50-FPN	30.751	59.586	27.223
R101-FPN	**33.786**	66.861	**33.246**
X101-FPN	32.738	**67.081**	27.074

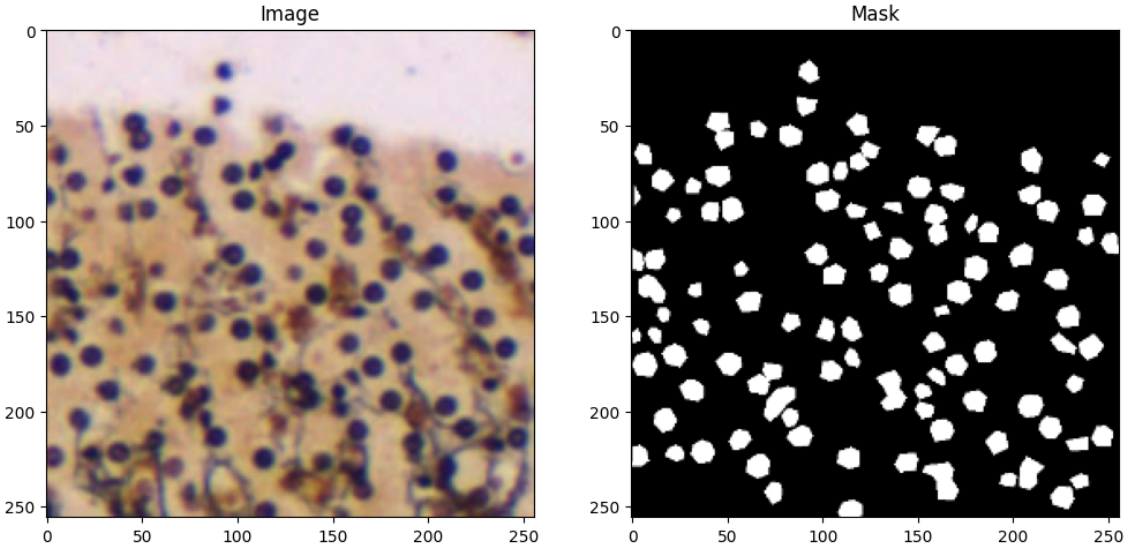
Na základě tabulek 4.2 a 4.3 lze konstatovat, že jako nejlepší model pro naši úlohu, tj. segmentace buněčných jader, se jeví ResNet101-FPN, který dosáhl nejvyšších hodnot téměř ve všech případech. Pouze v případě metriky AP50 u segmentační úlohy byl o něco lepší model ResNeXt101-FPN.



Obrázek 4.6: Obrázek vlevo reprezentuje vstupní obraz z testovací množiny, obrázek vpravo představuje obrázek s ground truth anotacemi, na obrázku vpravo jsou označená jednotlivá segmentovaná buněčná jádra. Ground truth obraz a výsledný obraz byly dvojnásobně zvětšeny pro lepší vizualizaci.

4.5 Segmentace buněčných jader pomocí U-Netu

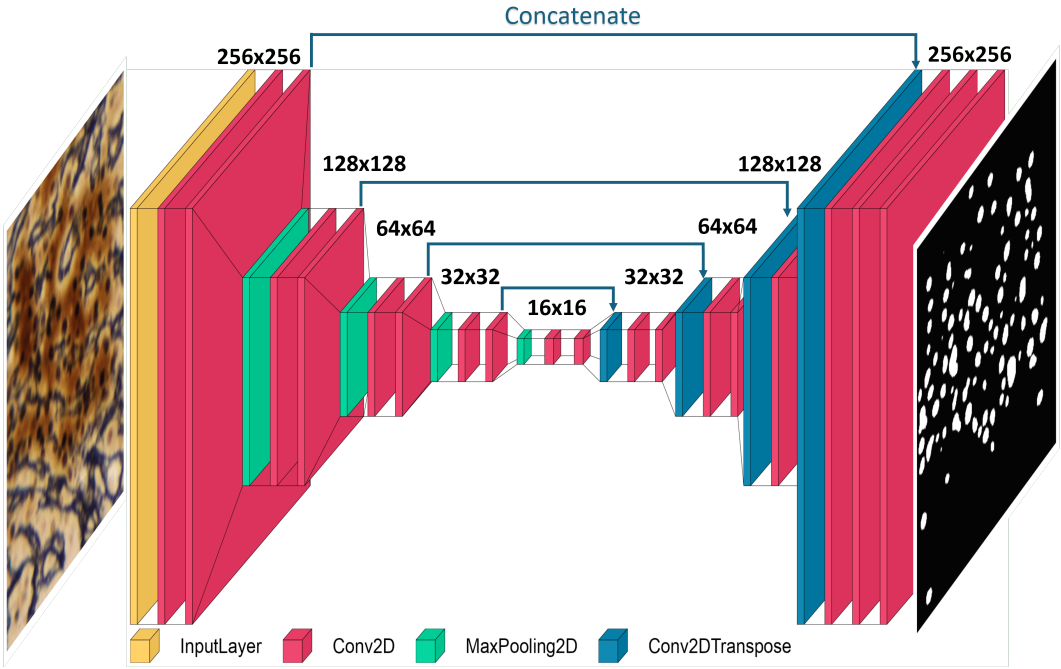
Segmentační síť U-Net jsme se rozhodli použít zejména z důvodu jeho určení pro segmentaci medicínských obrazů. Na rozdíl od Mask R-CNN, kde vstup byl kompletně tvořen pomocí datasetu ve formátu COCO, bylo třeba v případě U-Netu navíc vytvořit z .json souboru příslušné binární masky jednotlivých obrázků (4.7), pomocí kterých byly reprezentovány anotovaná data.



Obrázek 4.7: Obrázek vlevo reprezentuje původní vstupní obraz, obrázek vpravo představuje příslušnou binární masku. Velikost vstupního obrazu i masky byla změněna na 256×256 pixelů.

4.5.1 Architektura modelu

Námi převzatá veřejná architektura modelu U-Net (obr. 4.8) [72] byla vytvořena pomocí knihovny *Keras* (standardní knihovna programovacího jazyka Python). Použitá architektura byla, oproti architektuře původního U-Net modelu (obr. 3.9), modifikována. Na vstupu je barevný obraz o dimenzích $256 \times 256 \times 3$ a na výstupu je binární maska o dimenzích $256 \times 256 \times 1$. V první části (enkodéru) dochází k postupnému snižování rozlišení obrazu a ke zvyšování počtu kanálů. Ve druhé části (dekodéru) dochází naopak k upsamplingu feature map (ke zvýšení rozlišení). Architektura se skládá především z konvolučních vrstev (Conv2D, Conv2DTranspose) či max-pooling vrstev (MaxPooling2D). Dále byl v architektuře opakovaně použit dropout a konkatenace. Kompletní architektura je pak zachycena v tabulce A.1, která je součástí příloh.



Obrázek 4.8: Schéma architektury použitého U-Net modelu. Pro jednoduchost byly vynechány dropout vrstvy mezi konvolučními vrstvami. Žlutá barva je vstupní vrstva (InputLayer), červené boxy představují konvoluční vrstvy (Conv2D), zelené boxy reprezentují operaci/vrstvu max-pooling (MaxPooling2D) a modře znázorněné jsou transponované konvoluční vrstvy (Conv2DTranpose). Část schématu bylo vytvořeno pomocí knihovny *visualkeras* [73].

4.5.2 Trénování modelu

Trénování modelu bylo experimentálně nastaveno na 150 epoch, velikost batche byla nastavena na 10. Jako optimizer jsme zvolili Adam optimizer. Dataset byl ještě před samotným trénováním rozdělen do 2 podmnožin:

- Trénovací podmnožina (90 % datasetu)
- Validační podmnožina (10 % datasetu)

Jako ztrátovou funkci jsme použili hodnotu tzv. Jaccardova koeficientu J, známého jinak také IoU (Intersection over Union), přenásobenou -1 (rov. 4.3). Hodnotu Jaccardova koeficientu lze obecně vypočítat pomocí vzorce (4.1). V našem případě jsme použili následující tvar rovnice 4.1:

$$J(y_{true}, y_{pred}) = \frac{\sum_{i=1}^{n} y_{true_i} y_{pred_i} + 1.0}{\sum_{i=1}^{n} y_{true_i} + \sum_{i=1}^{n} y_{pred_i} - \sum_{i=1}^{n} y_{true_i} y_{pred_i} + 1.0}, \tag{4.2}$$

$J(y_{true}, y_{pred})$ reprezentuje Jaccardův koeficient mezi ground truth labely y_{true} a predikovanými labely y_{pred}, suma v čitateli představuje počet správně predikovaných pixelů (intersection), první dvě sumy ve jmenovateli počítají celkový počet pozitivních labelů v ground truth obrazu, respektive v predikovaném obrazu, pomocí poslední sumy v čitateli dochází k odečtení

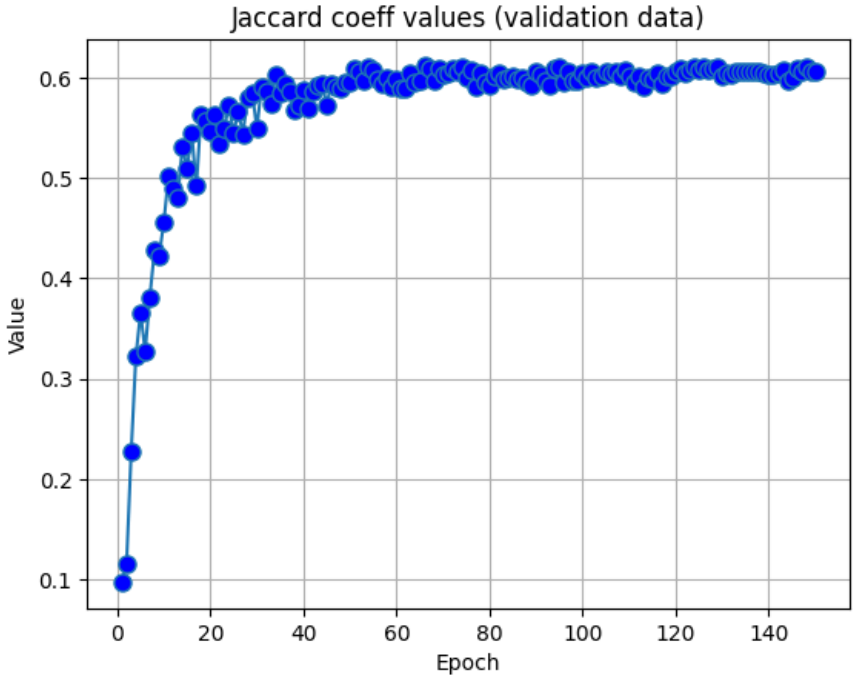
správně predikovaných pixelů, tak aby nedošlo ke dvojitému započítání, +1.0 v čitateli i jmenovateli slouží jako vyhlazovací faktor (vyhnutí se dělení 0).

Z toho následně vyplývá vzorec pro výpočet ztrátové funkce L:

$$L(y_{true}, y_{pred}) = -J(y_{true}, y_{pred}). \tag{4.3}$$

4.5.3 Vyhodnocení modelu

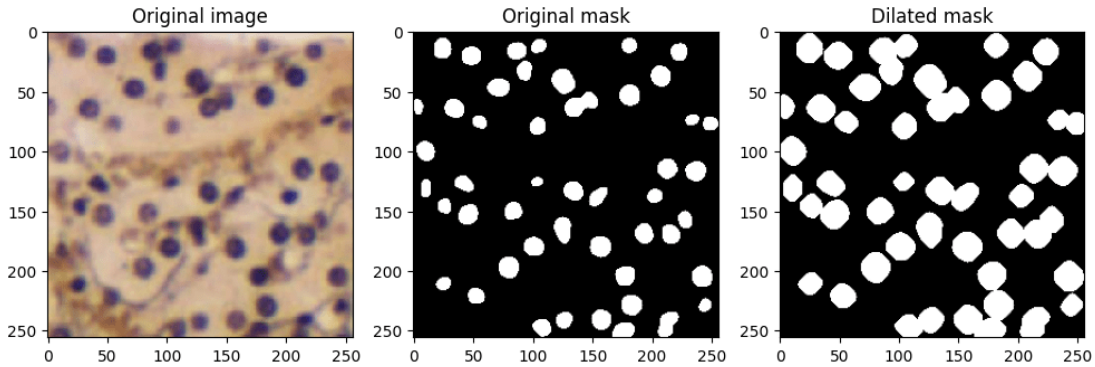
Úspěšnost modelu byla vyhodnocována pomocí již zmíněného Jaccardova koeficientu (rov. 4.2). Tato metrika byla zvolena zejména z toho důvodu, že buněčná jádra zabírají v obraze velice málo prostoru (velká část našeho obrazu je tvořeno pozadím), kvůli tomu jsme se vyhnuli použití např. metriky accuracy. Na obrázku 4.9 jsou zřetelné dosažené hodnoty Jaccardova koeficientu na validačních datech během trénování.



Obrázek 4.9: Graf znázorňující průběh hodnot Jaccardova koeficientu. Se zvyšujícím se počtem epoch dochází zároveň ke zpřesnění IoU hodnot. Na validačních datech se podařilo dosáhnout maximální hodnoty IoU kolem 0.6.

4.6 Odstranění buněčných jader

Pro odstranění buněčných jader jsme použili biharmonickou funkci z knihovny *scikit-image* (obr. 4.11b). Vstupní data představoval původní obraz s jádry buněk (obr. 4.11a) a binární maska získaná ze segmentace. Před aplikací biharmonické funkce jsme na masku opakovaně aplikovali dilataci, matematickou morfologickou operaci zvětšující plochu masky. Vstupní obraz, binární maska a dilatovaná binární maska jsou zobrazeny na obr. 4.10.

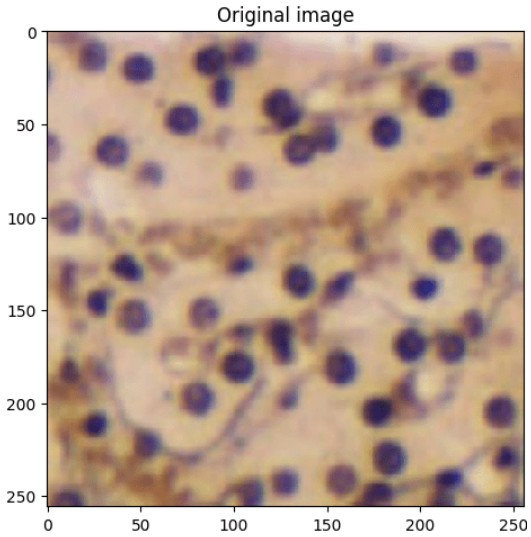


Obrázek 4.10: Obrázek nejvíce vlevo reprezentuje původní vstupní obraz, uprostřed se nachází příslušná binární maska získaná ze segmentace. Na obrázku vpravo je $4\times$ dilatovaná příslušná maska.

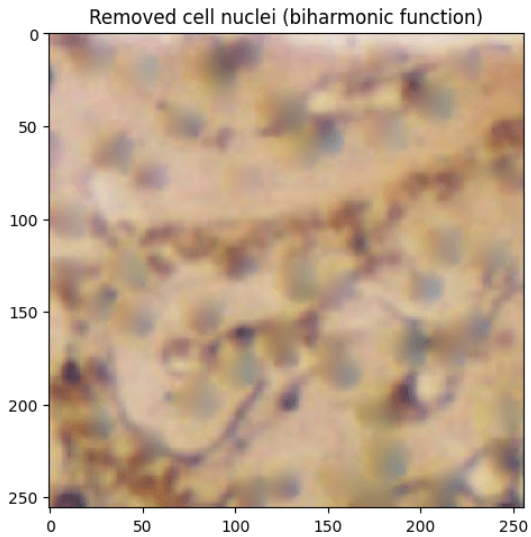
Dále jsme testovali i vlastní, jednodušší metody inpaintingu, které se ukázaly jako méně efektivní. Jedna z nich pracovala s hodnotami pixelů v bezprostředním okolí buněčných jader, přičemž buněčnému jádru jsme zkusili přiřadit průměrnou hodnotu RGB barevného odstínu vypočtenou na základě hodnot sousedních pixelů, které jsme určili na základě diferenční masky (rozdíl mezi jednou dilatovanou maskou a původní maskou) (obr. 4.11c). Průměrnou RGB hodnotu jsme zkusili určit i z Gaussovsky rozmazaného obrazu s parametry $\sigma = 100$, čímž jsme způsobili poměrně velké rozmazání vstupního obrazu, a $truncate = \frac{1}{5}$ (= parametr určující, jak velká část Gaussovské distribuce se má použít pro jádro filtru). K tomuto účelu jsme použili metodu ze standardní knihovny *skimage/scikit-image* (skimage.filters.gaussian()). Obdržený výsledek byl však velice podobný výsledku předcházející vlastní metody.

Na závěr jsme vyzkoušeli tuto problematiku řešit pomocí přístupu stable diffusion (obr. 4.11d), která bohužel v této úloze většinou nepřinesla uspokojivé výsledky. Model stable-diffusion-inpainting, který jsme použili, zřejmě nebyl natrénován na tomto typu dat, a proto nezafungoval tak dobře jako v případě na obrázku 3.19. Problém navíc vidíme ve stochastičnosti této metody, kdy můžeme u jednoho vstupního obrazu či u různých vstupních obrazů dostat pokaždé trochu jiný výsledek. Jinými slovy, ačkoliv tento způsob může v praxi teoreticky zafungovat u dílčích případů, nelze s jistotou prohlásit, že bude spolehlivý ve všech situacích. Z tohoto důvodu jsme na-

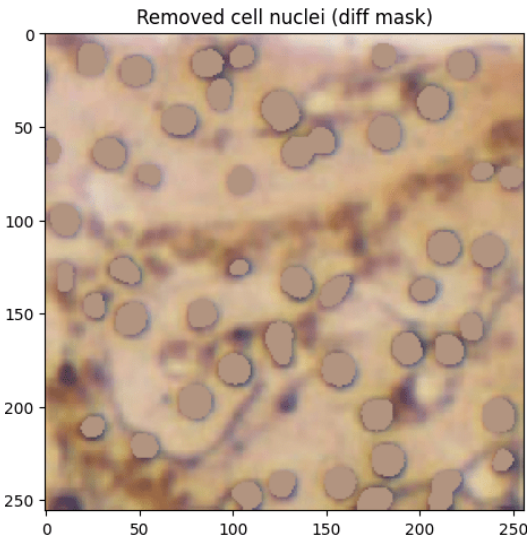
konec v naší metodě zvolili k odstraňování buněčných jader biharmonickou funkci, která je deterministická.



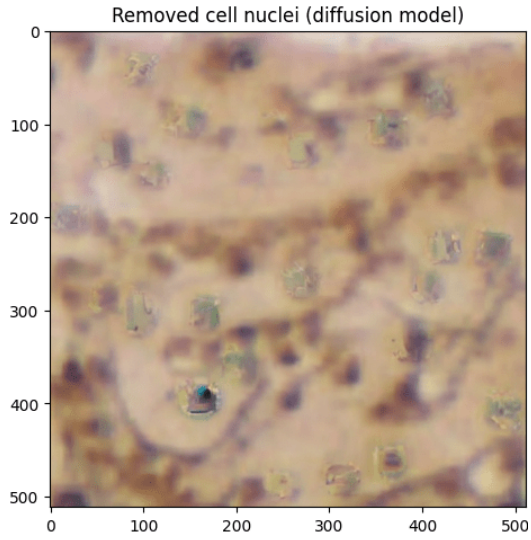
(a) Původní vstupní obraz.



(b) Výstupní obraz (biharm. funkce).



(c) Výstupní obraz (dif. maska).



(d) Výstupní obraz (diffusion model).

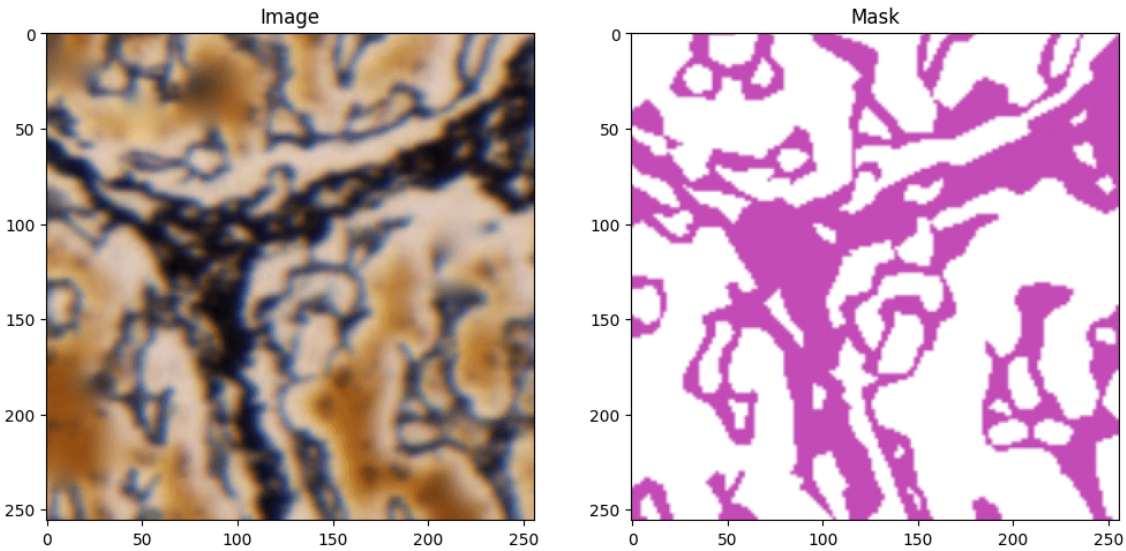
Obrázek 4.11: Vstupní obraz a jednotlivé výsledky odstranění buněčných jader.

4.7 Odstranění mezibuněčné hmoty

Po dokončení segmentace buněk (viz sekce 4.4 či 4.5) a odstranění buněčných jader (viz sekce 4.6) jsme se zaměřili na odstranění mezibuněčné hmoty (extracelulárního matrixu). První koncept spočíval ve využití barevných vlastností obrazu. Zkusili jsme obraz převést z RGB barevného modelu na HSV barevný model [37, 74] a tedy pracovat s odstínem (**H**ue), sytostí (**S**aturation) a hodnotou jasu (**V**alue) obrazu. S využitím těchto 3 atributů jsme definovali horní a dolní práh, díky čemuž jsme mohli vytyčit pásmo barevných odstínů, které jsme chtěli následně segmentovat. Tento postup však nevedl k příliš uspokojivým výsledkům. Nakonec jsme k tomuto účelu opět použili segmentační neuronovou síť U-Net, avšak s modelem tentokrát natrénovaným na datech s anotovanou mezibuněčnou hmotou.

4.7.1 Trénování modelu

Pro účely natrénování modelu byla použita stejná architektura (obr. 4.8) jako v případě 4.5. Tentokrát jsme však nevytvářeli binární masky, nýbrž RGB masky ke všem trénovacím obrazům (obr. 4.12), tak abychom se co nejvíce přiblížily chtěnému obrazu struktury. Kvůli tomu jsme tedy museli změnit třetí dimenzi finálního výstupu v U-Net architektuře. Výstup tedy vypadal následovně: $256 \times 256 \times 3$ (RGB obraz), namísto $256 \times 256 \times 1$ (černobílý obraz).



Obrázek 4.12: Obrázek vlevo reprezentuje původní vstupní obraz (obecně byly použity obrazy s již odstraněnými buněčnými jádry). Obrázek vpravo představuje příslušnou RGB masku. Velikost vstupního obrazu a rovněž masky byla změněna na 256×256 pixelů.

Před samotným trénováním byl dataset rozdělen do 2 podmnožin:

- Trénovací podmnožina (80 % datasetu)

- Validační podmnožina (20 % datasetu)

Trénování modelu bylo experimentálně nastaveno na 50 epoch, a na batch size o velikosti 5. Jako optimizer jsme zvolili Adam optimizer. Jako ztrátovou funkci jsme v tomto případě zvolili binary crossentropy, kterou lze definovat pomocí následujícího vzorce:

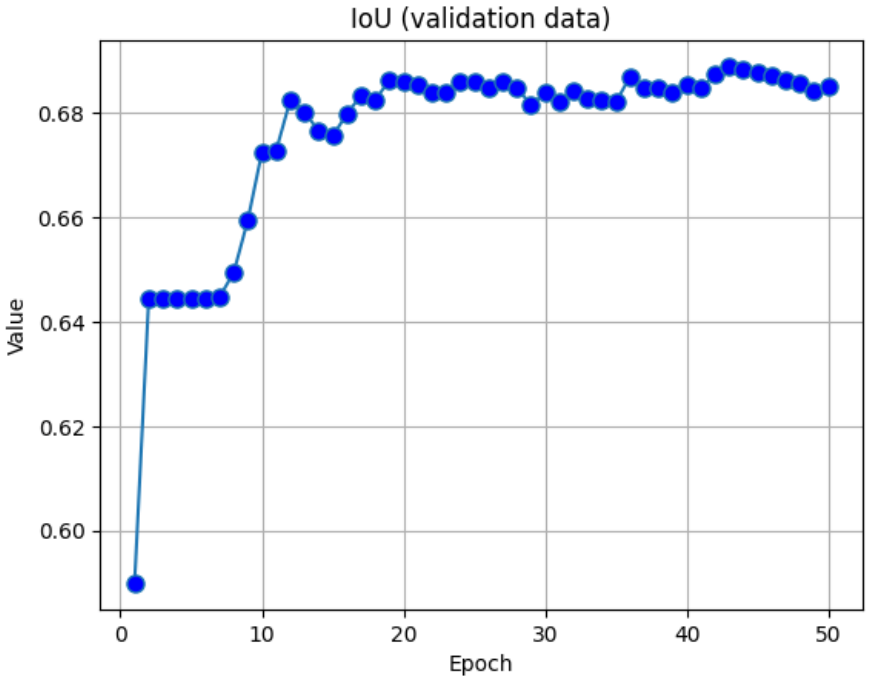
$$BCE = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p(y_i)) + (1 - y_i)\log(1 - p(y_i))], \tag{4.4}$$

kde y_i je aktuální label třídy pro data point i (nabývá hodnot 0 nebo 1), $p(y_i)$ je predikovaná pravděpodobnost data pointu i patřícího do třídy 1.

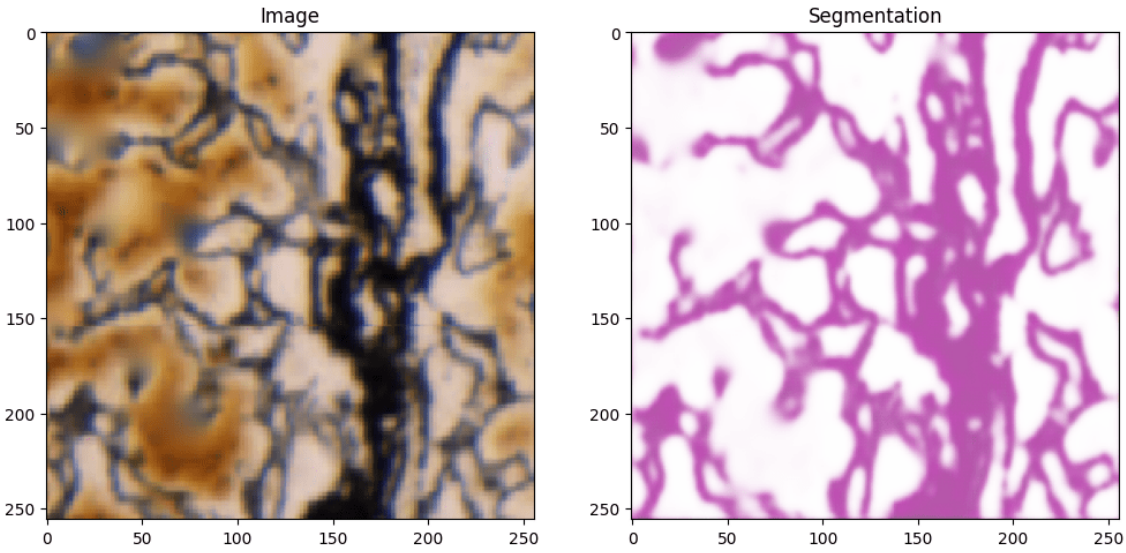
Tuto ztrátovou funkci jsme vybrali, jelikož v naší úloze potřebujeme pouze odlišit popředí od pozadí. V našem případě klasifikujeme každý pixel v obrázku do určité třídy (popředí nebo pozadí). Pomocí BCE tedy provádíme binární klasifikaci pro každý pixel a předpovídáme pravděpodobnost, že pixel patří do konkrétní třídy.

4.7.2 Vyhodnocení modelu

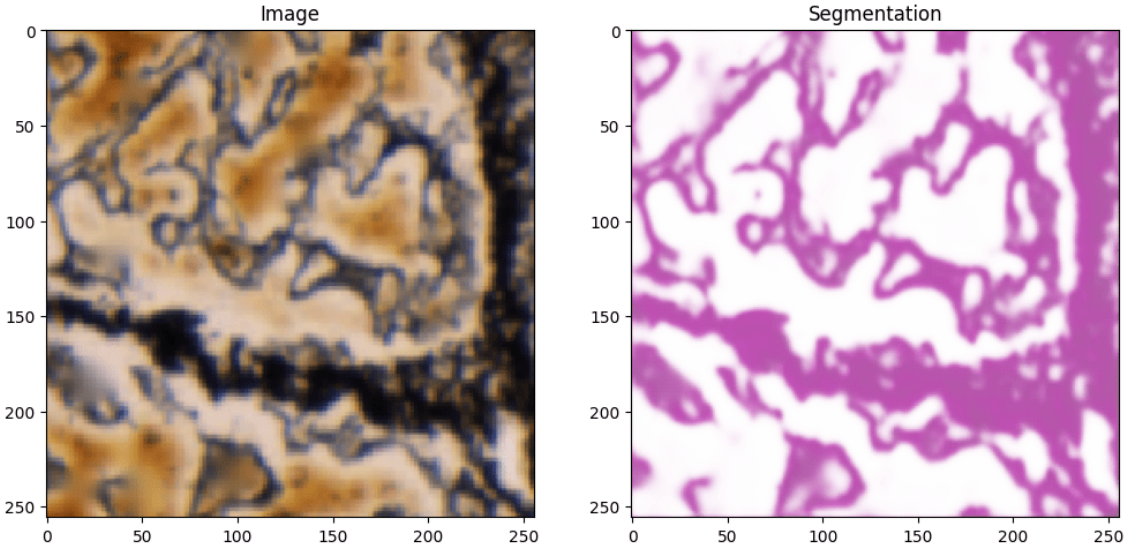
Model jsme vyhodnotili s použitím metriky IoU (rov. 4.1), přesněji BinaryIoU, jelikož jsme přímo implementovanou metriku v knihovně *Keras*. Na obrázku 4.13 jsou zřetelné dosažené hodnoty IoU na validačních datech během trénování. Podařilo se dosáhnout hodnoty IoU téměř 0.7. Dále jsme zkusili aplikovat natrénovaný model na testovací obraz (obr. 4.14 a 4.15).



Obrázek 4.13: Graf znázorňující průběh IoU hodnot. Na validačních datech se podařilo dosáhnout maximální hodnoty IoU kolem 0.68.



Obrázek 4.14: Testovací obrázek vlevo představuje vstup do našeho natrénovaného modelu. Obrázek vpravo pak reprezentuje příslušnou sémantickou segmentaci daného vstupního obrazu.



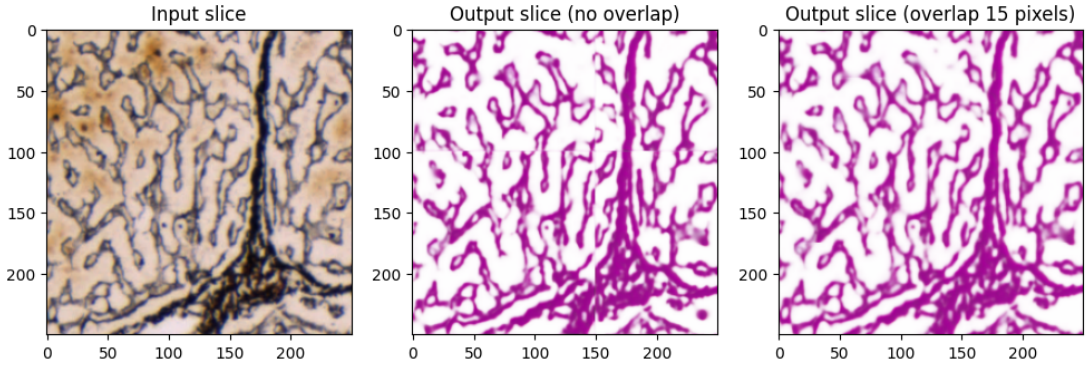
Obrázek 4.15: Testovací obrázek vlevo představuje vstup do našeho natrénovaného modelu. Obrázek vpravo pak reprezentuje příslušnou sémantickou segmentaci daného vstupního obrazu.

4.8 Hardwarové požadavky

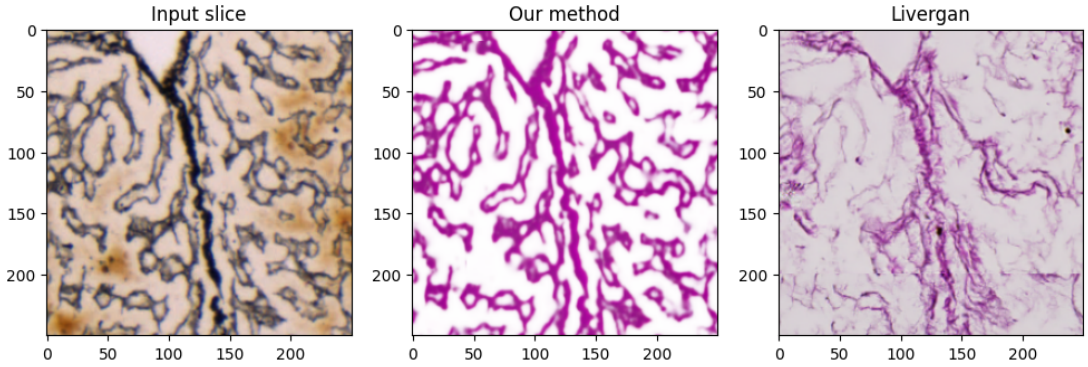
Jednotlivé získané modely byly natrénovány s využitím služby Google Colaboratory. Pro rychlejší trénování a inferenci bylo použito běhové prostředí s GPU Nvidia Tesla T4 (16 GB RAM) nebo Nvidia L4 (24 GB RAM) s CUDA verzí 12.2. Jako procesor používá Google Colaboratory Intel(R) Xeon(R) CPU @ 2.00GHz. Dále byly, v tomto vývojovém prostředí, vytvořeny rovněž pomocné sešity či hlavní sešit obsahující naimplementovanou metodu.

5. Ověření funkčnosti

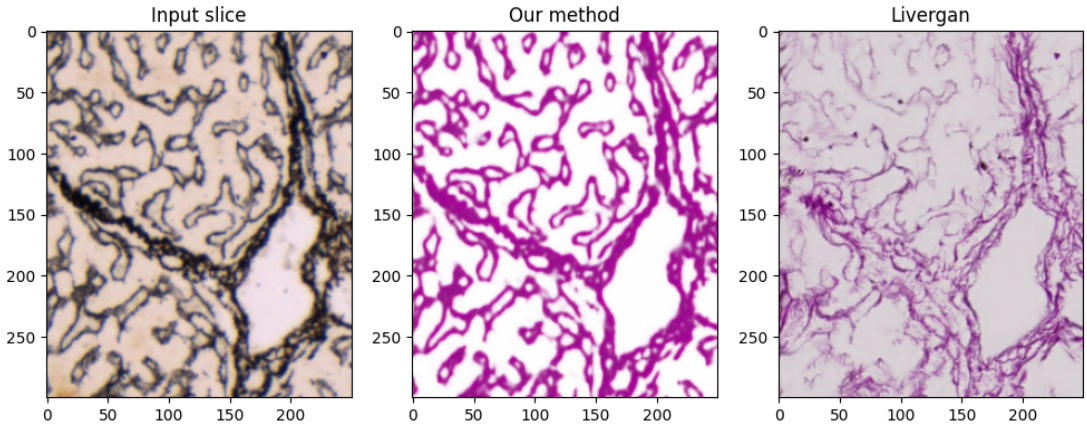
Pro ověření funkčnosti námi navržené metody jsme se rozhodli naše výsledky porovnat s výsledky metody navržené kolegou Václavem Javorkem [75], jehož metoda *livergan* [76] pro odstranění mezibuněčné hmoty je založena na použití GANů. Pro porovnání jsme vybrali výřezy z velkých vygenerovaných obrazů s `pixelsize_mm = [0.001, 0.001]`. V první části jsme se však rozhodli porovnat použití overlapu o námi navržené metody (obr. 5.1). Následně jsme se zaměřili na porovnání výsledků již zmíněných metod (obr. 5.2, 5.3, 5.4, 5.5).



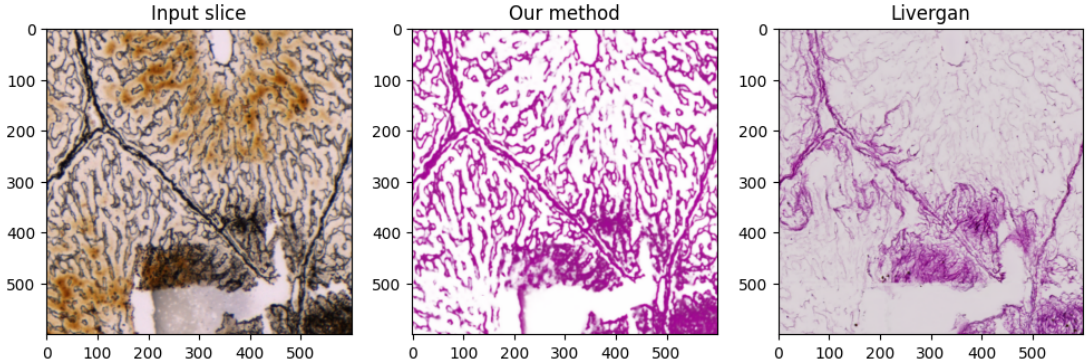
Obrázek 5.1: Obrázek vlevo reprezentuje zbarvení části původního obrazu, uprostřed se nachází výsledek námi navržené metody bez zadaného overlapu, nejvíce vpravo se pak nachází výsledek s definovaným overlapem o velikosti 15 pixelů.



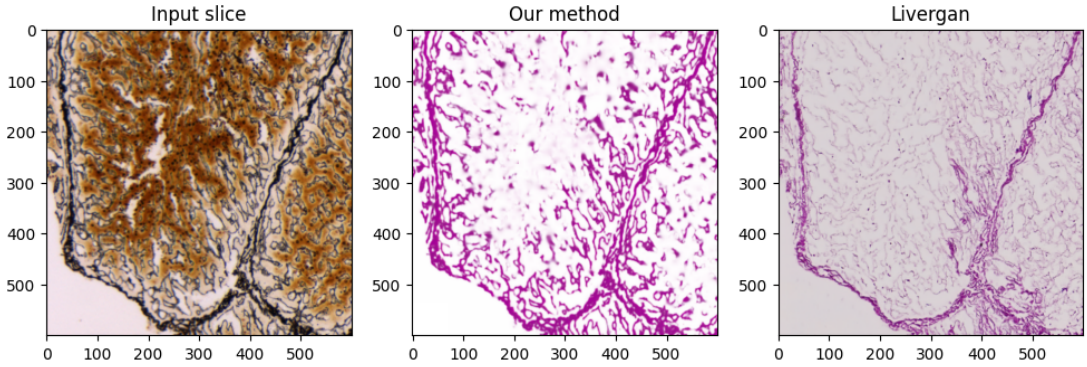
Obrázek 5.2: Obrázek vlevo reprezentuje zbarvení části původního obrazu, uprostřed se nachází výsledek námi navržené metody, nejvíce vpravo se pak nachází výsledek metody *livergan*.



Obrázek 5.3: Obrázek vlevo reprezentuje zbarvení části původního obrazu, uprostřed se nachází výsledek námi navržené metody, nejvíce vpravo se pak nachází výsledek metody *livergan*.



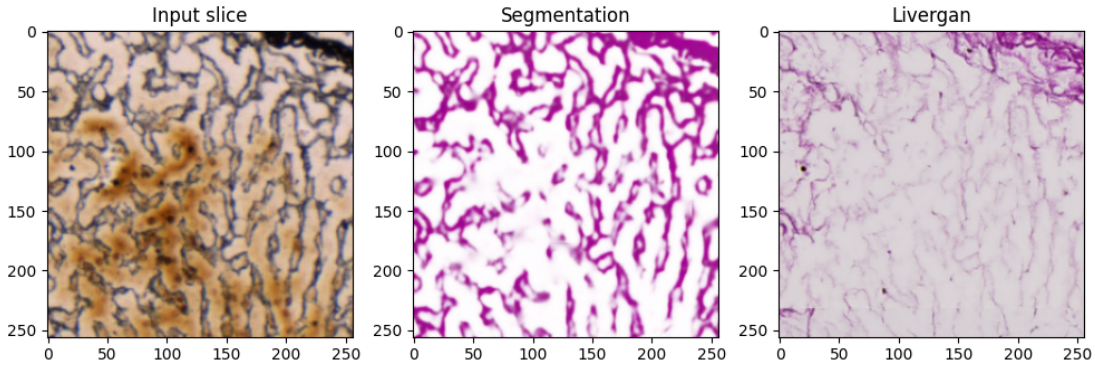
Obrázek 5.4: Obrázek vlevo reprezentuje zbarvení části původního obrazu, uprostřed se nachází výsledek námi navržené metody, nejvíce vpravo se pak nachází výsledek metody *livergan*.



Obrázek 5.5: Obrázek vlevo reprezentuje zbarvení části původního obrazu, uprostřed se nachází výsledek námi navržené metody, nejvíce vpravo se pak nachází výsledek metody *livergan*.

5.1 Vyhodnocení funkčnosti

Pro vyhodnocení funkčnosti námi navrženého řešení jsme se rozhodli odborníkům poskytnout celkem 470 náhodně vybraných obrázků z celkového dostupného počtu 6930, přičemž každý obrázek obsahoval další celkem 3 obrázky. Jeden z poskytnutých 470 obrázků je zobrazen obr. 5.6. První obrázek byl výřez původního vstupního obrazu, druhý obrázek byl tvořen výsledkem námi navržené metody a třetí obrázek byl výstup metody *livergan*. Výstupní obrázky obou metod byly získány jako výřezy o velikosti 256×256 z velkých výstupních obrazů. Odborníci měli za úkol rozhodnout, který výsledný obraz, získaný pomocí naší metody či *liverganu*, více odpovídá výřezu původnímu obrazu.

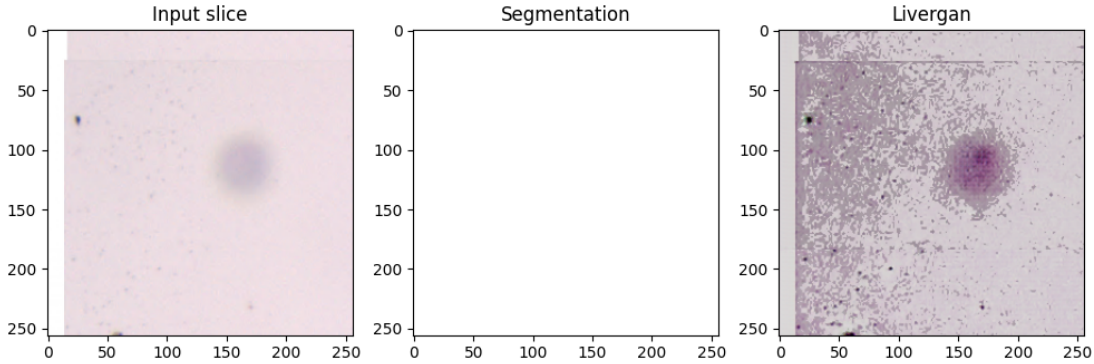


Obrázek 5.6: Jeden z poskytnutých 470 obrázků. Na prvním obrázku zleva je reprezentováno zbarvení části původního obrazu, uprostřed se nachází výsledek námi navržené metody, nejvíce vpravo se nachází výstup metody *livergan*.

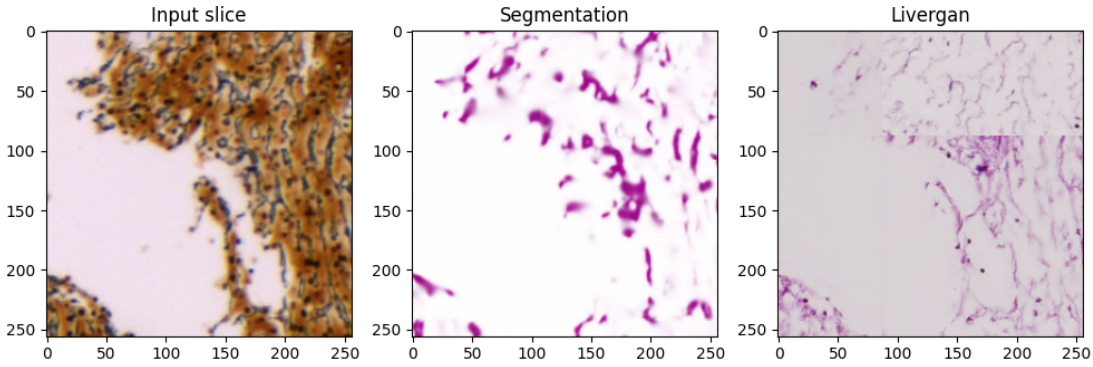
Pro účely hodnocení bylo po zpracování poskytnutých 470 obrázků odstraněno celkem 134 obrázků. Na 117 obrázcích se nic nenacházelo, jednalo se pouze o pozadí (obr. 5.7). Avšak u všech těchto obrázků, na kterých bylo pouze pozadí, dosáhla lepších výsledků námi navržená metoda. U dalších 17 bylo těžké určit, který přístup dosáhl lepšího výsledku, jelikož ani jedna z obou metod v těchto případech příliš nekorespondovala s reálným vstupním výřezem (obr. 5.8). Dále jsme tedy pracovali pouze s 336 obrázky. Získané výsledky byly zaneseny do tabulky 5.1. Celkem ve 226 případech, což odpovídá zhruba 67 % případů, se jevila jako lepší námi navržená metoda, v porovnání s *liverganem*.

Tabulka 5.1: Tabulka znázorňující získané výsledky. Ve druhém sloupci je číslo udávající, v kolika případech byla lepší daná metoda. Ve třetím sloupci je naopak číslo určující, v kolika případech byla daná metoda horší.

Method	Num better	Num worse
livergan	110	226
Our method	226	110



Obrázek 5.7: Jeden z vyřazených 117 obrázků, na kterém bylo pouze pozadí. Naše metoda ve všech těchto případech dosáhla lepších výsledků oproti *liverganu*. Na prvním obrázku zleva je reprezentováno zbarvení části původního obrazu, uprostřed se nachází výsledek námi navržené metody, nejvíce vpravo se nachází výsledek metody *livergan*.



Obrázek 5.8: Jeden z vyřazených 17 obrázků, u kterého bylo těžké určit, která metoda dosáhla lepšího výsledku, ani jedna z metod se totiž příliš nepřiblížila vstupnímu výřezu. Na prvním obrázku zleva je reprezentováno zbarvení části původního obrazu, uprostřed se nachází výsledek námi navržené metody, nejvíce vpravo se nachází výsledek metody *livergan*.

6. Diskuze

Řešený problém spadá do kategorie úloh, ke kterým nejsou k dispozici ground truth data. Tím pádem může být složitější ověřit funkčnost navrženého řešení. Hlavní problém se skrývá ve výběru správné metody pro ověření funkčnosti. V našem případě jsme funkčnost navržené metody ověřili na základě subjektivního hodnocení odborníků. Je patrné, že obrázky vygenerované pomocí *liverganu* lépe vystihují scaffold reálně obarvený pomocí H&E barvení.

Na druhou stranu, z našeho pohledu, výsledky námi vytvořené metody lépe a přesněji zobrazují chtěné struktury. Z 336 obrazů dosáhla naše metoda lepšího výsledku v přibližně 67 % případů. U metody s využitím *liverganu* se v některých případech rovněž může stát, že neuronová síť halucinuje (vymyslí) strukturu a tudíž minimálně zčásti neodpovídá skutečné struktuře. Je však třeba zmínit, že *livergan* byl natrénován na dlaždicích o velikosti 1000×1000 pixelů. Naopak náš výstupní obrázek vznikl z postupně zpracovávaných dlaždic o velikosti 150×150 pixelů, společně s nastaveným překryvem o velikosti 15 pixelů. Je však možné pracovat i s dlaždicemi o velikosti např. 500×500. Díky tomu dojde ke znatelnému zvýšení rychlosti zpracování vstupního obrazu, avšak dosažený výsledek je kvalitativně o něco horší v porovnání se zpracováním menších dlaždic.

Mezi hlavní výhody námi vytvořeného přístupu, oproti *liverganu*, patří zachování mikrostruktury v jaterních lalůčkách. Navíc k natrénování jednotlivých modelů nebylo potřeba obrovské množství dat. Lze vyzdvihnout i dobrou vysvětlitelnost naší metody, jelikož rozumíme, co dělají jednotlivé kroky. Každý krok jsme navíc schopni ladit zvlášť, nezávisle na sobě. V případě námi navržené metody může docházet k nepřesnostem ve struktuře především v místech s vysokou koncentrací buněčných jader. V těchto místech se může stát, že některá jádra nejsou správně segmentována a tedy následně odstraněna. Dále se může stát, že i přesto, že jsou jádra správně segmentována může v další části dojít, kvůli použití biharmonické funkce k jejich filtraci, k rozmazání struktur, což může vést k znepřesnění konečného výsledku. Obecně vzato, lze však říci, že dosažené výsledky naší metody jsou velice dobré.

V dalším vývoji bychom se chtěli pokusit nalézt či sami implementovat nějakou lepší metodu pro filtraci buněčných jader, která by lépe fungovala v případě vysoké koncentrace buněčných jader ve zpracovávané dlaždici. Dále bychom se případně zaměřili na vytvoření uživatelského rozhraní pro jednodušší použití navržené metody; tak aby mohli navrženou metodu pohodlně používat např. histologové.

Závěr

V předkládané diplomové práci jsme se v první části nejprve zaměřili na studium metod přípravy a hodnocení mikroskopických histologických snímků. Dále jsme se seznámili s metodami počítačového vidění. Důraz jsme kladli především na metody spojenými s detekcí, segmentací a filtrací. Nastudovali jsme i některé postupy, které byly zaměřené přímo na zpracování medicínských obrazů.

Ve druhé části práce jsme se soustředili na vytvoření metody pro odstranění mezibuněčné hmoty z mikroskopických histologických obrazů. V prvním kroku námi vytvořená metoda nejprve provede segmentaci buněčných jader. K tomuto účelu je použita buď neuronová síť Mask R-CNN (instance segmentation), anebo U-Net (semantic segmentation). Pro vyhodnocení segmentace v případě U-Netu i Mask R-CNN byla zvolena metrika IoU, přičemž v případě U-Netu se podařilo dosáhnout hodnoty na validačních datech kolem 0.6. Konečný výstup segmentace buněčných jader je reprezentován pomocí binární masky. Binární maska je následně použita v procesu inpaintingu/filtrace, tak abychom co nejlépe „zamaskovali“ buněčná jádra v původním obrazu. Ke splnění tohoto úkolu jsme použili biharmonickou funkci. V dalším kroku jsme se mohli začít zabývat samotným odstraněním mezibuněčné hmoty. Pro tuto část jsme opět zvolili segmentační síť U-Net. Na rozdíl od trénování modelu pro segmentaci buněčných jader, kde jsme použili binární černobílé masky, jsme v tomto případě použili barevné RGB masky a tím pádem i výstup segmentace je tvořen barevnou RGB maskou. V tomto případě se podařilo na validačních datech dosáhnout hodnoty IoU téměř 0.7.

Důležitou součástí této práce je i vytvoření knihovny *wsitools*, která využívá některé moduly z knihovny *scaffan*. Knihovna *wsitools* byla vytvořena v programovacím jazyce Python. Pomocí knihovny *wsitools* lze zpracovávat paměťově náročný mikroskopický histologický obraz po jednotlivých dlaždicích. Uživatel má možnost si definovat velikost dlaždice v pixelech, společně např. s dalšími parametry jako velikost překryvu dlaždice v pixelech či velikost jednoho pixelu v milimetrech.

V poslední části jsme se, po úspěšném vyřešení úlohy, nejprve zaměřili na porovnání námi vytvořené metody s podobnou metodou (*livergan*), která je založena na použití GANů. Dále jsme ověřili funkčnost námi navrženého řešení, a to pomocí subjektivního hodnocení odborníků. Tento typ vyhodnocení jsme zvolili z důvodu neexistence ground truth dat pro naši úlohu. Odborníci dostali k dispozici 470 trojic obrazů. Jedna trojice obrazů byla tvořena pomocí skutečného obrazu, další 2 obrazy byly tvořeny výstupy námi vytvořené metody a metody *livergan*. Úkolem odborníků bylo určit, který výstupní obraz více odpovídá skutečnému vstupnímu obrazu. Pro účely vyhodnocení došlo k odstranění celkem 134 trojic (117 z nich obsahovalo pouze pozadí a u 17 z nich nešlo jasně určit, jaká metoda je lepší, jelikož se ani jedna z metod příliš nepřiblížila ke vstupnímu obrazu).

Celkem tedy vyhodnocení proběhlo na 336 trojicích obrazů. V porovnání s *liverganem* vyšlo, že ve zhruba 67 % případů (226 trojic) dosáhla námi navržená metoda lepšího věrohodnějšího výsledku. Dále z porovnání vyplynulo, že ač na některých obrázcích dosáhl *livergan* přirozenějšího vyznění, neřešil úlohu zcela dokonale, a to zejména z toho důvodu, že není vysvětlitelný a občas halucinuje. Naše metoda oproti *liverganu* dosahuje konzistentních výsledků. Například, u obrázků s určitou plochou prázdné oblasti nedochází k halucinování, a tím pádem *livergan* překonává. Lze však konstatovat, že obě metody fungují správně, a to i z důvodu, že jsme z původního počtu 470 trojic obrazů vyřadili, pro účely hodnocení, pouze 17 trojic, u kterých nebylo možné zjistit, která z metod byla úspěšnější. Každá z obou zmíněných metod má své určité výhody a nevýhody.

Reference

[1] Jan Jelínek and Vladimír Zicháček. *Biologie pro gymnázia*. Nakladatelství Olomouc, Olomouc, 11. vyd edition, 2014.

[2] Zdeněk Wilhelm and Peter Hegyi. Fyziologie jater. *Farmacie pro praxi*, 3(5):242–245, 2007.

[3] Jiří Ehrmann, Květoslava Aiglová, Ondřej Urban, Silvia Cveková, and Pavol Dvoran. Onemocnění jater související s alkoholem (ald). *Vnitřní lékařství*, 66(5):e3–e15, 2020.

[4] Hyuna Sung, Jacques Ferlay, Rebecca L Siegel, Mathieu Laversanne, Isabelle Soerjomataram, Ahmedin Jemal, and Freddie Bray. Global cancer statistics 2020: Globocan estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA: a cancer journal for clinicians*, 71(3):209–249, 2021.

[5] Harriet Rumgay, Melina Arnold, Jacques Ferlay, Olufunmilayo Lesi, Citadel J. Cabasag, Jérôme Vignat, Mathieu Laversanne, Katherine A. McGlynn, and Isabelle Soerjomataram. Global burden of primary liver cancer in 2020 and predictions to 2040. *Journal of Hepatology*, 77(6):1598–1606, 2022.

[6] Jindřich Fínek et al. Onkologická léčba nádorů jater. *Onkologie*, 2(4):223–224, 2008.

[7] David Anwanwan, Santosh Kumar Singh, Shriti Singh, Varma Saikam, and Rajesh Singh. Challenges in liver cancer and possible treatment approaches. *Biochimica et Biophysica Acta (BBA) - Reviews on Cancer*, 1873(1):188314, 2020.

[8] Lada Eberlova, Anna Maleckova, Patrik Mik, Zbynek Tonar, Miroslav Jiřík, Hynek Mirka, Richard Palek, Sarah Leupen, and Vaclav Liska. Porcine liver anatomy applied to biomedicine. *Journal of Surgical Research*, 250:70–79, 2020.

[9] Andriy Nykonenko, Petr Vávra, and Pavel Zonča. Anatomic peculiarities of pig and human liver. *Exp Clin Transplant*, 15(1):21–26, February 2017.

[10] Vladimíra Moulisová, Miroslav Jiřík, Claudia Schindler, Lenka Červenková, Richard Pálek, Jáchym Rosendorf, Janine Arlt, Lukáš Bolek, Simona Šůsová, Sandor Nietzsche, Václav Liška, and Uta Dahmen. Novel morphological multi-scale evaluation system for quality assessment of decellularized liver scaffolds. *Journal of Tissue Engineering*, 11:2041731420921121, 2020. PMID: 32523667.

[11] V Liška, V Moulisová, R Pálek, J Rosendorf, L Červenková, L Bolek, and V Třeška. Cesta k novým játrům: Decelularizace prasečích jater a jejich znovuosídlení buňkami. *Rozhledy v chirurgii*, 98(10):388–393, 2019.

[12] Michael H Ross and Wojciech Pawlina. *Histology*. Lippincott Williams & Wilkins, 2006.

[13] Jaroslav Slípka and Zbyněk Tonar. *Základy histologie*. Univerzita Karlova, nakladatelství Karolinum, Praha, 2. edition, 2018.

[14] Petr Ferczadi. *Učební text k předmětu histologická technika*. SZŠ Plzeň, 2019.

[15] Štefan Polák, Ivan Varga, et al. *Úvod do histológie a histologickej techniky*. Univerzita Komenského Bratislava, 2010.

[16] I. Vrtková. Genetic admixture analysis in prestice black-pied pigs. *Archives Animal Breeding*, 58(1):115–121, 2015.

[17] Marc Key. Immunohistochemistry staining methods. *Education Guide Immunohistochemical Staining Methods Fourth Edition*, page 47, 2006.

[18] Milena Beranová and Zbyněk Tonar. Principy a příklady imunohistochemie. *Ústav histologie a embryologie LF UK v Plzni*, page 7, 2002.

[19] George GoMori. A rapid one-step trichrome stain. *American Journal of Clinical Pathology*, 20(7_ts):661–664, 07 1950.

[20] Tonar Zbyněk. Atlas kvantitativní histologie. *Plzeň: Lékařská fakulta UK*, 2008.

[21] Zeiss. Zeiss zen microscopy software. https://www.zeiss.com/microscopy/en/products/software/zeiss-zen.html. [Online; accessed March 25, 2024].

[22] Tony J. Collins. Imagej for microscopy. *BioTechniques*, 43(1S):S25–S30, 2007. PMID: 17936939.

[23] Shijie Hao, Yuan Zhou, and Yanrong Guo. A brief survey on semantic segmentation with deep learning. *Neurocomputing*, 406:302–321, 2020.

[24] Abdul Mueed Hafiz and Ghulam Mohiuddin Bhat. A survey on instance segmentation: state of the art. *International Journal of Multimedia Information Retrieval*, 9(3):171–189, Sep 2020.

[25] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollar. Panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[26] Zhen Qin, Qingliang Zeng, Yixin Zong, and Fan Xu. Image inpainting based on deep learning: A review. *Displays*, 69:102028, 2021.

[27] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning*. Cambridge University Press, 2023. https://D2L.ai.

[28] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[29] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, Sep 2013.

[30] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[31] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[33] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 213–229, Cham, 2020. Springer International Publishing.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[35] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[36] Fahad Shamshad, Salman Khan, Syed Waqas Zamir, Muhammad Haris Khan, Munawar Hayat, Fahad Shahbaz Khan, and Huazhu Fu. Transformers in medical imaging: A survey. *Medical Image Analysis*, 88:102802, 2023.

[37] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision*. Cengage Learning, 2014.

[38] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[39] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[40] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[41] Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[42] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[43] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[44] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[45] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017.

[46] Sihang Zhou, Dong Nie, Ehsan Adeli, Jianping Yin, Jun Lian, and Dinggang Shen. High-resolution encoder–decoder networks for low-contrast medical image segmentation. *IEEE Transactions on Image Processing*, 29:461–475, 2020.

[47] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes,

N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[48] Xian Wu, Kun Xu, and Peter Hall. A survey of image synthesis and editing with generative adversarial networks. *Tsinghua Science and Technology*, 22(6):660–674, 2017.

[49] Michael Majurski, Petru Manescu, Sarala Padi, Nicholas Schaub, Nathan Hotaling, Carl Simon Jr, and Peter Bajcsy. Cell image segmentation using generative adversarial networks, transfer learning, and augmentations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[50] Nasim Souly, Concetto Spampinato, and Mubarak Shah. Semi supervised semantic segmentation using generative adversarial network. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[51] A. Criminisi, P. Perez, and K. Toyama. Object removal by exemplar-based inpainting. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–II, 2003.

[52] S. B. Damelin and N. S. Hoang. On surface completion and image inpainting by biharmonic functions: Numerical aspects. *International Journal of Mathematics and Mathematical Sciences*, 2018:1–8, 2018.

[53] scikit image. https://scikit-image.org/docs/stable/auto_examples/filters/plot_inpaint.html. [Online; accessed May 15, 2024].

[54] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Generative image inpainting with contextual attention. *CoRR*, abs/1801.07892, 2018.

[55] Karim Armanious, Youssef Mecky, Sergios Gatidis, and Bin Yang. Adversarial inpainting of medical image modalities. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3267–3271, 2019.

[56] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[57] Karim Armanious, Chenming Jiang, Marc Fischer, Thomas Küstner, Tobias Hepp, Konstantin Nikolaou, Sergios Gatidis, and Bin Yang. Medgan: Medical image translation using gans. *Computerized Medical Imaging and Graphics*, 79:101684, 2020.

[58] Qianna Wang, Yi Chen, Nan Zhang, and Yanhui Gu. Medical image inpainting with edge and structure priors. *Measurement*, 185:110027, 2021.

[59] Li Xu, Qiong Yan, Yang Xia, and Jiaya Jia. Structure extraction from texture via relative total variation. *ACM transactions on graphics (TOG)*, 31(6):1–10, 2012.

[60] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[61] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022. [Online; accessed March 22, 2024].

[62] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[63] Patrick von Platen, Suraj Patil, and Anton Lozhkov. In-painting pipeline for stable diffusion using diffusers. https://colab.research.google.com/github/huggingface/notebooks/blob/main/diffusers/in_painting_with_stable_diffusion_using_diffusers.ipynb, Sep 2022. [Online; accessed March 22, 2024].

[64] Piotr Skalski. Make Sense. https://github.com/SkalskiP/make-sense/, 2019. [Online; accessed April 23, 2024].

[65] COCO data format. https://cocodataset.org/#format-data. [Online; accessed March 21, 2024].

[66] Jan Burian. Coco_format. https://github.com/janburian/COCO_format/, Dec 2022. [Online; accessed March 22, 2024].

[67] Miroslav Jiřík. ScaffAn. https://github.com/mjirik/scaffan, Oct 2018. [Online; accessed March 22, 2024].

[68] Jan Burian and Miroslav Jiřík. wsitools. https://github.com/mjirik/wsitools, Feb 2024. [Online; accessed March 22, 2024].

[69] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. [Online; accessed March 22, 2024].

[70] Soumyajit Podder, Somnath Bhattacharjee, and Arijit Roy. An efficient method of detection of covid-19 using mask r-cnn on chest x-ray images. *AIMS Biophysics*, 8:281–290, 07 2021.

[71] facebookresearch. Detectron2 Model Zoo and Baselines. https://github.com/facebookresearch/detectron2/blob/main/MODEL_ZOO.md, July 2021. [Online; accessed: May 10, 2024].

[72] Sreenivas Bhattiprolu. python_for_microscopists. https://github.com/bnsreenu/python_for_microscopists/blob/master/207-simple_unet_model_with_jacard.py, 2023. [Online; accessed April 23, 2024].

[73] Paul Gavrikov. visualkeras. https://github.com/paulgavrikov/visualkeras, 2020. [Online; accessed April 30, 2024].

[74] Noor A Ibraheem, Mokhtar M Hasan, Rafiqul Z Khan, and Pramod K Mishra. Understanding color models: a review. *ARPN Journal of science and technology*, 2(3):265–275, 2012.

[75] Václav Javorek. Image Domain Transfer for Liver Analysis in Histology. Diplomová práce, Západočeská univerzita v Plzni, Plzeň, 2024.

[76] Václav Javorek. livergan. https://github.com/VaJavorek/livergan, Feb 2024. [Online; accessed May 9, 2024].

Seznam obrázků

Seznam tabulek

A. Přílohy

A.1 První příloha

Příslušné zdrojové kódy, jupyter notebooky, datasety a natrénované modely jsou dostupné prostřednictvím veřejného GitHubu repozitáře.

A.2 Druhá příloha

Listing A.1: Implementace metody process_tile()

```
def process_tile(tile: np.array) -> np.array:
   """
   Process a tile by drawing a red square on it.

   Parameters:
   - tile (np.array): Input tile image (assumed to be in RGB format).

   Returns:
   - np.array: Processed tile with a red square drawn on it.
   """
   if tile.shape[2] != 3:
      raise ValueError("Image ndarray must have 3 channels for RGB.")

   # Create a copy of the image to avoid modifying the original array
   result_tile = np.copy(tile)

   # Set the red color (assuming RGB format)
   red_color = [255, 0, 0]

   # Draw the red square
   result_tile[50:50 + 50, 50:50 + 50, :] = red_color

   return result_tile
```

A.3 Třetí příloha

Tabulka A.1: Architektura zvoleného U-Net modelu.

	Name	Type	Shape
0	input_3	InputLayer	[(None, 256, 256, 3)]
1	conv2d_38	Conv2D	(None, 256, 256, 16)
2	dropout_18	Dropout	(None, 256, 256, 16)
3	conv2d_39	Conv2D	(None, 256, 256, 16)
4	max_pooling2d_8	MaxPooling2D	(None, 128, 128, 16)
5	conv2d_40	Conv2D	(None, 128, 128, 32)
6	dropout_19	Dropout	(None, 128, 128, 32)
7	conv2d_41	Conv2D	(None, 128, 128, 32)
8	max_pooling2d_9	MaxPooling2D	(None, 64, 64, 32)
9	conv2d_42	Conv2D	(None, 64, 64, 64)
10	dropout_20	Dropout	(None, 64, 64, 64)
11	conv2d_43	Conv2D	(None, 64, 64, 64)
12	max_pooling2d_10	MaxPooling2D	(None, 32, 32, 64)
13	conv2d_44	Conv2D	(None, 32, 32, 128)
14	dropout_21	Dropout	(None, 32, 32, 128)
15	conv2d_45	Conv2D	(None, 32, 32, 128)
16	max_pooling2d_11	MaxPooling2D	(None, 16, 16, 128)
17	conv2d_46	Conv2D	(None, 16, 16, 256)
18	dropout_22	Dropout	(None, 16, 16, 256)
19	conv2d_47	Conv2D	(None, 16, 16, 256)
20	conv2d_transpose_8	Conv2DTranspose	(None, 32, 32, 128)
21	concatenate_8	Concatenate	(None, 32, 32, 256)
22	conv2d_48	Conv2D	(None, 32, 32, 128)
23	dropout_23	Dropout	(None, 32, 32, 128)
24	conv2d_49	Conv2D	(None, 32, 32, 128)
25	conv2d_transpose_9	Conv2DTranspose	(None, 64, 64, 64)
26	concatenate_9	Concatenate	(None, 64, 64, 128)
27	conv2d_50	Conv2D	(None, 64, 64, 64)
28	dropout_24	Dropout	(None, 64, 64, 64)
29	conv2d_51	Conv2D	(None, 64, 64, 64)
30	conv2d_transpose_10	Conv2DTranspose	(None, 128, 128, 32)
31	concatenate_10	Concatenate	(None, 128, 128, 64)
32	conv2d_52	Conv2D	(None, 128, 128, 32)
33	dropout_25	Dropout	(None, 128, 128, 32)
34	conv2d_53	Conv2D	(None, 128, 128, 32)
35	conv2d_transpose_11	Conv2DTranspose	(None, 256, 256, 16)
36	concatenate_11	Concatenate	(None, 256, 256, 32)
37	conv2d_54	Conv2D	(None, 256, 256, 16)
38	dropout_26	Dropout	(None, 256, 256, 16)
39	conv2d_55	Conv2D	(None, 256, 256, 16)
40	conv2d_56	Conv2D	(None, 256, 256, 1)