P R O H L Á Š E N Í

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

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

V Plzni dne …….. ……

Bohdan Yeremenko

Západočeská univerzita v Plzni
Fakulta aplikovaných věd

Katedra kybernetiky
Program: Aplikované vědy a informatika

BAKALÁŘSKÁ PRÁCE:

Využití hlubokých neuronových sítí pro segmentaci parenchymatózních orgánů břišní dutiny prasete domácího

Use of deep neural networks for segmentation of parenchymatous organs of the abdominal cavity of domestic pigs

Vypracoval: Bohdan Yeremenko
Vedoucí práce: Ing. Miroslav Jiřík, Ph.D.
Rok: 2023

Jméno a příjmení autora:	Bohdan Yeremenko
Název bakalářské práce:	Využití hlubokých neuronových sítí pro segmentaci parenchymatózních orgánů břišní dutiny prasete domácího
Název v angličtině:	Use of deep neural networks for segmentation of parenchymatous organs of the abdominal cavity of domestic pigs.
Studijní program:	Aplikované vědy a informatika
Studijní obor:	Kybernetika a řídicí technika
Vedoucí bakalářské práce:	Ing. Miroslav Jiřík, Ph.D.
Rok obhajoby:	2023

Anotace v češtině

Tato bakalářská práce se zabývá možností aplikace hlubokých neuronových sítí v medicinských úlohách, konkrétně úlohou segmentace parenchymatózních orgánů břišní dutiny prasete domácího. Pro realizaci jsme využili framework detectron2 založenou na architektuře Mask R-CNN. V práci jsou popsány technologie používané pro strojové zpracování obrazu. Důležitou částí práce je popis konvolučních neuronových sítí určených pro zpracování obrazu. Hlavní část práce se věnuje popisu Mask R-CNN a detectron2, a to prostřednictvím analýzy výsledku trénování této sítě na medicínských datech. Na závěr je uvedeno rozhodnutí, je-li síť vhodná pro využití v praxi.

Anotace v angličtině

This work deals with the possibility of applying deep neural networks in medical tasks, especially in the segmentation of parenchymatous organs of the abdominal cavity of the domestic pig. For this, we used “detectron2”, which is based on “Mask R-CNN”. In this work we described technology, which is used for digital image processing. Important part of this work is description of Convolutional Neural Networks and their usage for working with digital images. Main part describes „Mask R-CNN“, „detectron2“ and analyzes results of our training on medical dataset. In conclusion we gave a decision about the network (whether it is useful for practice).

Poděkování:

Rád bych věnoval poděkování doc. Ing. Miroslavu Jiříkovi, Ph.D., za podporu při psaní této bakalářské práce a za jeho cenné rady.

Obsah

Úvod

Rakovina je dnes jedním z nejdůležitějších témat v medicíně [17]. Kvůli zvýšení počtu případů onemocnění ve vyspělých zemích lékaři získávají více financí na výzkum v této oblasti. Rovněž vysoká míra rizika nedovoluje použít klinicky neověřené metody léčby pacientů. Při testování na zvířatech je však nutné dodržovat řadu etických pravidel., proto se experimenty neprovádějí na lidech, ale na zvířatech. Dnes se takové experimenty provádějí i ve Fakultní nemocnici v Plzni. Hlavní motivací mé bakalářské práce a nejdůležitějším směrem výzkumu je podpora takovýchto experimentů, ulehčení práce lékařům a zlepšení kvality dat získávaných z experimentů. V našem případě jsou testovacími subjekty prasata domácí a jejich parenchymatózní orgány. Pro získávání potřebných výsledků musí lékaři sledovat stav jednotlivých orgánů i v době mezi operacemi, a proto využívají mimo jiné technologii výpočetní tomografie, zkráceně CT (z anglického **Computed Tomography**). Pro zjednodušení budu v dalších částech kvalifikační práce používat zkratky. Konečným cílem práce je automatizace procesů segmentace jednotlivých parenchymatózních orgánů prasete domácího z CT snímků. Program bude předzpracovávat data (CT snímky), která do něj přicházejí, a provádět manipulace, jejichž výsledkem bude oddělení jednotlivého orgánu pro další pozorování lékaři. Pro lepší pochopení bude další část bakalářské práce obsahovat jednotlivé sekce popisující různá témata důležitá pro práci. Jedním z takových témat je medicína. Porozumění medicíně je zásadním kritériem pro vývoj programu v této oblasti. Bez základních znalostí v oblasti anatomie a principu práce tomografu nelze dokonale chápat konečný cíl tématu. Práce s vizuálními obrazy je samozřejmě problematikou oblasti „počítačového vidění", jehož metody budou v práci popsány. Zajímá nás především použití neuronových sítí v problematice zpracování vizuálních obrazů. Svou kapitolu také potřebuje námi využitá konkrétní implementace segmentačního procesu, u které předpokládáme využití klasické klasifikační technologie nebo technologie učení neuronové sítě, přičemž obě dvě metody budou vyžadovat informace od učitelů. Pro dosažení našeho cíle byla také velmi důležitá experimentální část práce, jejímuž popisu bude v práci věnován dostatečný prostor.

1 Medicínská teorie

1.1 Výpočetní tomografie

Historie vzniku CT je spojena s velkým rozvojem digitálních a výpočetních technologií během 20. století. Předpokladem pro směřování vývoje v tomto směru se stala nedokonalost klasického rentgenu. Myšlenka, která se stala základem pro danou technologii, byla kompilace velkého počtu navzájem odlišných snímků pořízených pod různými úhly a stanovení požadované informace získané matematickým zpracováním denzity zkoumané látky v řadě sekcí. Rozvoj výpočetní techniky umožnil sestavit 3D model požadovaných objektů pomocí velkého počtu řezů. Prvním, kdo přišel s myšlenkou využití takové technologie v medicíně, byl americký neurorentgenolog William Oldendorf v roce 1961. Největšího úspěchu v této oblasti však dosáhli jihoafricko-americký fyzik Allan

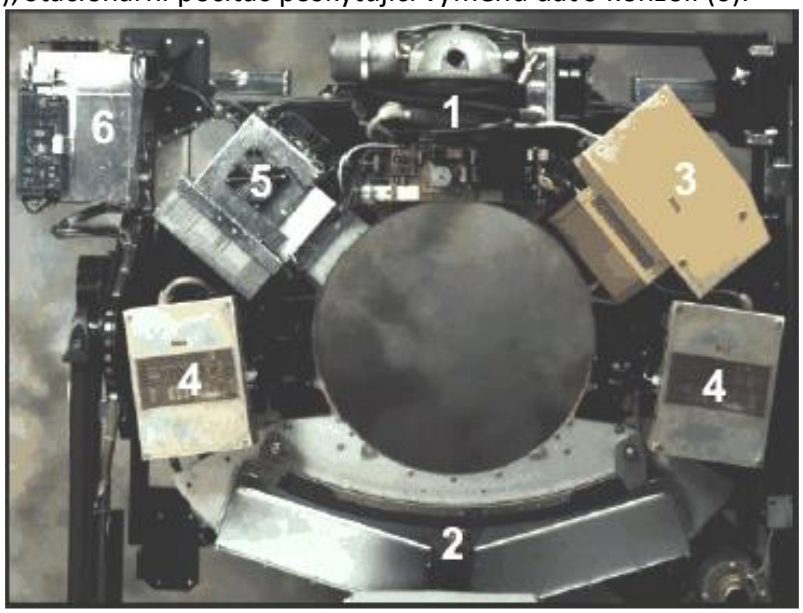
McLeod Cormack a anglický inženýr Godfrey Newbold Hounsfield, kteří v roce 1963 jako první ukázali možnost rekonstrukce vizuálního zobrazení pomocí laboratorních experimentů. První kvalitní tomogram mozku člověka pak získali v roce 1972, ten však ještě nebyl využit v praxi. Komercializovat metodu se podařilo G. N. Hounsfieldovi. V rámci spolupráce s anglickou firmou EMI byl zkonstruován skener mozku. Pomocí něj mohli lékaři získat zobrazení 80 x 80 pixelů, což odpovídalo 3 mm na 1 pixel. Sběr dat trval 4,5 minuty a jejich zpracování a rekonstrukce do obrazu pomocí počítače trvaly 2,5 hodiny. Kvůli tak pomalé práci se první tomografy využívaly jen pro výzkum mozku. Přestože cena tomografu činila přibližně 390 000 dolarů, přístroj se stal populárním a v roce 1979 už byl používán ve více než 2000 nemocnicích světa. V tomto roce A. Cormak a G. Hounsfield získali za vývoj technologie CT Nobelovu cenu za fyziologii a lékařství. O tři roky později, v roce 1982, získal Nobelovu cenu za chemii Aaron Klug, který se zabýval vývojem experimentálních a výpočetních metod v trojrozměrném CT. V historii vývoje CT se klasifikuje pět generací CT skenerů. Tomograf měl jednu rentgenku a jeden snímač, které se synchronně pohybovaly podél rámce. Měření se provádělo ve 180 polohách rentgenky, poté se rám otáčel o 1° a měření se opakovala. Veškerý proces trval stejnou dobou jako u skeneru EMI. Tomografy druhé generace už měly několik snímačů fungujících paralelně a rentgenka namísto přímého paprsku generovala vícesměrové záření. Díky tomu se šířka skenované plochy zvětšila na 30° a proces skenování se zrychlil na 20 vteřin. Třetí generace vznikla v polovině sedmdesátých let. Tyto přístroje stejně jako přístroje druhé generace využívají technologii záření, ale ještě se zvětšila šířka záběru. Snímače, kterých už bylo kolem 700, začaly měnit své umístění z paralelního na obloukovité. Tato změna v konstrukci tomografu umožnila provádět snímání při kontinuálním otáčení rentgenky a snímače o 360° po směru hodinových ručiček pomocí kluzného kroužku. Díky tomu se proces zrychlil na 10 vteřin a od této doby je tomograf již možné využít pro analýzu pohyblivých částí těla, jako jsou plíce a dutina břišní. Také díky tomu se zlepšily i teoretické koncepce a tomografy přešly na spirální algoritmus sběru dat. V dnešní době všechny medicinské tomografy náleží třetí generaci. Tomografy čtvrté generace mají monolitní kolo detektorů a rentgenka jezdí uvnitř kola, což umožnilo zrychlit tomograf a zlepšit kvalitu snímků ještě víc. Ale u daného typu tomografu je třeba počítat s efektem rozptylu při přenosu záření, který v závislosti na typu záření může byt Rayleighůvym nebo Comptonovým rozptylem [1].

Tomografy páté generace se začaly vyrábět počátkem 80. let a klasifikujeme je ještě jako „Electron beam computed tomograph“. Jejich princip je takový, že tok elektronů se generuje pomocí nehybného generátoru paprsku elektronů. Po průchodu vakuem se tok elektronů následně fokusuje a pomocí elektromagnetického vinutí je směrován na wolframový terč ve formě kruhového oblouku přibližně 200°, který se nachází pod stolem pacienta. Pro práci s vizuálními daty z výpočetního tomografu je tato informace o principu fungování tohoto zařízení je dostačující, pro lepší pochopení fyzické báze dnešní tomografie ji však dále probereme detailněji. Všechny dnešní výpočetní tomografy obsahují čtyři hlavní komponenty:

1) „Gantry”. Jinak řečeno sestava válcového skeneru používaná pro lékařské 3D zobrazování nebo léčbu.
2) Generátor vysokého napětí.
3) Výpočetní systém.

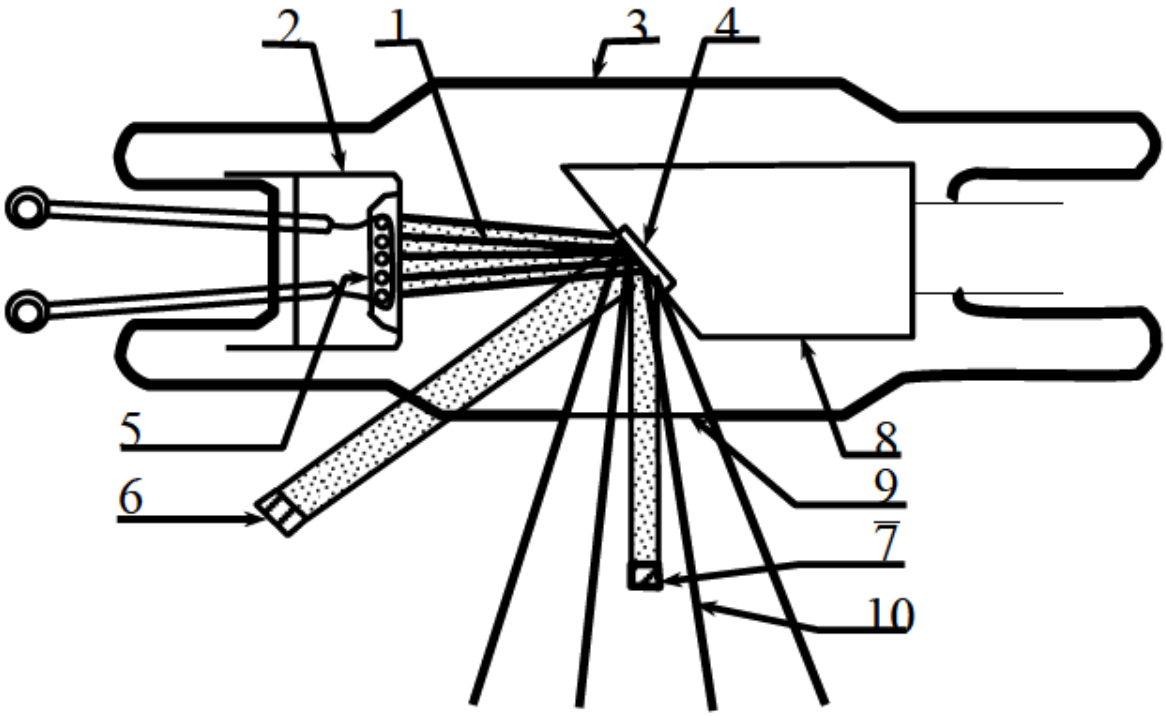
4) Konzole operátora/uživatele.

Uvnitř „Gantry" se nacházejí bloky odpovídající za sběr dat (obrázek 1.1): rentgenka a kolimátory (1), snímače a systém sběru dat (2), řídicí jednotka rotace rentgenky (3), generátor vysokofrekvenčních signálů (4), vestavěný mikropočítač řídící napětí a proud v rentgence (5), stacionární počítač poskytující výměnu dat s konzolí (6).



Obrázek 1.1: Sestava válcového skeneru používaná pro lékařské 3D zobrazování nebo léčbu [2].

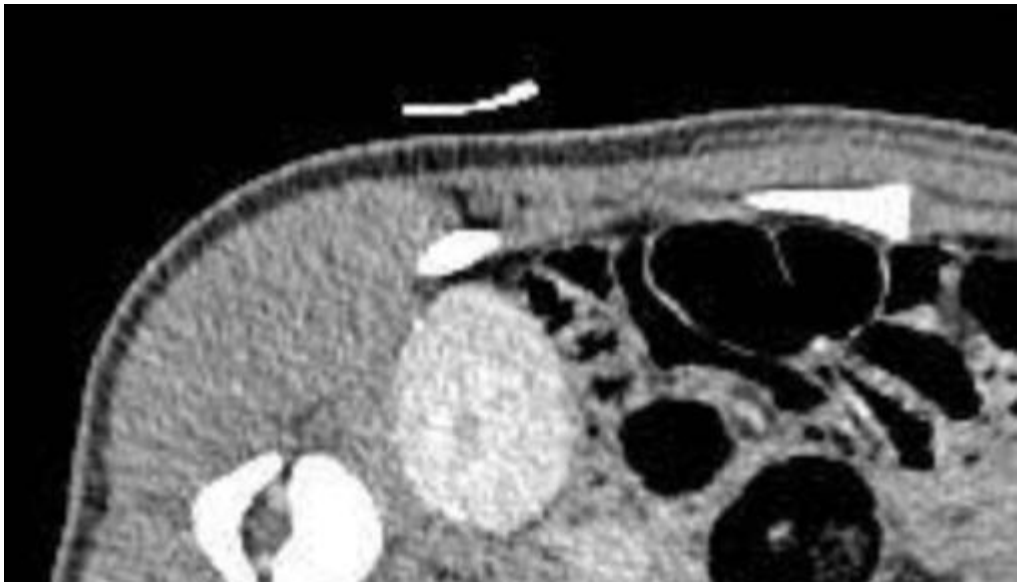
Rentgenové záření je generováno pomocí rentgenky (obrázek 1.2). Zdrojem elektronů (katody) je v našem případě wolframové vlákno rozehřáté elektrickým proudem. Poté se elektrony fokusují na anody. Kvůli velkému snížení rychlosti elektronů při dopadu na anody vznikají elektromagnetické vlny o délce od 10^{-14} do 10^{-7}, které řadíme do rentgenového záření. V nedávné minulosti měly rentgenky CT systémů výkon 80–140 kW při napětí 80–140 kV. Při využití systému v režimu maximálního výkonu mohly být tyto tomografy kvůli riziku přehřátí těchto elementů používány pouze po omezený čas. Takové limity závisí na vlastnostech anody a generátoru. Dnešní technologie využití několika řad detektorů a efektivního využití možností rentgenky nám však umožňují takové omezení téměř zanedbat. Také kvůli rozvoji medicíny se elektrický proud omezuje na rozsah od 10 mA do 440 mA, aby vliv ozáření pacienta byl minimálně potřebný pro získání vhodné kvality CT snímků. Analogový signál jde ze snímače do A/D převodníku, jehož výstup vyžaduje počítač/rekonstruktor. Všechna data zůstávají v jeho operační paměti až do konce skenování, což dává možnost rekonstruovat obraz v zadané perspektivě.



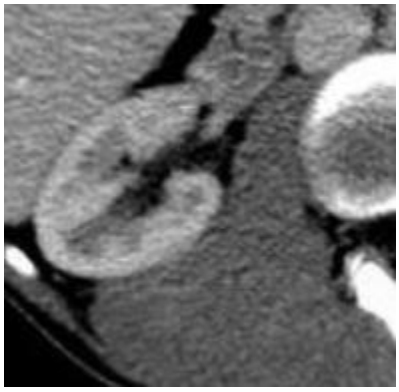
Obrázek 1.2. Schéma rentgenky: 1) paprsek elektronů; 2) katoda s fokusovanými elektrony; 3) tělo ze skla; 4) wolframový terč (antikatoda); 5) žhavicí vlákno; 6) skutečně ozářená oblast náhradní; 7) ohnisko (místo fokusu paprsku); 8) měděná anoda; 9) okno; 10) rozptýlené rentgenové záření [2].

1.2 Anatomie

Parenchym je funkční epitelová tkáň orgánů, jako jsou například játra, plíce nebo ledviny. Orgány tvořené parenchymem nazýváme též parenchymové orgány. Ve své práci jsme především sledovali segmentaci ledvin, a to kvůli jejich vhodné formě, což znamená malou deformaci během změny řezu, a také dobrý kontrast orgánu oproti jeho pozadí (obrázek 1.3). Ledviny jsou párový orgán, jehož funkcí je regulace chemické homeostázy, lépe řečeno čištění krve tvorbou moči. Nacházejí se v bederní oblasti jak u člověka (obrázek 4), tak i u prasete domácího. Lidská ledvina je fazolovitého tvaru, čemuž aproximativně odpovídá ledvina prasete domácího. Z vnější strany jsou ledviny savců pokryté denzitou vláknitou tobolkou, což se na CT snímku projevuje vyšším kontrastem objektu, kolem něhož je břišní dutina a svaly s nízkou denzitou. Fixace ledvin v těle také pomáhá při volbě klasifikátoru. Renální fascie, tuková kapsle, svalový kompartment a ledvinové pedikly bezpečně udržují ledviny v přibližně konstantní poloze. Kromě samotného fixačního mechanismu má na udržení polohy orgánu vliv nitrobřišní tlak. Když z jakéhokoliv důvodu fixační mechanismus neposkytuje vhodnou polohu orgánu, vzniká odchylka polohy ledviny směrem dolů, což nazýváme nefroptózou. Při segmentaci tomu také musíme věnovat velkou pozornost. Díky podobné lokaci, formě a dalším vlastnostem ledvin u savců můžeme při nalezení správné metody využívat znalosti o segmentaci lidských ledvin [3].



Obrázek 1.3: Ledvina prasete domacího.



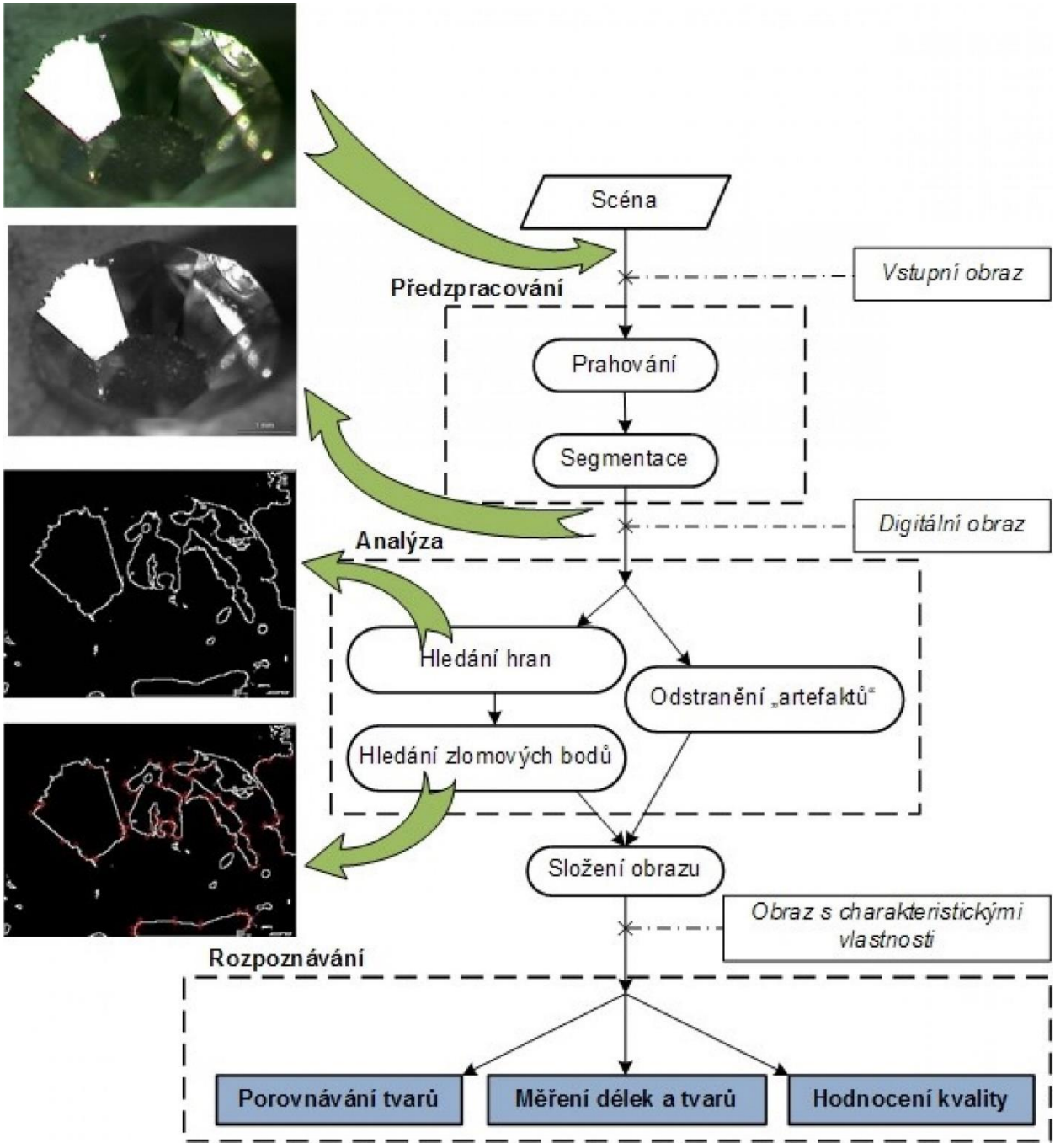
Obrázek 1.4: Ledvina člověka.

2 Strojové zpracování obrazu

2.1 Klasické zpracování obrazu

2.1.1 Zpracování obrazu pomocí počítačového vidění

Podle klasifikace považujeme zpracování obrazu za součást vědní disciplíny s názvem „Strojové vidění“ (angl. machine vision). Její kořeny však můžeme najít i v oblasti televizní techniky, kybernetiky, výpočetní techniky, optiky či fyziky obecně. Hlavní principy byly deklarovány na základě pochopení počítačové reprezentace optických vlastnosti objektů, ale také na základě vizuální reprezentace fyzikálních vlastností obrazů. Jako příklad můžeme uvést CT. To v sobě zahrnuje vizualizaci denzity orgánů, jejich formy a objemu. Kromě toho dáváme v tomto případě velký pozor na polohu segmentovaných objektů, v daném případě orgánů.



Obrázek 2.1: Základní zásady strojového zpracování obrazu [4].

Obraz, který zpracováváme, se nazývá scéna. Ta musí být nejprve předzpracována. Jak je patrné z grafu 2.1, základními operacemi prováděnými v rámci předzpracování jsou prahování a segmentace [4]. Prahování funguje podle principu převodu obrazu do binární struktury. Zavádíme konstantu T, kterou nazýváme práh (angl. threshold). Koeficient intenzity (jas) veškerých pixelů $I_{i,j}$ porovnáváme s T a podle toho provádíme jejich klasifikaci do 0 nebo 1. Prahování nám slouží k tomu, abychom si mohli ujasnit, kde je pozadí a kde popředí. Obvykle to funguje tak, že přiřadíme 0 pro pixely v pozadí a 1 pro pixely v popředí. Kromě toho, když je rozdíl mezi jasy popředí a pozadí malý, přímé použití

prahování je prakticky nemožné.

Převod na šedotónové spektrum je také často používanou metodou předzpracování obrazu. Při analýze hustoty (v radiologie denzity) segmentovaného objektu je pro počítač vhodnější popis pomocí stejnobarevného spektra v popisu analyzovaného obrazu. V případě segmentace orgánu není informace o barvě natolik důležitá, abychom ji nemohli zanedbat.

Musíme ale vzít v potaz, že libovolná manipulace s původním obrazem vede nejen k pozitivním změnám. Velmi často metody zanedbávají některé informace a při náhodném využití takových metod může dojít k převodu dat do nevyužitelného stavu [4].

Kromě digitálního zpracování zkoumaného obrazu jsou také důležitá vhodná vstupní data. V našem případě jsme to řešit nemuseli, protože jsme získali dataset od Biomedicinského centra Lékařské fakulty UK v Plzni, kde předpokládáme kvalitní snímky.

Po kroku předzpracování následuje krok analýza. Jeho druhý název je segmentace, což jeho funkce popisuje výstižněji. Protože náš stroj musí provádět analýzu vstupního signálu, v našem případě obrazu, potřebuje nějakým způsobem rozlišovat objekty v něm. Základní segmentace se provádí pomocí hledání hran objektu, tj. hledání zlomových bodů. Také během segmentace stroj obvykle musí odstranit takzvané artefakty. To znamená zanedbání malých objektů, které pro nás nenesou žádnou zajímavou informaci. Jestliže je objekt velký, ponecháváme ho, protože ho zanedbáme v příštím kroku.

Poslední krok se nazývá rozpoznávání obrazu. V tomto kroku je realizován proces „myšlení“. Pomocí znalostí o obrazech, které jsme získali v rámci předchozího kroku, provádíme finální operace na obrazu, během které se snažíme dozvědět potřebný výsledek. Obvyklé operace jsou porovnání tvaru, měření délek tvaru, hodnocení kvality [4].

2.1.2 Proč využíváme hluboké neuronové sítě

Dnes se pro zpracování obrazu začínají čím dál častěji využívat hluboké neuronové sítě (angl. Deep Neural Networks). Počátek využití takové metody pojen s praktickým využitím takového způsobu, který nevyhovuje ideální znalosti zkoumaného obrazu. Ještě v roce 2008 většina vědců věřila tomu, že systém počítačového vidění musí být ručně laděn s pochopením všech aspektů objektu, na kterém provádíme experiment. Podle většinového názoru účastníků vědecké konference z roku 2008 nebude problematika klasifikace objektu v digitálním obrazu nikdy řešena tak, že neuronové síti ukážeme příkladu obrázku s klasifikovanými objekty a neuronová síť z něj získá všechny své znalosti. Myšlenka však spočívá v tom, že ruční programování pomocí inženýra, který analyzuje obraz na bázi svých expertních znalostí v oblasti, nemůže být lépe vyhodnocena než obvyklý proces učení, který je objektivně popsatelný. Při dostatečném počtu dat pro učení jsou vykazovány lepší výsledky než v případě ručního programování, zvláště ve složitějších vstupních signálech obsahujících velký počet objektů a šumů, se kterými musíme pracovat. V současnosti vědci došli k tomu, že pro efektivní práci deep learning algoritmu založeného na principu backpropagation potřebujeme velké množství dat a výpočtů s nimi [5].

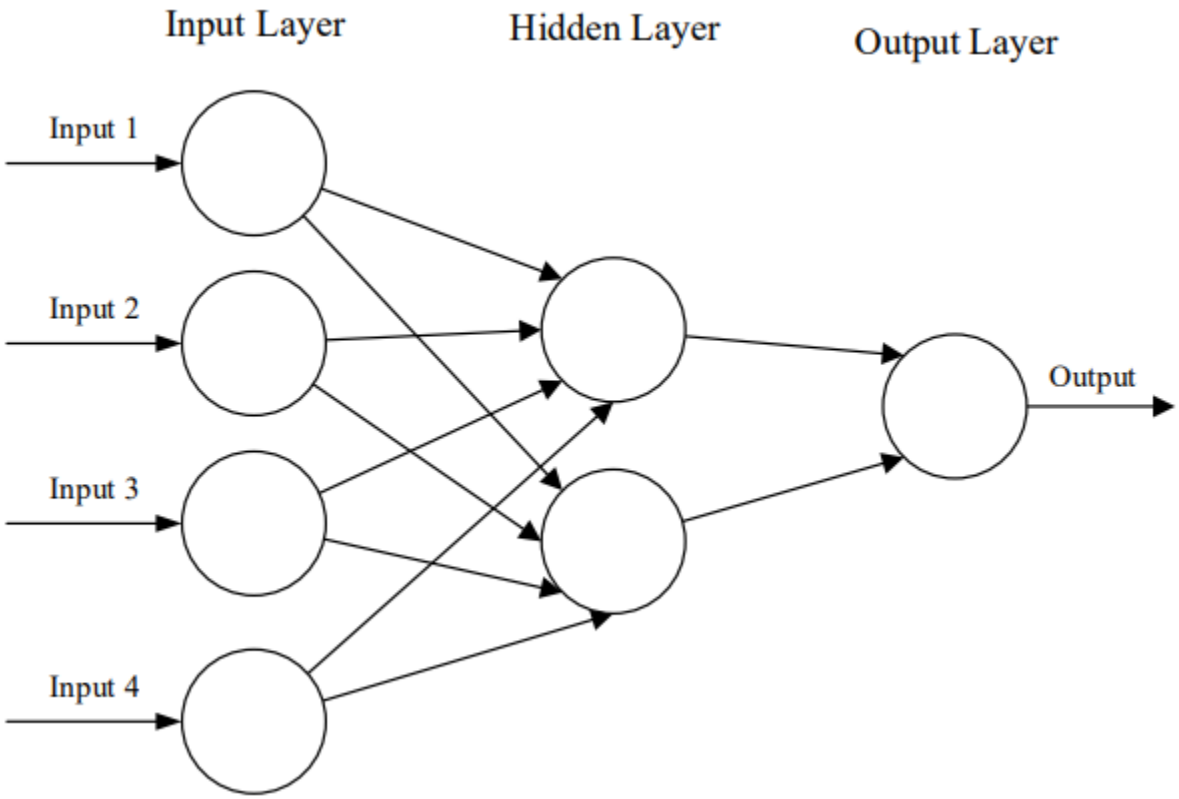
Jako teoretický příklad jsme v práci uvedli *ImageNet Classification with Deep Convolutional Neural Networks*. Tato práce je jednou z prvních, která přivedla použitelné výsledky využití hlubokých neuronových sítí a získala schválení od vědecké komunity. (Jeho

skupina dosáhla nejlepších výsledku v „ImageNet Large-Scale Visual Recognition Challenge“). Během zpracování své práce jsem velmi často směřoval k principům, které popsal Alex Krizhevsky ve své práci. Vzhledem k tomu, že zpracování medicínských dat vyžaduje pro klasické ruční ladění metod strojového zpracování obrazů hluboké znalosti v oblasti medicíny, zvolili jsme v naší práci přístup, který minimalizuje znalostní požadavky. Jinak řečeno, přestože jsme neměli možnosti pro častou komunikaci s expertem v této oblasti, zkoumali jsme možnost efektivního trénování hluboké neuronové sítě při datasetu o velikosti 20 000 obrazů [5].

2.2 Neuronové Sítě

2.2.1 Artificial Neural Networks

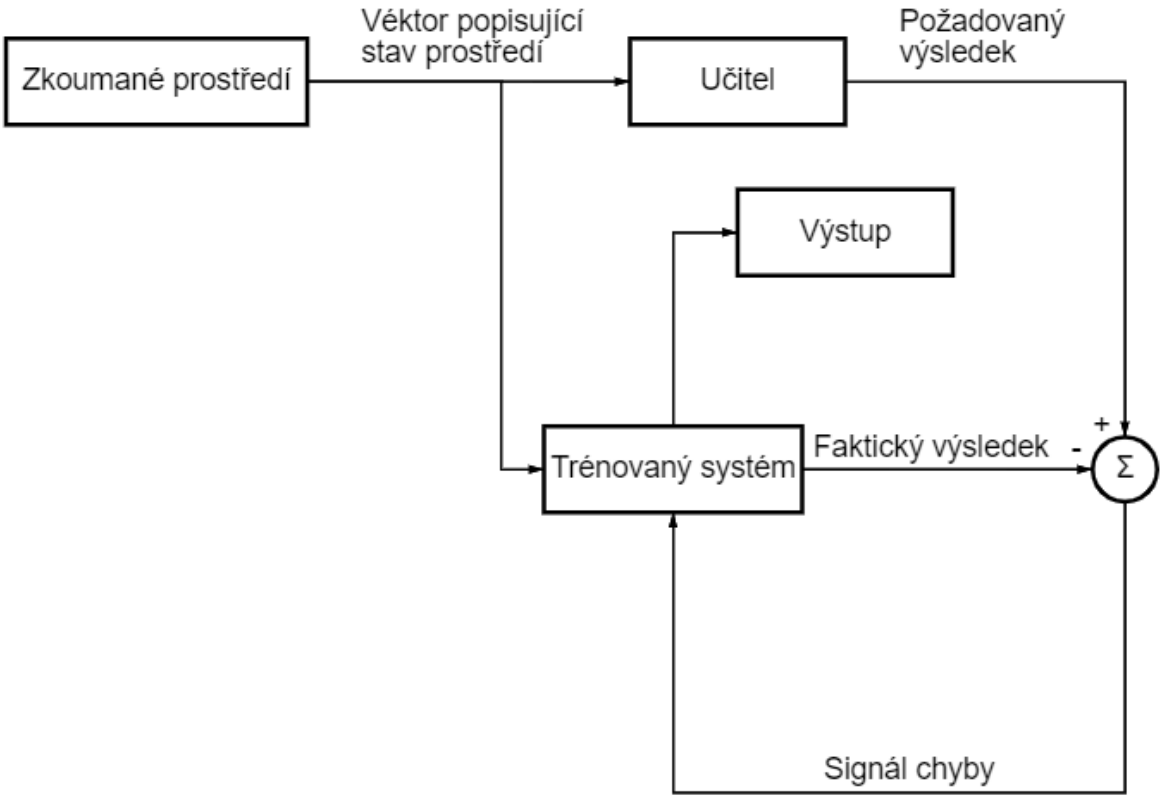
Ve své práci Alex Krizhevsky využil síť typu Convolutional Neural Networks (CNN) [5]. Pro jejich pochopení potřebujeme rozumět základní architektuře Artificial Neural Networks (ANN). ANN jsou systémy výpočetních procesů, které jsou silně inspirovány tím, jak fungují biologické nervové systémy (proto jsou označovány jako neuronové sítě). Kvůli imitaci činnosti lidského mozku jsou součástí ANN prvky, které nazýváme výpočetní uzly. Ty jsou vzájemně propojeny mezi sebou a jejich práce je provázána do jediného celku [6].



Obrázek 2.2: Základní architektura ANN [6].

Na vstup našeho ANN obvykle pošleme signál s informací, kterou reprezentuje multidimenzionální vektor. Vstupní vrstva pak přenáší informace do skryté vrstvy. Skryté vrstvy následně činí rozhodnutí z předchozí vrstvy a vyhodnocují, zda stochastická změna konečný výstup poškozuje, nebo zlepšuje. To se označuje jako proces učení. V případě nalezení modelu s více než jednou skrytou vrstvou, jež jsou mezi sebou vzájemně spojeny, budeme takovou síť označovat jako hlubokou [6].

Při využití neuronových sítí ve zpracování vizuálního obrazu existují dvě paradigmata: učení s učitelem a učení bez učitele. Učení s učitelem předpokládá trénování prostřednictvím asociací mezi obrazem zpracovaným pomocí experta a výchozím obrazem, který expert poté zpracovával. Jinak řečeno, snažíme se naučit neuronovou síť provádět mapování příznaků mezi sadou vstupních proměnných X a výstupní proměnnou Y a využít toto mapování k predikci výstupů pro nová data, která nebyla využita pro trénování. Učení s učitelem je nejdůležitější metodou strojového učení a má rovněž velký význam pro zpracování multimediálních dat [7].

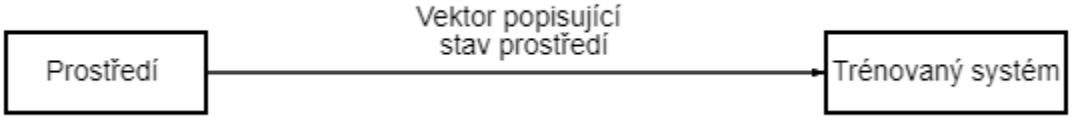


Obrázek 2.3: Diagram učení s učitelem [7].

Konceptuálně můžeme učitele chápat jako informaci o zkoumaném prostředí v podobě páru signálů vstup-výstup. Trénovaná neuronová síť přitom nemá přímý kontakt se zkoumaným prostředím, to je pro trénovanou neuronovou síť neznámé. Předpokládáme, že do neuronové sítě a učiteli se posílá trénovací vektor ze zkoumaného prostředí. Na základě vestavěných znalostí může učitel generovat požadovanou odezvu a přenášet ji do trénované neuronové sítě, odpovídající vstupnímu vektoru. Tato odezva reprezentuje

optimální výsledek, který nám má neuronová síť dávat. Parametry neuronové sítě korigujeme pomocí trénovacího vektoru a signálu chyby odezvy. Signál chyby odezvy (error signal) je rozdíl mezi požadovaným signálem a aktuální odezvou neuronové sítě. Korekce parametrů se provádí krok za krokem s cílem imitace (emulation) chování učitele. Tato emulace musí být optimální v rámci námi vybraného statistického kritéria. Takovým způsobem přenášíme znalosti od učitele do neuronové sítě v plném rozsahu. Po procesu trénování můžeme učitele odpojit od systému a dát neuronové síti možnost pracovat ve zkoumaném prostředí samostatně. Takový formát učení odpovídá učení na základě chyb. Je to uzavřený systém se zpětnou vazbou, která v sobě neobsahuje zkoumané prostředí. Výkon takového systému můžeme ohodnotit pomocí střední kvadratické chyby nebo sumy kvadrátů odchylek na tréninkovém vzorku reprezentovaném v podobě funkcí volných parametrů systému. Pro takovou funkci můžeme udělat multidimenzionální chybovou plochu (error surface) v souřadnicích volných parametrů. Při tom reálnou plochu chyby průměrujeme (avereged) podle všech dostupných příkladů reprezentovaných v podobě páru vstup-výstup. Libovolná akce provedená v rámci trénování odpovídá jednomu bodu na ploše chyb. Pro zvětšení výkonu systému v čase se musí chybová hodnota posouvat na ploše chyb blíže k minimu. Takové minimum může být jak lokální, tak globální. To můžeme udělat, když má systém užitečné informace o gradientu plochy chyb odpovídající aktuálnímu chování systému. Gradient plochy chyb v libovolném bodě je vektor, který určuje směr nejrychlejšího sestupu na této ploše. V případě učení s učitelem počítáme okamžitý odhad (instantaneous estimate) vektoru gradientu na příkladech, kde je vstupní vektor uvažován jako funkce času. Při využití výsledku takového hodnocení posunu bodu na ploše chyb má obvykle podobu „náhodné procházky“. Nicméně při využití relevantního algoritmu minimalizace nákladové funkce, adekvátního souboru trénovacích dat v podobě vstup-výstup a při dostatku času na učení našeho systému může systém vyřešit problémy rozpoznávání obrazu aproximací funkce atd. [8]

Učení bez učitele (unsupervised) neboli učení na základě sebeorganizace (self-organized) provádíme bez zásahu vnějšího učitele nebo korektoru, který opravuje proces učení. Existuje jen úlohově nezávislá míra kvality (task-independent measure) výkonu, kterou se neuronová síť musí naučit, a volné parametry sítě optimalizujeme podle vztahu k této míře. Po trénování sítě na statistických pravidelnostech vstupního signálu může síť tvořit vnitřní reprezentaci zakódovaných prvků vstupních dat a díky tomu automaticky tvořit nové třídy [9].



Obrázek 2.4: Diagram učení bez učitele [9].

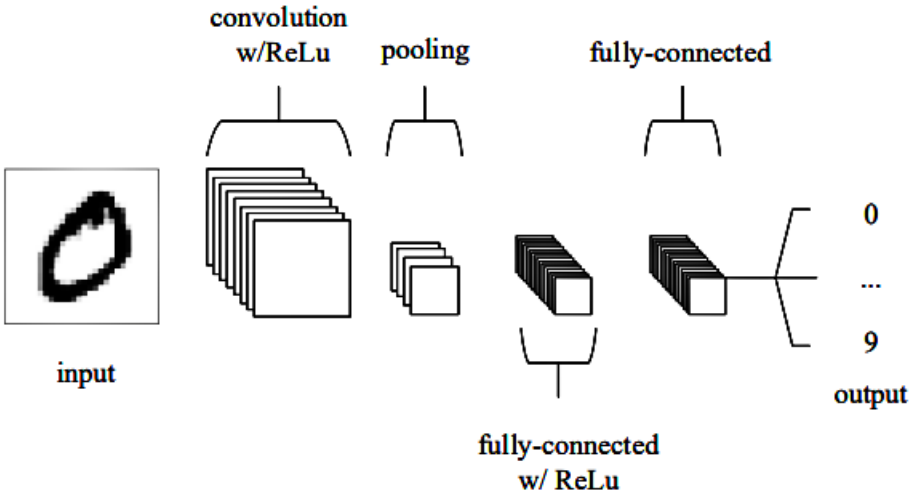
Jako příklad učení bez učitele existuje pravidlo konkurenčního trénování. Například můžeme využít neuronovou síť skládající se ze dvou vrstev – vstupní a výstupní. Vstupní vrstva získává dostupná data. Výstupní vrstva se skládá z neuronů, které soupeří o právo reagovat na signál. Nejzákladnější varianta je princip „vítěz bere vše“, což znamená, že na

vstup reaguje neuron s největším množstvím vstupních signálu a všechny ostatní neurony se deaktivují [8].

2.2.2 Konvoluční neuronové sítě

Konvoluční neuronové sítě (CNN) jsou analogické tradičním ANN v tom, že se skládají z neuronů, které se samy optimalizují během učení. Jediný významný rozdíl mezi CNN a tradičními ANN je ten, že CNN se používají hlavně v oblasti rozpoznávání obrazu. Konceptuální podobnost ale neznamená, že CNN nemají svou specifickou architekturu. Takový exaktní účel se zaměřuje na architekturu, která má být nastavena způsobem, který nejlépe vyhovuje potřebě vypořádat se s konkrétním typem dat, v našem případě vizuálním (jpg, png atd.). Vrstvy uvnitř CNN se skládají z neuronů organizovaných do tří dimenzí: prostoru vstupu (výška a šířka) a hloubky. Hloubku potřebujeme, abychom mohli rozšířit „aktivační objem" do 3. Dimenze, přičemž hloubka nemá vliv na celkový počet vrstev. Na rozdíl od standardních ANN se neurony uvnitř libovolné vrstvy CNN připojí pouze k malé oblasti vrstvy, která předchází. Zmenšením propojení mezi vrstvami se snažíme vyřešit problém „přetrénování" (overtraining). Taková architektura nám dává možnost kondenzovat plnou vstupní dimenzi do menšího objemu třídicích skóre podaných přes hloubkovou dimenzi [6].
Příklad základní architektury CNN:

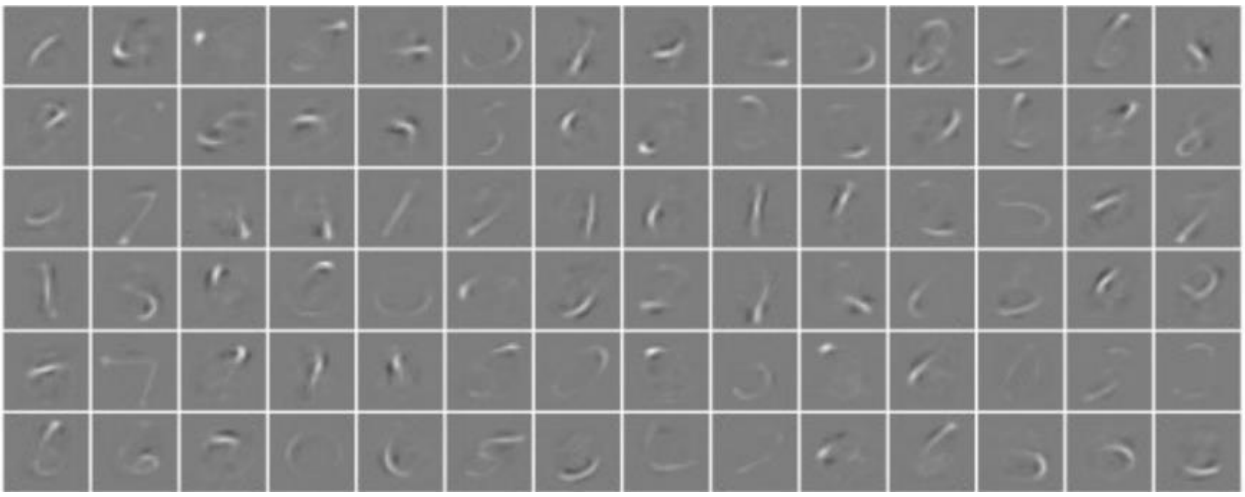
1. **Vstupní vrstva**. Obsahuje hodnoty pixelů obrazu.
2. **Konvoluční vrstva.** Určí výstup neuronů, které jsou připojeny k místním oblastem vstupu pomocí výpočtu skalárního produktu mezi jejich hmotností a oblastí připojenou ke vstupnímu objemu.
3. **Sdružovací vrstva** (pooling layer). Provádí downsampling podél prostorové dimenze daného vstupu, což dále snižuje počet parametrů v rámci této aktivace.
4. **Plně propojené vrstvy** (fully-connected layers). Provádí vytvoření třídicího skóre z aktivací, které mají být použity pro klasifikaci.



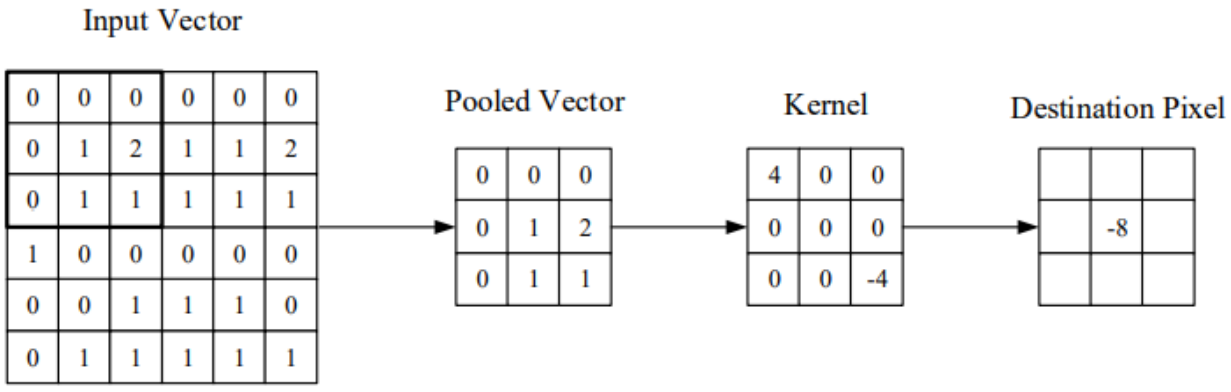
Obrázek 2.5: Základní architektura CNN [6].

Prostřednictvím této jednoduché metody transformace jsou CNN schopny transformovat původní vstup vrstvu po vrstvě pomocí konvolučních a downsamplingových technik k vytvoření třídicích skóre pro účely klasifikace a regrese.

Konvoluční vrstva má největší vliv na to, jak CNN funguje. Podle názvu vidíme, že bází konceptu konvolučních sítí musí být operace konvoluce. Parametry konvolučních vrstev se zaměřují na použití naučitelných jader (kernels). Tato jádra jsou obvykle malá v prostorové dimenzi, ale šíří se po celé hloubce vstupu. Když data narazí na konvoluční vrstvu, vrstva stočí každý filtr přes prostorovou dimenzi vstupu, a vytvoří tak 2D aktivační mapu. (obr. 4.3). Při klouzání vstupu se skalární součin vypočítá pro každou hodnotu v daném kernelu. Z toho se síť naučí jádra, která generují signál neboli reakci, když zaznamená konkrétní funkci v dané prostorové poloze vstupu. Ty jsou běžně známé jako aktivace [6].



Obrázek 2.6: Aktivace převzaté z první konvoluční vrstvy zjednodušující hluboké CNN [6].



Obrázek 2.7: Vizuální reprezentace konvoluční vrstvy [6].

Každý kernel bude mít odpovídající aktivační mapu, která bude vytvořena podél dimenze hloubky, aby vytvořila z konvoluční vrstvy plný výstupní objem. Rozměrnost této

oblasti se běžně označuje jako velikost receptivního pole neuronu. Každý neuron v konvoluční vrstvě je spojen pouze s malou oblastí vstupu. Velikost připojení přes hloubku je téměř vždy rovna hloubce vstupu. Pomocí takového konvolučního procesu dosahujeme značného zmenšení počtu vah. Například, když na vstup do naší sítě pošleme RGB obrázek 64 × 64, kde rozměr signálu je 64 × 64 × 3. Nastavili jsme velikost receptivního pole jako 6 × 6, takže na každém neuronu v konvoluční vrstvě bychom měli celkem 108 vah. V případě odeslání do jiného druhu ANN sítě dosáhl počet vah pro 1 neuron hodnoty 12,288. Konvoluční vrstvy jsou také schopny výrazně snížit složitost modelu pomocí optimalizací jeho výstupu. Výstup můžeme optimalizovat pomocí tří hyperparametrů: hloubky, kroku a nastavení nulové výplně (zero-padding). Hloubka výstupního objemu produkovaného konvolučními vrstvami může být ručně nastavena jako počet neuronů ve vrstvě na stejnou oblast vstupu. Jsme také schopni definovat krok, ve kterém nastavujeme hloubku kolem prostorové dimenze vstupu, abychom umístili receptivního pole. Zero-padding je jednoduchý proces výplně hranice vstupu a je účinnou metodou pro další kontrolu dimenzionality výstupních objemů. Je však důležité si uvědomit, že pomocí těchto technik změníme prostorovou dimenzi výstupu konvolučních vrstev, kterou můžeme vypočítat pomocí vzorce:

$$\frac{(V-R)+2Z}{S+1},$$

kde V reprezentuje velikost vstupního objemu, R reprezentuje velikost receptivního pole neuronu, Z ukazuje na množství provedených zero-paddingů a S odkazuje na informace o kroku. Pokud výsledek této rovnice není roven celému číslu, pak byl krok nesprávně nastaven, protože neurony nebudou schopny přesně projít daným vstupem. Sdílení parametrů funguje za předpokladu, že pokud je jedna funkce regionu užitečná pro výpočet v nastavené prostorové oblasti, pak je velká pravděpodobnost toho, že bude užitečná i v jiné oblasti. Pokud omezíme každou jednotlivou aktivační mapu v rámci výstupního objemu na stejné váhy a odchylky, zaznamenáme masivní snížení počtu parametrů produkovaných konvoluční vrstvou [6].

Sdružovací vrstva je vrstva, která má za cíl postupně snižovat rozměrnost reprezentace, a tím dále snižovat počet parametrů a výpočetní složitost modelu. Sdružovací vrstva provádí operace na každé aktivační mapě. Průměruje jejich rozměrnost pomocí MAX funkce. Kvůli tomu, že tato vrstva odpovídá destruktivnímu chování, jsme silně omezeni v manipulacích, které s ní můžeme provádět. Nevhodným nastavením této vrstvy dojde k velké ztrátě informace o vstupním obrazu a tím i ke ztrátě účinnosti modelu. Takovým omezením je velikost kernelu, která nesmí překročit 3 [6].

Tyto dvě vrstvy představují unikátnost a základní odlišnost CNN od jiných druhů ANN. Námi využité technologie jsou nadstavbou nad CNN sítěmi. Hlavní příčina efektivity konvolučních neuronových sítí je v tom, že místo zaměření se na celkový problém zaměřuje síť pozornost na využití znalostí o konkrétním typu vstupu. Znalosti získané od něj pak může efektivně zpracovávat. To umožňuje použít mnohem jednodušší síťovou architekturu.

Ve své práce Alex Krizhevsky využil CNN [5]. Ta v sobě obsahuje 8 trénovacích vrstev. Pět z nich jsou konvoluční a tři další jsou plně propojené vrstvy. Standardním způsobem je

modelování výstupu neuronu f jako funkce s vstupní proměnnou x pomocí $f(x) = \tanh(\mathrm{x})$ nebo $f(x) = (1 + e^{-x})^{-1}$. Trénování pomocí takových saturačních aktivačních funkci je však pomalejší než nesaturační aktivační funkce $f(x) = \max(0, \mathrm{x})$, proto v této neuronové síti preferujeme nestandardní aktivační funkce. Neurony s nelinearitou $(x) = \max(0, x)$ označujeme jako rektifikované lineární jednotky (ReLUs) [5].

Hluboké CNN s ReLUs se trénují několikrát rychleji než jejich ekvivalenty s jednotkami funkce tanh(x). Podle A. Krizhevského ReLUs dosahají 25% trénovací chyby 6x rychleji než sítě tanh(x). Přitom experiment byl prováděn na stejné architektuře s čtyřvrstvou konvoluční sítí a datasetem CIFAR-10. Podle svého konceptu ReLUs nevyžaduje žádnou normalizaci vstupních dat. Při nalezení normalizace se však efektivita neuronové sítě reprezentované pomocí trénovací chyby zlepšuje v porovnání s nenormalizovaným vstupem. Normalizace byla provedena pomocí funkce:

$$b^i_{x,y} = a^i_{x,y} \,/\, \left(k + a \sum_{j=max(0,i-n/2)}^{min(N-1,i+n/2)} \left(a^j_{x,y}\right)^2\right)^{\beta},$$

Kde $a^i_{x,y}$ je aktivita neuronu vypočtená pomocí aplikace kernelu i na pozici (x, y), $b^i_{x,y}$ reprezentuje normalizovanou aktivitu $a^i_{x,y}$ vypočtenou pomocí aplikace nelinearity ReLU. Součet běží nad n „sousedními“ mapami kernelu ve stejné prostorové poloze a N je celkový počet jader ve vrstvě. Tento druh normalizace odezvy implementuje formu laterální inhibice inspirované funkcí skutečných neuronů konkurujících si pro velké aktivity mezi výstupy neuronů vypočtených pomocí různých kernelů. Konstanty k, n, α a β jsou hyperparametry, jejichž hodnoty jsou určeny pomocí validační sady. Během těchto experimentů A. Krizhevsky nalezl tyto parametry jako k = 2, n = 5, $\alpha = 10^{-4}$, and β = 0,75. Na základě experimentů na datasetu CIFAR-10 bylo zjištěno, že při nenormalizovaném vstupu bylo dosaženo 13% trénovací chyby a při využití této normalizační metody – 11% trénovací chyby [5].

Sdružovací vrstva v CNN shrnuje výstupy sousedících skupin neuronů ve stejné mapě jádra. Skupiny sdružené sousedícími sdružovacími jednotkami se tradičně nepřekrývají. (LeCun Y., 2010, Convolutional networks and applications in vision). Jak jsme uvedli výše, sdružovací vrstva je popsaná pomocí parametrů s a z, kde s je počet pixelů mezi sdružovacími jednotkami a z je rozměrnost z × z bloku, který sdružovací jednotka sdružuje kolem sebe. V tomto případě bylo použito nastavení s = 3, z = 2. Takový překrývající se model nám zlepšuje trénovací chybu na 0,3 % v porovnání s nastavením s = 2, z = 2. Také bylo zjištěno, že u modelů s překrývajícím se sdružováním dochází méně často k přetrénovaní [5].

Výsledná neuronová síť A. Krizhevského ukázala, že velká hluboká CNN je schopna dosáhnout rekordních výsledků na vysoce náročném datovém souboru pomocí čistého učení s učitelem. Je pozoruhodné, že výkon sítě se zhoršuje, pokud je odstraněna jedna konvoluční vrstva. Jeho práce se stala jedním z počátečních impulsů k vývoji v této oblasti strojového zpracování obrazu. V praxi ukázala efektivitu takového postupu zpracování a došla k závěru, že pomocí velkého datasetu můžeme nahradit expertní znalosti ve zkoumané oblasti a také se zbavit problému s hodnocením objektivnosti experta. Za rostoucí popularitou této metody stály taképroporcionálně se zvětšující výpočetní kapacity moderních počítačů [5].

3 Použité neuronové sítě

3.1 Vstupní informace

V této práci jsme využili framework detectron2 založenou na architektuře Mask R-CNN. Tato technologie byla zvolena jako framework pro segmentaci instancí. Segmentace instancí je složitá, protože vyžaduje detekci všech objektů v obraze a zároveň detailní segmentování každé instance. Proto tento framework kombinuje různé úlohy z klasického počítačového vidění. První úlohou je detekce objektů (object detection). Jejím cílem je klasifikovat jednotlivé objekty a každý z nich lokalizovat pomocí bounding boxu. Druhou úlohou je sémantická segmentace, kde cílem je klasifikovat každý pixel do pevné sady kategorií bez rozlišování instancí objektů. Metodu Mask R-CNN rozšiřuje Faster R-CNN přidáním elementu pro predikci segmentačních masek na každou oblast zájmu (Region of Interests) paralelně s existujícími elementy pro klasifikaci a regresi bounding boxu. Tento element nazýváme „větev“ (podle anglického „branch“). Větev masky je malá Fully Convolutional Network (FCN) aplikovaná na každý Region of Interests (RoI) předpovídající segmentační masku způsobem pixel-to-pixel. Díky tomu, že Mask R-CNN je rozšíření Faster R-CNN, je jednoduchá na implementaci a trénování, což usnadňuje širokou škálu flexibilních architektonických návrhů. Kromě toho větev masky pouze nevýznamně zatěžuje výpočetní kapacitu, zároveň však umožňuje natrénovat rychlý systém a provádět rychlé experimentování. I když Mask R-CNN je jen rozšíření, pravidelná konstrukce větve masky je kritická pro dobré výsledky. Samostatný Faster R-CNN nebyl navržen pro zarovnání pixelů (pixel-to-pixel) mezi síťovým vstupem a výstupem [10].

3.2 R-CNN

Pro detailnější pochopení Mask R-CNN musíme pochopit, co je samotné R-CNN a jaký změnami se od jiných druhů konvolučních neuronových sítí. R-CNN můžeme rozklíčovat jako „Regions with CNN features“ (Regiony s CNN funkcí). Nejprve nás zajímá, jak R-CNN detekuje objekty ve vstupním obrazu. Systém detekce objektů se skládá ze tří modulů. První generuje návrhy regionů nezávislých na kategoriích. Tyto návrhy definují sadu potenciálních kandidátských objektů, které budou k dispozici pro náš detektor/klasifikátor. Druhým modulem je velká konvoluční neuronová síť, která z každé oblasti extrahuje feature vector s pevnou délkou. Třetím modulem je sada lineárních SVM (Support Vector Machines) specifických pro třídy [11].

Nejprve zmíníme „návrhy regionů“. Rozlišovacím znakem R-CNN je to, že narozdíl od jiných sítí, které se zaměřují na metody nalezení exaktního regionu, je nezávislá na metodě návrhu konkrétního regionu. V ní R. Girshick použil selektivní vyhledávání, které umožňuje kontrolované srovnání s předchozí detekční prací. Selektivní vyhledávání je založeno na principu hledání hierarchie uvnitř vstupního obrazu. Jinak řečeno, R-CNN využívá hierarchický seskupovací algoritmus, který tvoří základ našeho selektivního vyhledávání. Protože samotný proces seskupování je hierarchický, můžeme přirozeně generovat umístění ve všech měřítcích pokračováním procesu seskupování, dokud se celý

obrázek nestane jedinou oblastí. (J. Uijlings, 2013, Selective search for object recognition.) Důležitý je rovněž proces hledání takzvaných „features“. V případě R-CNN z každého návrhu oblasti extrahujeme 4096dimenzionální vektorový prvek. Pro jeho nalezení se využívá open source CNN knihovny Caffe (Convolutional Architecture for Fast Feature Embedding) [13]. Funkce se počítají jako dopředné šíření průměrně odečteného 227 x 227 RGB obrazu přes pět konvolučních vrstev a dvě plně propojené vrstvy. (Neuronová síť A. Krizhevského). Pro výpočet funkce pro návrh regionu musíme nejprve převést obrazová data v této oblasti do podoby, která je kompatibilní s CNN, v našem případě odpovídající 227 × 227 pixelů. Při tom využijeme nejzákladnější logiku. Nezávisle na rozměru oblasti deformujeme všechny pixely v těsném bounding boxu kolem něj na požadovanou velikost. Před deformací rozšiřujeme těsný bounding box tak, aby při deformované velikosti bylo kolem původního rámečku přesně P pixelů informace kolem obrazu. (Obr. 5.1) [11].

Obrázek 3.1: Deformované vstupní trénovací obrazy [11].

Proces trénování je také velmi důležitý. R-CNN během svého trénování využívá předtrénování (pre-training) s učitelem. Předtrénování bylo provedeno na velkém pomocném datovém souboru (ILSVRC 2012) s anotacemi na úrovni obrazu. To znamená, že nebyla provedena označení pro bouding boxy. Toho bylo dosaženo s využitím CNN knihovny Caffe, stejně tak, jako pro nalezení feature vektoru [13].
Pak se provádí v R-CNN ladění pro specifické domény. (Domain-specific fine-tuning.) Důvodem ladění je to, aby naše síť mohla provést adaptaci na novou úlohu (detekce) a novou doménu (deformování VOC okna/obrázku). Jinak řečeno, pokračujeme v trénování parametrů CNN pomocí metody „stochastic gradient descent“ (SGD) pouze s využitím deformovaných regionů od VOC. Kromě nahrazení 1000. klasifikační vrstvy specifické pro CNN ImageNet náhodně inicializovanou 21. klasifikační vrstvou (pro 20 tříd VOC plus pozadí) se architektura R-CNN nezmění v porovnání s CNN ImageNet. Síť předpokládá, že všechny návrhy regionů s ≥ 0,5 IoU (intersection-over-union) se překrývají s ground-truth box jako pozitiva pro třídu tohoto class boxu a zbytek jako negativa. Začínáme SGD s trénovací rychlostí 0,001, jinak řečeno 1/10 z počáteční rychlosti předběžného tréninku, což umožňuje dosáhnout pokroku v jemném ladění, aniž bychom museli zasahovat do inicializace. V každé iteraci SGD jednotně vzorkujeme 32 pozitivních oken (ve všech třídách) a 96 oken na pozadí, abychom vytvořili mini-dávku o velikosti 128. Zkreslujeme vzorkování směrem k pozitivním oknům, protože jsou ve srovnání s pozadím extrémně vzácná [11].
Třetí modul nese název „klasifikátory kategorií objektů“ (Object category classifiers). Ve své práci předvádí R. Girshick příklad s klasifikací auta. Vytvořil binární klasifikátor, který říká,

co „je auto“ a co „není auto“ [11]. Je pochopitelné, že oblast obrazu těsně obklopující auto by měla být pozitivním příkladem. Podle stejné logiky musí být pozadí klasifikováno jako negativní příklad. Zůstávají však oblasti částečně překrývající auto. Tento problém byl řešen prahem překrývání IoU, pod kterým jsou regiony definovány jako negativa. Prahová hodnota překrytí 0,3 byla vybrána vyhledáváním mřížky nad (0, 0,5) na validační sadě. Při nastavení tohoto parametru na 0,5 dochází ke snížení mAP o 5 bodů. (J. Uijlings, 2013, Selective search for object recognition.). Podle experimentu, kdy byla prahová hodnota překrytí nastavena na 0, pozorujeme snížení mAP o 4 body. Z toho vyplývá, že nastavení tohoto parametru je velmi důležité pro finální výsledek. Pozitivní příklady jsou definovány jednoduše jako hraniční ground-truth boxy pro každou třídu. Jakmile jsou funkce extrahovány a jsou použity štítky školení, optimalizuje se jeden lineární SVM na třídu [11]. V případě, kdy jsou trénovací data příliš velká na to, aby se vešla do paměti, přijmeme standardní tvrdou negativní metodu těžby (standard hard negative mining method) [12]. Tato metoda konverguje rychle a v praxi se mAP přestane zvyšovat po jediném průchodu všemi obrázky. Díky R-CNN došlo ve strojovém zpracování obrazu k využití neuronových sítí předtrénovaných na velké databázi ještě před aplikací na konkrétní úloze.

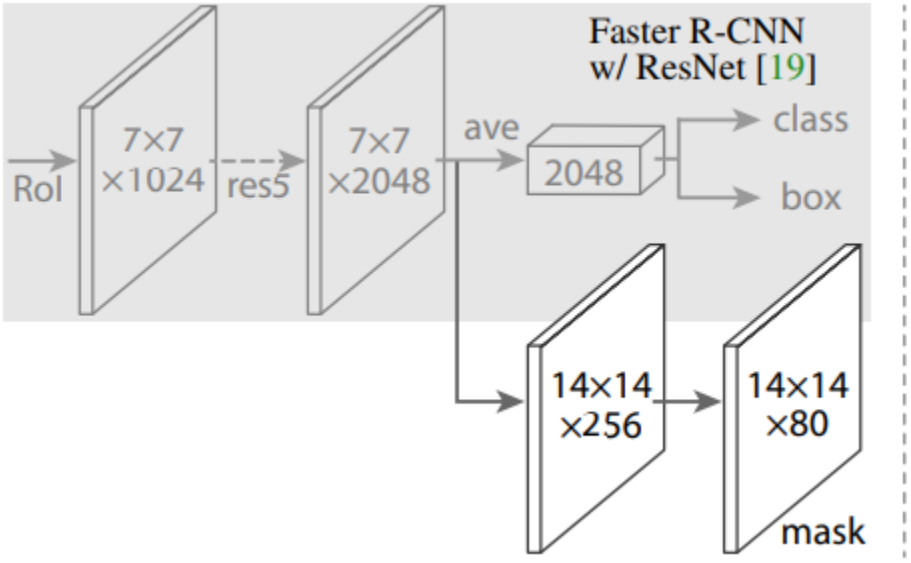
3.3 Mask R-CNN

Koncept Mask R-CNN je velmi pochopitelný. Faster R-CNN obsahuje 2 výstupy pro každý kandidátský objekt: „class label“ a „pole bounding-boxu“. K tomu přidáme 3. větev, která vydává masku objektu. Další výstup masky se však liší od výstupů třídy a boxu – vyžaduje extrakci mnohem jemnějšího prostorového uspořádání objektu.
Fast R-CNN se skládá ze dvou fází. První fáze, nazvaná Region Proposal Network (RPN), navrhuje bounding boxy kandidátských objektů, jak jsme uvedli výše. Druha fáze je podstatou celé Faster R-CNN. Extrahuje příznaky pomocí RoIPool z každého kandidátského boxu a provádí klasifikaci a regresi bounding-boxu [10][14].

Mask R-CNN plně odpovídá první fázi Fast R-CNN. Ke změnám dochází až ve druhé fázi, kde výstup první fáze paralelně s obvyklým Fast R-CNN posílá binární masku na výstup pro každý region of interests (RoI). Z tohoto důvodu během trénování definujeme ztrátovou funkci pro každý RoI v podobě $L = L_{cls} + L_{box} + L_{mask}$. L_{cls} je ztrátová funkce klasifikace a L_{box} je ztrátová funkce bounding-boxu. $L_{cls} = -\log p_u$, kde p je diskrétní rozdělení pravděpodobnosti (za RoI) nad K + 1 kategoriemi. p se vypočítá pomocí funkce softmax na výstupech K + 1 plně propojené vrstvy. K je počet tříd klasifikovaných objektů. L_{box} je definována n-tící regresních cílů „true” bounding-boxů pro třídy $u, v = (v_x, v_y, v_w, v_h)$ a předpokládaná n-tice $t^u = (t_x^u, t_y^u, t_w^u, t_h^u)$, pro třídu u. Pomocí Iversonovy funkce jsme vyhodnotili třídu u. h, w zde představují výšku a šířku exaktního RoI, a x, z jsou souřadnice horního levého úhlu. Podle toho je třída pozadí označena u = 0. Samotná ztrátová funkce pro bounding-box vypadá takto: $L_{box} = \sum_{i \in \{x,y,w,h\}} smooth_{L1}(t_i^u - v_i)$, kde $smooth_{L1}(x) = \begin{cases} 0{,}5x^2, když\ |x| < 1 \\ |x| - 0{,}5,\ v\ jiných\ případech. \end{cases}$ [10].

Větev masky má Km^2 dimenzionální výstup pro každou RoI, která kóduje K binárních

masek s rozlišením m × m, jednu pro každou třídu K. K tomu použijeme per-pixel sigmoid a definujeme masku jako průměrnou ztrátu binární křížové entropie. Pro RoI spojený s ground-truth třídou k je L_{mask} definovaná pouze na k-té masce (ostatní výstupní masky nepřispívají ke ztrátě). Taková definice ztrátové funkce masky umožňuje síti generovat masky pro každou třídu bez konkurence mezi třídami. Spoléháme se na vyhrazenou klasifikační větev, abychom předpověděli označení třídy použité k výběru masky výstupu. Toto odděluje masky a predikce třídy. Maska kóduje prostorové uspořádání vstupního objektu. Na rozdíl od labelu třídy a offsetů boxu, které jsou jednoznačně reprezentované pomocí výstupního vektoru generovaného plně propojenou vrstvou, extrahování prostorové struktury masek je řešeno pomocí vztahu pixel-to-pixel řešeného pomocí konvoluce [14]. K. He vybral ve své práci architekturu Faster R-CNN ResNet C4, popsanou v jeho článku Deep residual learning for image recognition, ke které pak přidal větev masky. Na obr. 5.2 vidíme vizualizaci této architektury. Čísla na obrázku označují prostorové rozlišení a kanály. Šipky označují konvoluční, dekonvoluční nebo plně propojené vrstvy. (Konvoluční sníží prostorovou dimenzi, zatímco dekonvoluční ji zvýší). Všechny konvoluční vrstvy jsou 3 × 3, s výjimkou výstupu, který je 1 × 1, dekonvoluční jsou 2 × 2 s krokem 2 a ReLU používáme ve skrytých vrstvách. Mask R-CNN však nemusí vždy mít stejnou architekturu [14].

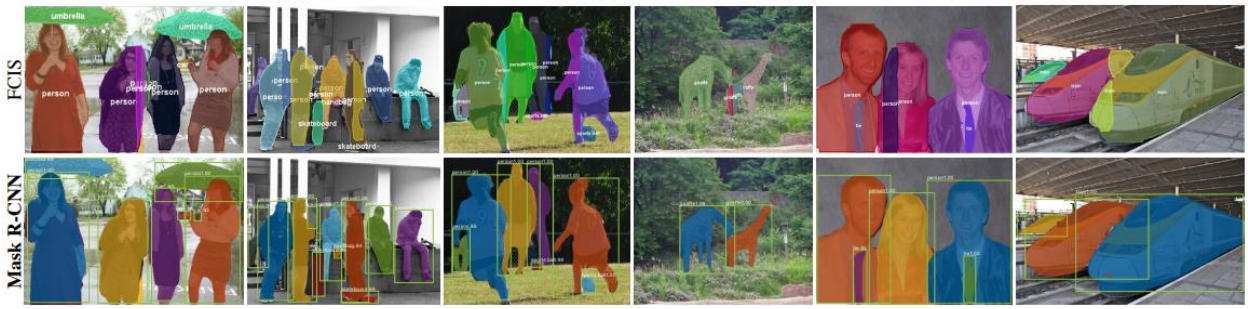


Obrázek 3.2: Architektura Faster R-CNN ResNet s přidanou maskou [14].

Proces trénování Mask R-CNN je také hodně spojen s Faster R-CNN. Odtud získáváme hodnocení RoI. Když je IoU hodnota ground-truth boxu ≥ 0,5 klasifikujeme ji jako „pozitivní“ případ, v opačném případě jako „negativní“. Ztrátovou funkci masky L_{mask} zavedeme jenom pro „pozitivní“ RoI. Cíl masky je průsečík RoI a jemu odpovídající ground-truth masky. Pro trénování jsme použili princip „zaměření na obraz“ (image-centric training). Obrázky jsou zvětšeny tak, že jejich měřítko je 800 pixelů. Každá iterace posílá 2 obrázky na GPU a každý obrázek má N vzorkovaných RoI s poměrem 1:3 pozitivních k negativním. Pro vybranou architekturu ResNet C4 N=64. Celkem má proces následující

algoritmus: Nejprve na návrzích provozujeme větev predikce boxů následovanou non-maximum potlačením. (Počet návrhů pro C4 je 300). Poté aplikujeme větev masky na detekční boxy s nejvyšším skóre 100. To se liší od paralelního výpočtu používaného při tréninku, což urychluje odvození a zlepšuje přesnost díky použití menšího počtu přesnějších RoI. Větev masky dokáže předpovědět K masek na RoI, ale používáme pouze k-tou masku, kde k je předpovězená třída podle větve klasifikace. Protože počítáme pouze masky na top 100 detekčních boxech, Mask R-CNN přidává malou režii svému rychlejšímu protějšku R-CNN.

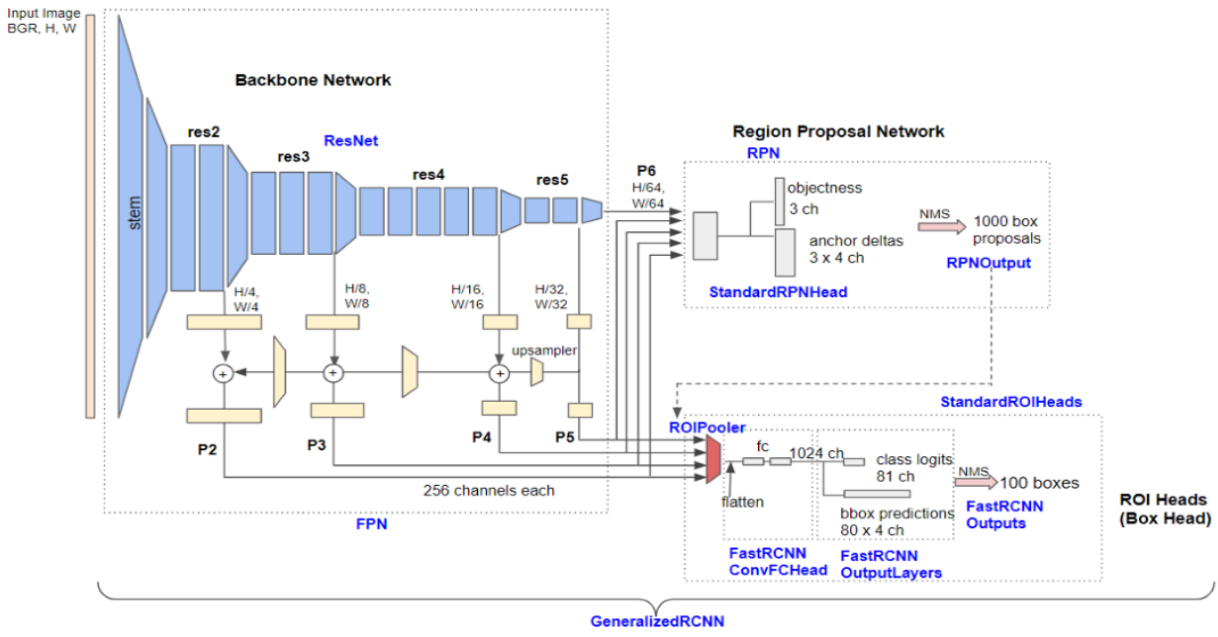
Podle experimentu dosahuje Mask R-CNN dobrých výsledků i za náročných podmínek. V porovnání s FCIS+++ nevykazuje systematické artefakty na překrývajících se instancích. Můžeme to vidět na obr. 5.3 [14].



Obrázek 3.3: FCIS+++ (top) vs. Mask R-CNN (bottom) [14].

Vidíme, že masky, které jsou součástí Mask R-CNN, významně zlepšují detekci v detailních obrázcích s překrývajícími se elementy. Proto je tato metoda využitelná pro segmentaci orgánů [14].

3.3 Detectron2



Obrázek 5.3 Detailní architektura Base-RCNN-FPN. (Modré štítky představují názvy tříd.)

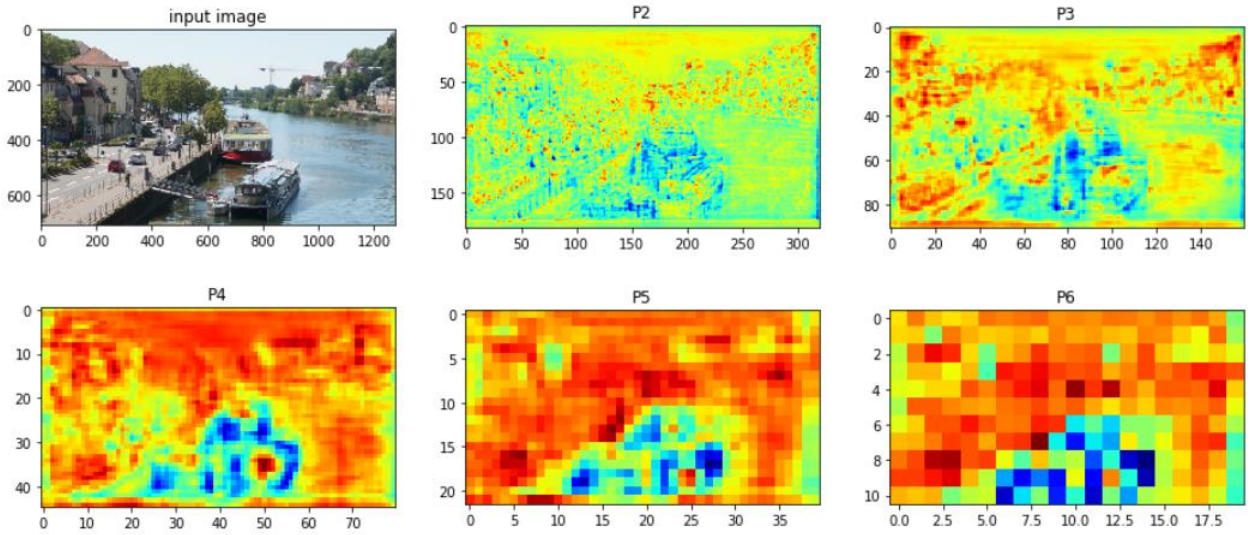
Tento graf reprezentuje architekturu, které využívá detektron2, který jsme využili pro segmentaci parenchymatózních orgánů prasete domácího. [16]

3.3.1 Backbone Network

Nejprve se podíváme na „Backbone Network“, neboli „Feature Pyramid Network“ (FPN). Úlohou páteřní sítě (FPN) je extrahovat feature mapy ze vstupního obrazu. Na vstup posíláme input (torch.Tensor): (B, 3, H, W) obrázek. B, H a W znamenají velikost dávky, výšku obrazu a šířku. Vstupní obrázek musí mít barevné kanály v pořadí BGR, nikoliv RGB. Výstupní signál je output (dict of torch.Tensor): (B, C, H / S, W / S) feature maps. C a S znamenají velikost kanálu a krok. Ve výchozím nastavení C = 256 pro všechny stupnice A, S = 4, 8, 16, 32 a 64 pro výstupy P2, P3, P4, P5 a P6. Příklad: jestliže pošleme jako vstupní signál obrázek o rozměrech H = 800, W = 1280, pak velikost vstupního tenzoru je „torch.Size([1, 3, 800, 1280]“ a výstupním signálem bude následující slovník (dictionary):

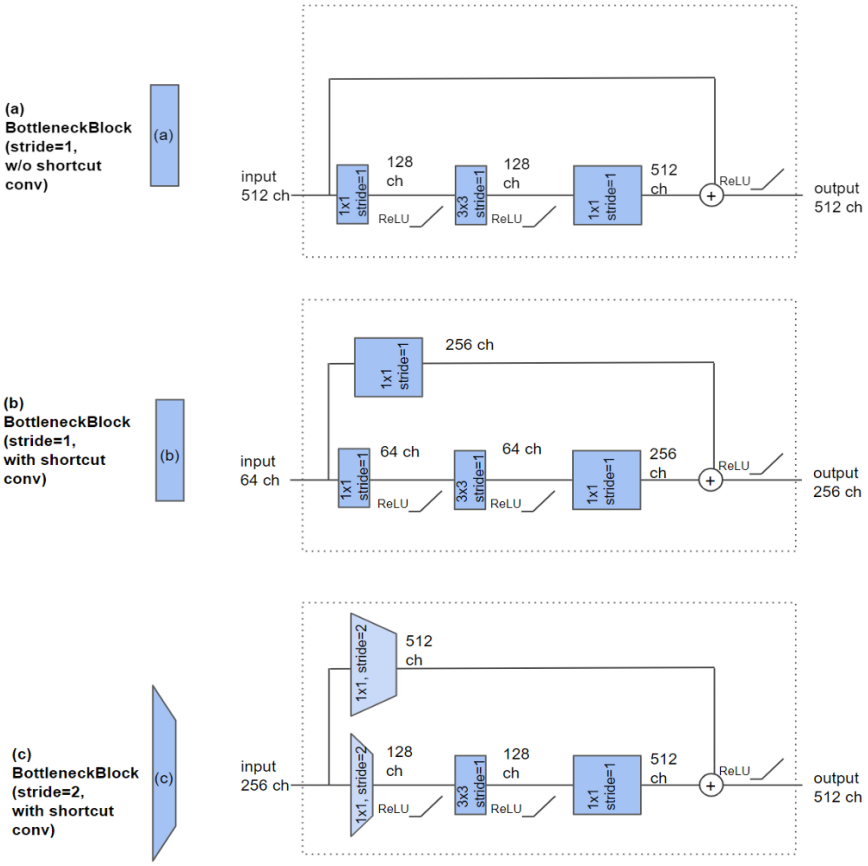
```
output["p2"].shape -> torch.Size([1, 256, 200, 320]) # stride = 4
output["p3"].shape -> torch.Size([1, 256, 100, 160]) # stride = 8
output["p4"].shape -> torch.Size([1, 256, 50, 80])   # stride = 16
output["p5"].shape -> torch.Size([1, 256, 25, 40])   # stride = 32
output["p6"].shape -> torch.Size([1, 256, 13, 20])   # stride = 64
```

Lehčí pochopení můžeme získat na základě konkrétního příkladu na níže uvedeném obrázku. Na obrázku 5.4 vidíme, jak výstupní feature mapy vypadají. Jeden pixel funkce P6 odpovídá širší oblasti vstupního obrazu než P2 – jinými slovy P6 má větší receptivní pole než P2. FPN může extrahovat multi-měřítkové feature mapy s různými receptivními poli [16].



Obrázek 3.4: Příklad vstupu a výstupu FPN [16].

Pro modelování Backbonu využili vědci ResNet. Ten se skládá z kmenového (stem) bloku a „fází“, které obsahují více úzkých bloků (bottleneck blocks). Stem blok dvakrát zmenšuje vstupní obrázek pomocí 7 × 7 konvoluce se stride = 2 a maximálního (max) sdružování se stride = 2. Výstupem stem bloku je tenzor feature mapy. Bottleneck blok obsahuje tři konvoluční vrstvy, jejichž kernel je velikosti 1 × 1, resp. 3 × 3, 1 × 1. Čísla vstupních a výstupních kanálů konvoluční vrstvy 3 × 3 jsou pro efektivní výpočet menší než vstup a výstup bloku. Existují 3 typy různých bottleneck bloku, což vidíme na obrázku 5.5 [15].

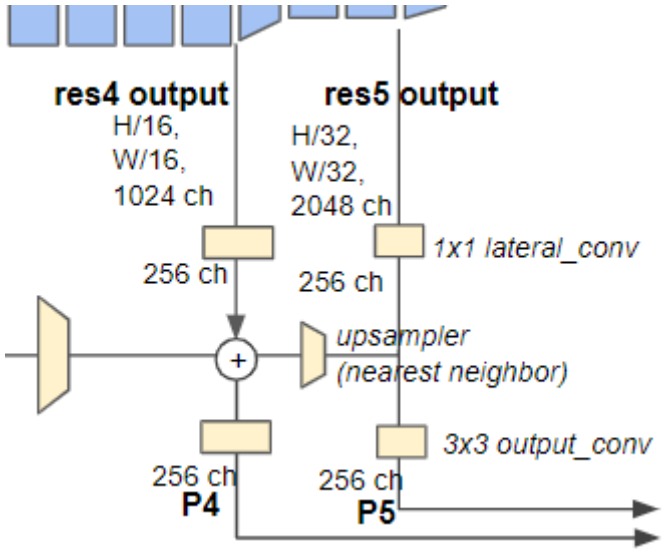


Obrázek 3.5: Tři typy bottleneck bloku [16].

ResNet rovněž obsahuje zkrácenu konvoluční vrstvu (shortcut), která přidává funkce vstupu a výstupu. Pro první blok stupně (block of a stage) se zkrácená konvoluční vrstva využívá pro porovnání počtů kanálů vstupu a výstupu. Zkratka je využita ve variantách bottleneck bloku „b“ a „c“ na obrázku 5.5. Varianta „c“ kromě toho využívá downsampling feature mapy pomocí konvoluční vrstvy s stride = 2.

Další součástí je FPN, která obsahuje ResNet, boční a výstupní konvoluční vrstvy, up-samplery a last-level maxpool vrstvu. První vrstva se nazývá „boční“ konvoluce, protože FPN je původně zobrazena jako pyramida, kde je kmenová vrstva umístěna na dně. Výstupní konvoluční vrstva obsahuje konvoluci 3 × 3, která nemění počet kanálů. Dopředný proces v FPN provádíme pro hledání čtyř tenzorů označených P2 (1/4 měřítka),

P3 (1/8), P4 (1/16) and P5 (1/32). Tento proces začíná ve výstupu res5. (obr. 5.6) Po průchodu boční konvolucí posíláme získanou 256kanálovou feature mapu do výstupní konvoluce, aby byla zapsána do seznamu výsledků jako P5 (měřítko 1/32). Poté stejný signál přivádíme do up-sampleru a získáme P4 (1/16). Totéž provedeme ještě třikrát a získáme ostatní P. Pro hledání P6 ale musíme přidat max pooling vrstvu o velikosti kernelu = 1 a stride = 2 do finálního bloku ResNet. Tato vrstva převádí P5 na P6 (1/64 měřítka) [16].



Obrázek 3.6: Část FPN obsahující res4 a res5 [16].

Pro trénování detekčního modelu využívá detecrton2 vstupního páru obrázek-anotace. Tato data jsou používána v RPN a Box Headu, stejně jako ve Fast R-CNN. RPN se nenaučí klasifikovat kategorie objektů, štítky kategorií se tedy používají pouze v hlavách RoI. Vytváří síť dataset s názvem mydataset, kam kopíruje všechny naše páry, přičemž potřebujeme masky ve formátu JSON. Potém probíhá proces mapování, během kterého jsou registrované záznamy anotace vybírány jeden po druhém. Potřebuje síť aktuálních obrazových dat (nikoli cestu) a odpovídající anotace. Dataset mapper se zabývá přidáním 'image' a 'instance' z těchto záznamů do dataset_dict pomocí transformací nad obrazy. Anotace 'bbox' jsou registrovány do objektů struktury polí, které mohou být uloženy jako seznam bounding boxů [16].

3.3.2 Region Proposal Network (RPN)

Výstup z Backbone posíláme na RPN. Každá velikost tenzoru udává dávku, kanály, výšku a šířku. Tyto parametry featuru používáme v RPN. Na RPN posíláme také boxy z našeho datasetu. Kvůli tomu, že báze detectron2 je Mask R-CNN, kopíruje její funkce a architekturu. (viz 3.3 Mask R-CNN) [16].

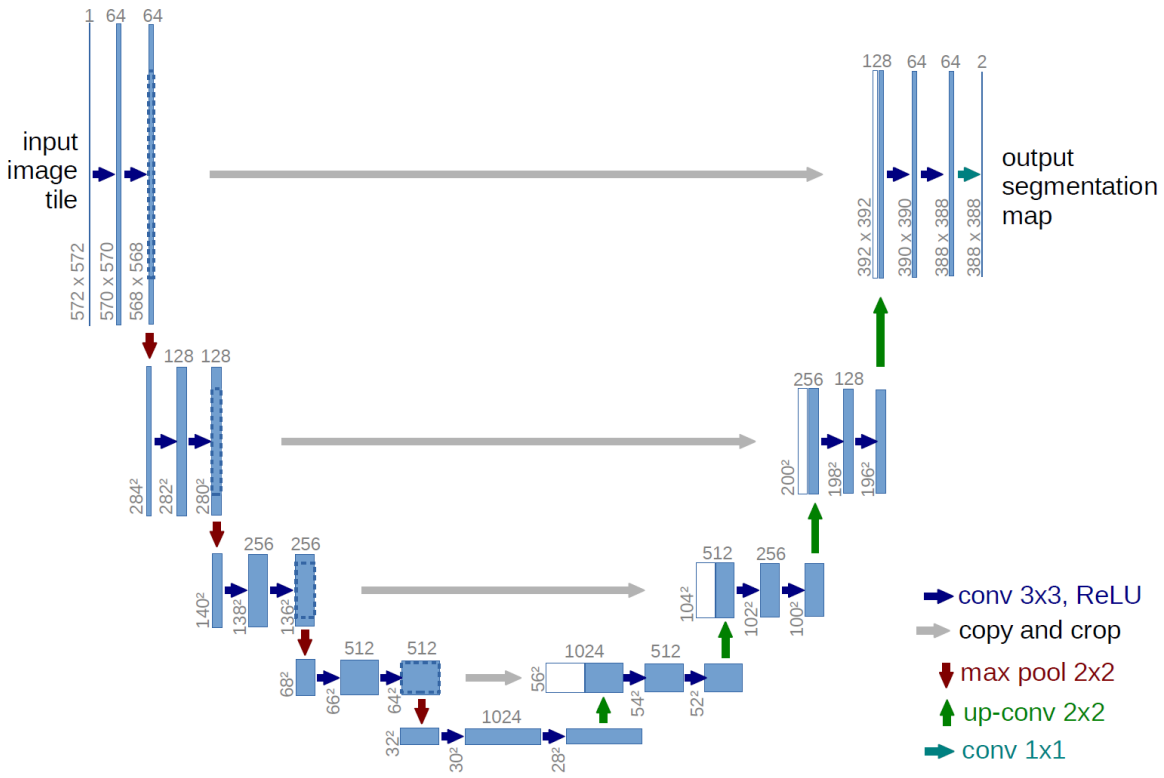
3.3.3 Region of interests (ROI) Head

RoI také vyžaduje výstup z Backbone. Kromě toho však potřebuje feature mapy, které jsou výstupy z RPN. Tudy probíhá trénování klasifikace a také sledování ztrátové funkce. Tato součást také pochází z architektury Mask R-CNN. (viz 3.3 Mask R-CNN) [16].

3.4 U-NET

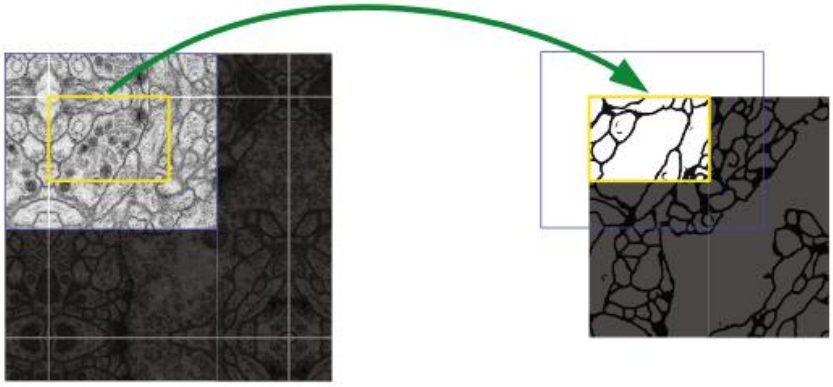
Architektura U-NET byla představena v roce 2015 Olafem Ronnebergem, Philippem Fischerem a Thomasem Broxem. Byla vyvinuta pro segmentaci v medicínské oblasti. V naší práci jsme porovnali výsledky získané pomocí U-NET s výsledky získanými pomocí detekční sítě s architekturou detektron2. Za bázi architektury U-NET byla zvolena architektura nazývaná „fully convolutional network" [19]. Její hlavní myšlenkou je doplnit běžnou kontrakční síť po sobě jdoucími vrstvami, kde jsou operátory pro seskupování nahrazeny operátory pro zvětšování. Tímto způsobem tyto vrstvy zvyšují rozlišení výstupu. Aby bylo možné výstup lokalizovat, jsou vysokorozlišovací příznaky z kontrakční cesty kombinovány se zvětšeným výstupem. Poté se může následující konvoluční vrstva naučit sestavit přesnější výstup na základě této informace [18].

Architektura se skládá z kontrakční a expanzní cesty. Kontrakční cesta následuje typickou architekturu konvoluční sítě. Skládá se z opakované aplikace dvou 3 × 3 konvolucí, jinak řečeno konvolucí bez nulového doplnění, z nichž je každá následována usměrněnou lineární jednotkou ReLU a operací max poolingu o velikosti 2 × 2 s krokem 2 pro zmenšení vzorkování. Zmenšení vzorkování potřebujeme pro zvětšení počtu příznakových kanálů. V našem případě každý takový krok zdvojnásobuje počet. Expanzní cesta funguje opačně. Počet jejích kroků je stejný jako u kontrakční cesty. V poslední vrstvě se používá 1 × 1 konvoluce k namapování každého 64složkového vektoru příznaků na požadovaný počet tříd. Celkově má síť 23 konvolučních vrstev. Její architekturu můžeme vidět na obrázku 3.7 [19].



Obrázek 3.7: Architektura U-NET (příklad pro nejnižší rozlišení 32 × 32 pixelů). Každý modrý blok odpovídá mnohokanálové mapě příznaků. Počet kanálů je uveden nad blokem. Velikost x-y je uvedena v levém dolním rohu bloku. Bílé bloky představují zkopírované mapy příznaků. Šipky označují různé operace. [19]

Důležitou modifikací U-NET v porovnání s fully convolutional network je, že má velký počet příznakových kanálů ve zvětšovací části, což síti umožňuje přenášet informace o kontextu do vrstev s vyšším rozlišením. Jako důsledek je expanzní cesta téměř symetrická s kontrakční cestou a vytváří architekturu ve tvaru U. Síť nemá žádné plně propojené vrstvy a používá pouze platnou část každé konvoluce, což znamená, že mapa segmentace obsahuje pouze pixely, pro které je dostupný celý kontext vstupního obrazu. Tato strategie umožňuje plynulou segmentaci libovolně velkých obrázků pomocí strategie overlap-tile. Pro předpovídání pixelů v okrajové oblasti obrázku je chybějící kontext extrapolován zrcadlením vstupního obrázku. Tato strategie Overlap-tile je důležitá pro použití sítě na velké obrázky, protože jinak by rozlišení bylo omezeno pamětí GPU. Strategie je zobrazena na obrázku 3.8 [19].



Obrázek 3.8: Strategie Overlap-tile pro bezproblémovou segmentaci libovolně velkých obrázků (zde segmentace neuronálních struktur v zásobnících („stacks“) elektronové mikroskopie) [19].

Predikce segmentace v žluté oblasti vyžaduje použití obrazových dat v modré oblasti jako vstupu. Chybějící vstupní data jsou extrapolována zrcadlením [19].

I když na začátku byla síť vyvinuta pro detekci buněk a využita v analýze mikroskopických dat, nyní je hojně využívána při segmentaci orgánů na CT snímcích, protože při segmentaci orgánů narážíme na obdobné problémy. Například problém nízké kontrastnosti orgánů oproti pozadí a malá délka mezi objekty uvnitř břišní dutiny [20].

4 Praktická část

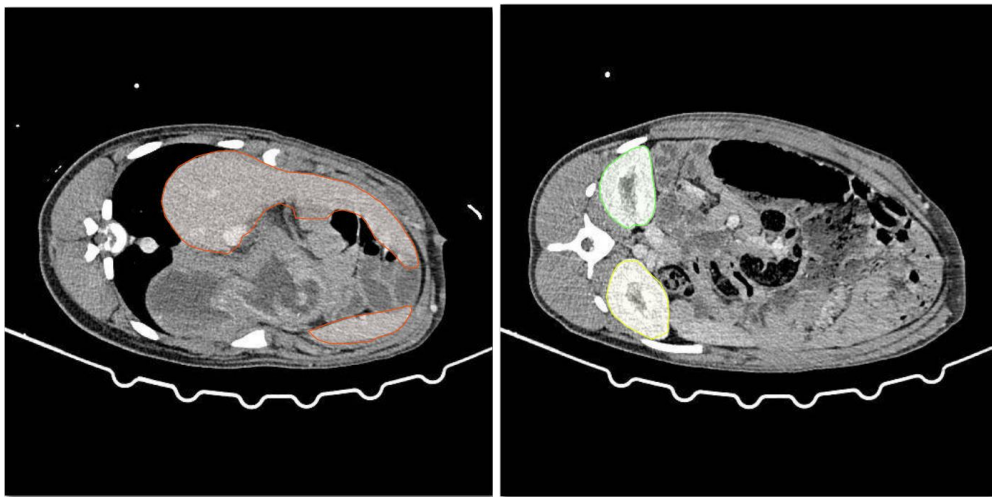
4.1 Popis datasetu

V rámci spolupráce mezi Západočeskou univerzitou v Plzni a Biomedicínským centrem Lékařské fakulty Univerzity Karlovy v Plzni vznikl projekt zaměřený na podporu experimentální medicíny. Projekt se zabývá vývojem možností léčby rakoviny jater pomocí prasat domácích jako subjektů experimentu. Jedním z cílů projektu je automatizace detekce parenchymatózních orgánů. Získané masky orgánů jsou vhodné pro analýzu a výpočet zdravých částí orgánů, které by odpovídaly normě naměřené lékaři u zdravých jedinců. Dataset obsahuje celkem 14 pacientů v podobě 3D objektů získaných z CT snímků ve formátu DICOM, které poskytují 3D model vnitřní stavby organismu.

Takový počet unikátních dat představoval hlavní motivaci pro využití neuronové sítě Detectron2, která je specializovaná na detekci objektů ve 2D prostoru. Původní dataset obsahující 14 prasat jsme považovali za nedostatečně velký, proto jsme se rozhodli převést data do formátu PNG, přičemž jsme využili řezů, ze kterých byl pomocí metody DICOM vytvořen 3D model. Tímto přístupem jsme získali celkem 12 986 unikátních 2D obrázků. Každý z těchto 2D obrázků měl rozměry 512 pixelů na 512 pixelů a byl transformován do šedotónového barevného spektra.

Barevné spektrum bylo využito k reprezentaci denzity objektů na obrázcích, přičemž

černá barva představovala břišní dutinu a ostatní objekty s denzitou rovnou nule, zatímco bílá barva indikovala objekty s nejvyšší denzitou, v našem případě kosti. Orgány a svaly na obrázcích měly barevné spektrum mezi černou a bílou barvou, které odráželo jejich denzitu. Vzhledem k odlišnostem v pozadí a složitosti tvarů orgánů mělo učení neuronové sítě vliv na schopnost detekovat jednotlivé objekty. Například na obrázku 4.1 byl demonstrován rozdíl mezi ledvinami a játry. Pro usnadnění interpretace byly jednotlivým obrázkům přidány anotace, které označovaly přítomnost konkrétního orgánu.



Játra Ledviny

Obrázek 4.1: Informace od učitele: anotace jater a ledvin.

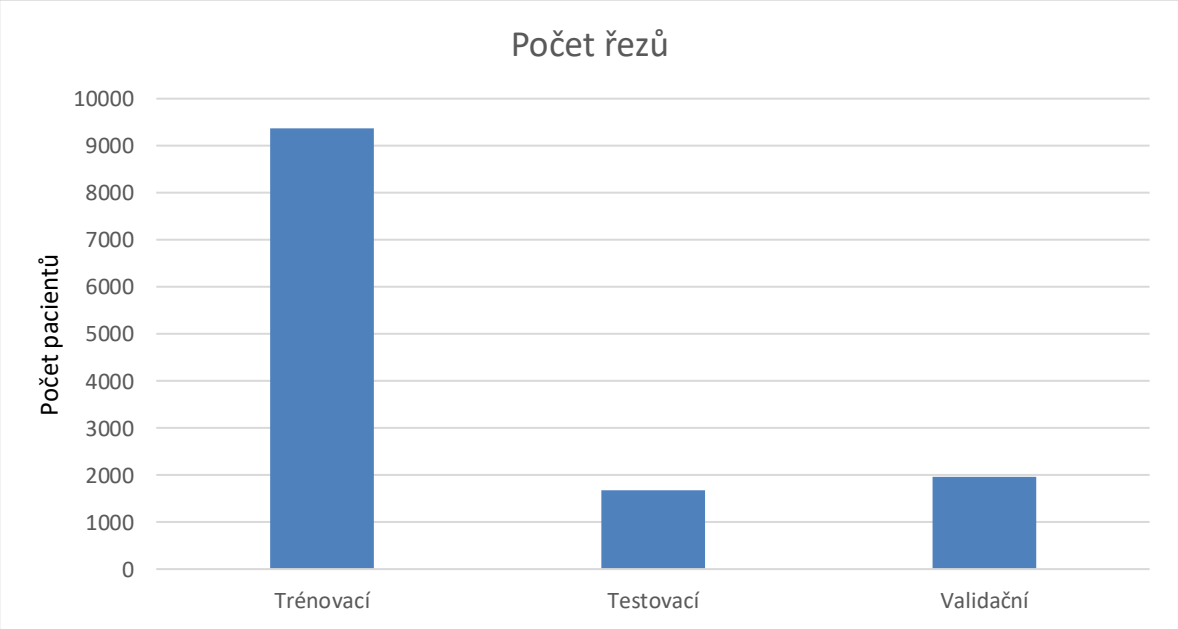
Na obrázku 4.1 je dobře patrné, že jak pro lidskou detekci, tak pro detekci počítačem jsou ledviny snadněji rozpoznatelné než játra. Ledviny mají výraznější kontrast v porovnání s játry, která mají téměř stejnou denzitu jako svaly v jejich okolí. Forma jater je navíc složitější a není vždy jednoduché identifikovat všechny jejich části na jednom řezu. Bez 3D kontextu mohou játra působit chaoticky. Proto jsme pro náš experiment vybrali právě tyto tři orgány. Játra představují finální cíl našeho výzkumu a jejich detekce má pro nás nejvyšší prioritu. Ledviny jsou na druhou stranu příkladem parenchymatózních orgánů, které jsme zahrnuli do naší metodologie, abychom porozuměli vhodnosti této metody pro práci s orgány a perspektivy segmentace pacienta ve 2D reprezentaci.

Získání správných datových sad pro trénování bylo nezbytné pro úspěšnou a korektní tréninkovou fázi. Data jsme rozdělili na tři části: trénovací, testovací a validační. Pro spravedlivé rozložení a vyvážení jsme toto rozdělení prováděli s ohledem na 3D objekty, aby řezy pocházející od jednoho pacienta náležely pouze jedné z těchto tříd. Konkrétně jsme do trénovací datové sady zařadili řezy z 10 pacientů, do testovací sady jsme zařadili řezy z 2 pacientů a do validační sady řezy z dalších 2 pacientů. Přesné zařazení pacientů ke specifickým datovým sadám je uvedeno v tabulce 4.2.

Název datasetu	Číslo pacienta									
Trénovací	33	34	35	36	37	38	39	28	45	46
Testovací			25					27		
Validační			32					44		

Tabulka 4.2: Rozdělení pacientů do trénovací, testovací a validační sady.

Z hlediska neuronové sítě je trénovací dataset hlavním zdrojem informací, a proto jsme se do něj snažili zahrnout co největší počet unikátních pacientů. Pro objektivní vyhodnocení výsledků je nutné mít minimálně rozumný počet testovacích objektů, čehož jsme dosáhli zařazením 2 pacientů do testovacího datasetu. Abychom zajistili, že se neuronová síť neomezí pouze na imitaci jednoho pacienta, rozhodli jsme se pro validační dataset vybrat další 2 pacienty. Obrázek 4.3 ukazuje rozložení počtu řezů v rámci datových sad.



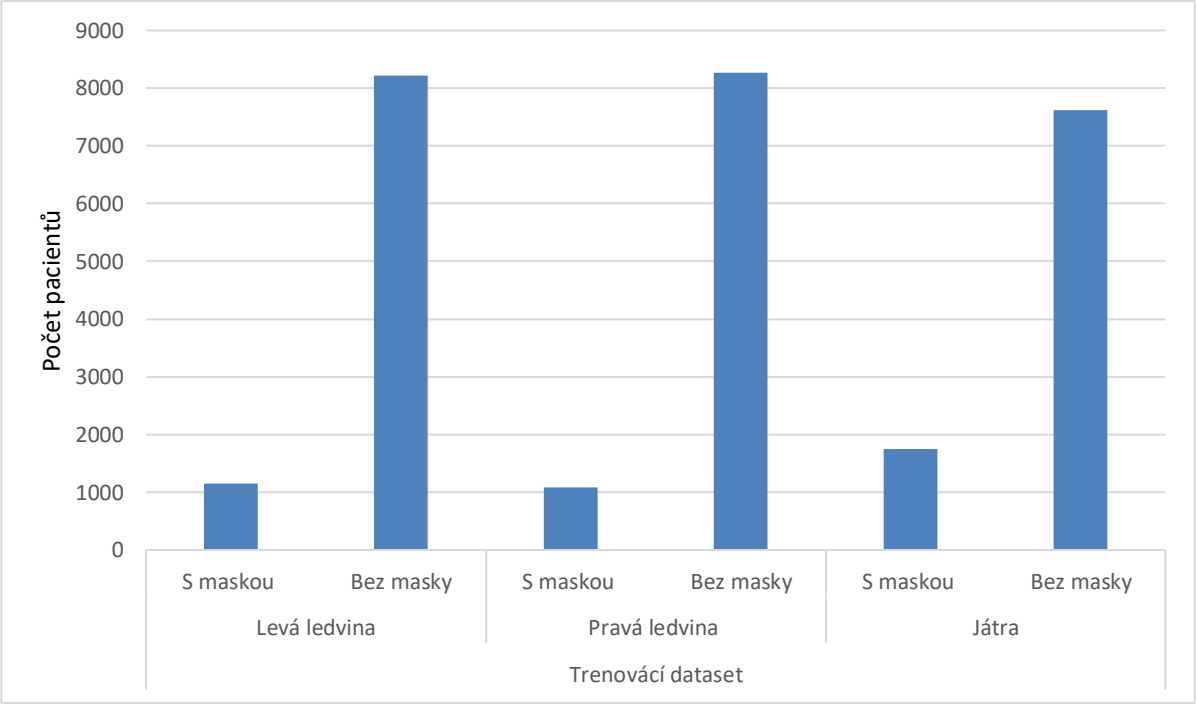
Obrázek 4.3: Počet 2D obrázků-řezů v 3 sadách.

Pacienti jsou unikátní a počet řezů v jejich datasetech se může lišit. Z obrázku 4.3 je patrné, že validační dataset obsahuje více řezů než testovací dataset. Nicméně pro finální vyhodnocení to není problém, protože náš cíl spočívá v segmentaci celého pacienta, nikoli jednotlivých řezů.

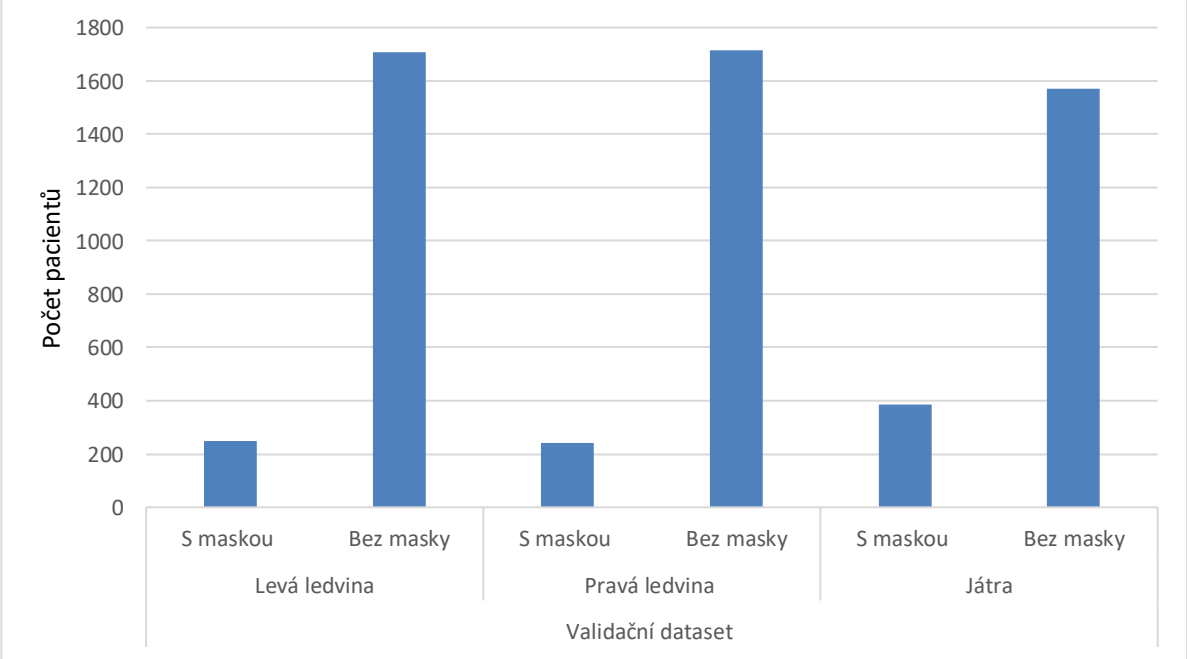
Během trénování a následného vyhodnocení jsme se setkali s problémem, že mnoho řezů v našem datasetu neobsahovalo požadované orgány. To způsobovalo, že neuronová síť často pracovala s prázdnými 2D objekty, které nebylo možné spojit do kontextu jednoho pacienta. To bylo způsobeno použitím detekční sítě Detectron2, která se specializuje na detekci objektů.

Pro lepší pochopení tohoto problému jsme zaznamenali konkrétní hodnoty masky pro každý orgán v rámci jednotlivých částí našeho datasetu. Tyto hodnoty jsou zobrazeny

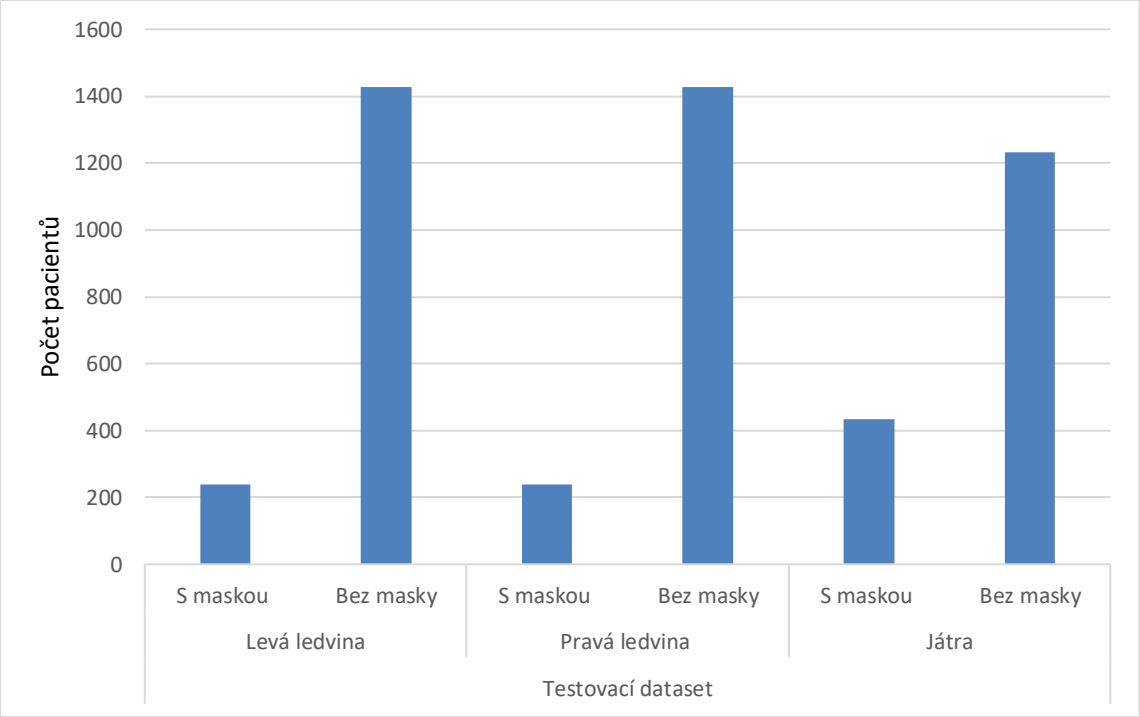
v obrázcích 4.4, 4.5 a 4.6.



Obrázek 4.4: Počet řezů obsahujících anotaci konkrétního orgánu oproti řezům, které ho v trénovacím datasetu neobsahují.



Obrázek 4.5: Počet řezů obsahujících anotaci konkrétního orgánu oproti řezům, které ho ve validačním datasetu neobsahují.



Obrázek 4.6: Počet řezů obsahujících anotaci konkrétního orgánu oproti řezům, které ho v testovacím datasetu neobsahují.

Pro analýzu závislosti mezi počtem řezů obsahujících daný orgán a celkovým počtem řezů byla vytvořena Tabulka 4.7. Uvedená tabulka prezentuje procentuální podíl řezů, ve kterých je přítomný konkrétní orgán, vzhledem k celkovému počtu řezů. Po provedení průměrování bylo zjištěno, že levá ledvina je obsažena průměrně v 13,09 % řezů, játra ve 21,48 % řezů a pravá ledvina ve 12,78 % řezů.

Existence masky v řezu								
Trénovací dataset			Testovací dataset			Validační dataset		
Levá ledvina	Pravá ledvina	Játra	Levá ledvina	Pravá ledvina	Játra	Levá ledvina	Pravá ledvina	Játra
12,31 %	11,63 %	18,68 %	14,29 %	14,34 %	25,99 %	12,68 %	12,37 %	19,79 %

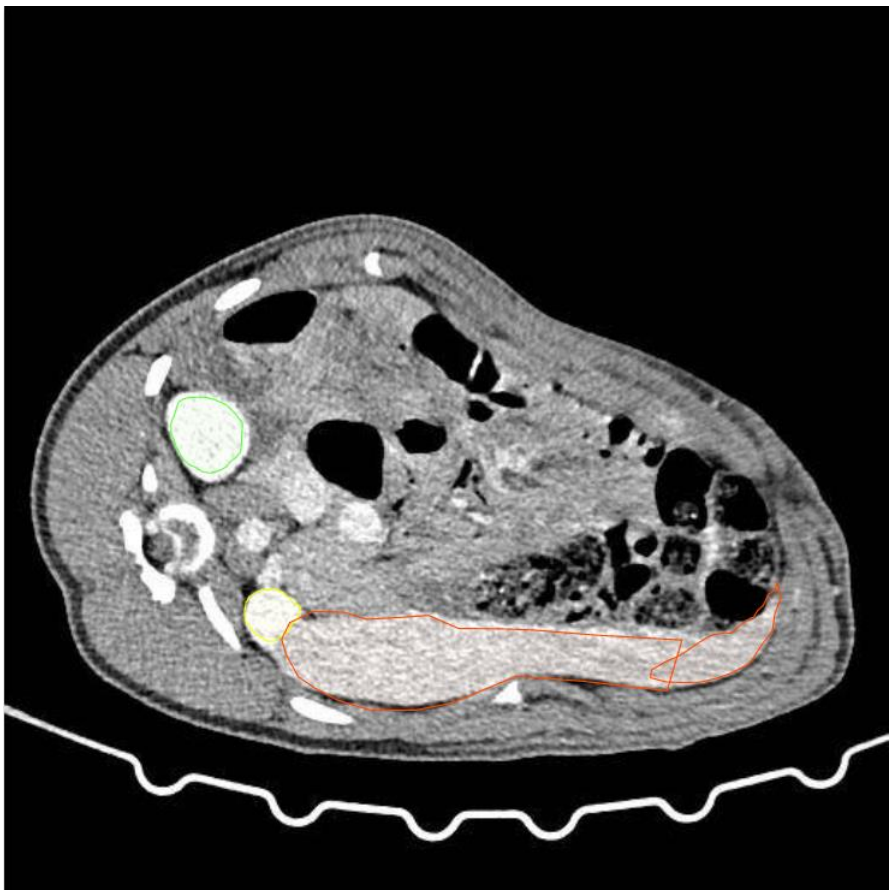
Tabulka 4.7: Procentuální podíl řezů obsahujících anotace orgánů v datasetu.

4.2 Příprava datasetu

Jak je popsáno v předchozí kapitole, náš dataset byl původně získán ve formátu DICOM přímo z tomografu. Pro správné trénování neuronové sítě bylo nezbytné provést předzpracování těchto snímků. Nejprve jsme museli převést 3D obrazy pacientů na soubor 2D dat. K tomuto účelu jsme využili knihovnu Io3d, která umožňuje práci s 3D obrazy v jazyce Python a obousměrnou konverzi mezi 2D a 3D objekty. S pomocí této knihovny

jsme byli schopni rozložit objemový objekt na postupné řezy, které jsme poté opět spojili a vytvořili tak původní objemový objekt. Tímto procesem jsme získali 14 souborů 2D obrázků.

Pro účely učení sítě bylo nezbytné vytvořit anotace. Od lékaře jsme získali anotaci pouze pro jednoho pacienta a ostatní pacienty jsme anotovali sami. Tento fakt přispěl ke zvýšení nepřesnosti výsledků. K anotacím jsme použili nástroj CVAT, který je na katedře kybernetiky FAV ZČU široce využíván pro anotace vizuálních obrázků. Každý pacient byl považován za samostatnou anotační úlohu a byl reprezentován v získaných 2D souborech. Hlavním cílem anotace bylo označení jater a pravé a levé ledviny, jak bylo uvedeno v kapitole 4.1. Informace o ostatních orgánech nebyly využity kvůli časové náročnosti vytváření takového datasetu, což také mělo vliv na finální výsledky. Příklad anotace v nástroji CVAT je zobrazen na obrázku 4.8. Pro tento příklad byl vybrán řez, na kterém jsou anotovány všechny tři orgány.

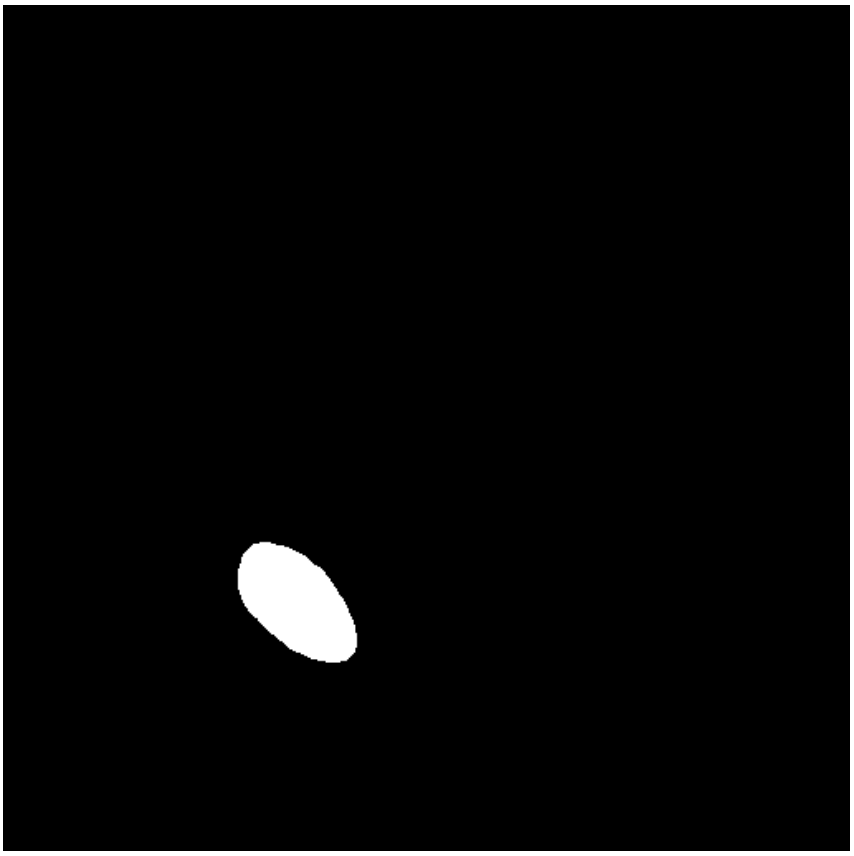


Obrázek 4.8: Anotace ledvin a jater na jednom řezu.

Na přípravě datasetu se podíleli studenti katedry kybernetiky FAV ZČU, kteří poskytli anotace namísto lékaře. Přímo na tomto obrázku lze vidět, že informace od studentů nejsou ideální. Nicméně vzhledem k tomu, že hlavním cílem je zkoumání pozitivních nebo negativních tendencí využití detekční sítě, nepředstavuje to pro nás velký problém. Při vyhodnocení výsledné segmentace však může nastat problém, kdy nám taková kvalita anotací od studentů jako učitelů může v testovacím datasetu zkreslit porovnání, které provádíme na základě přesnosti v pixelech.

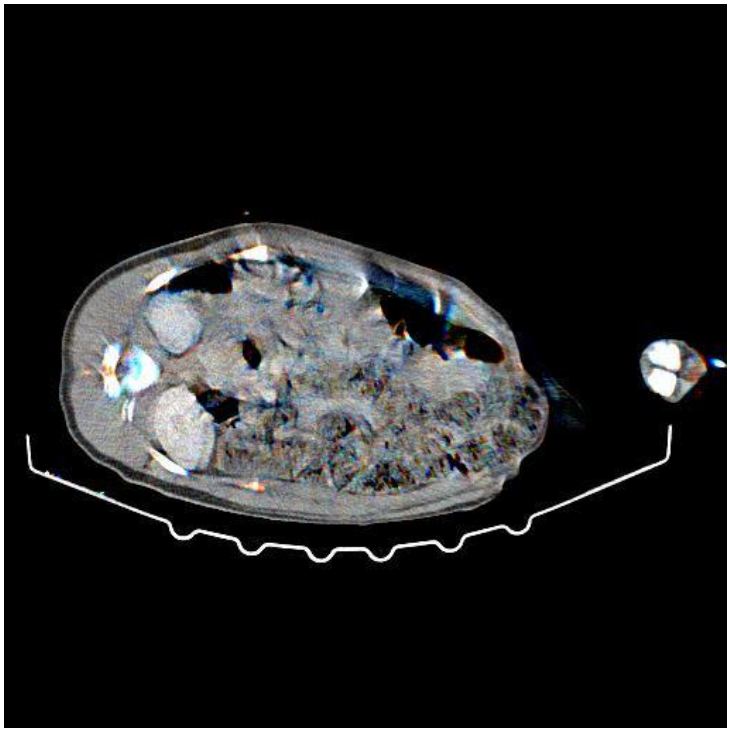
Po anotování všech pacientů jsme jejich anotace sloučili do jednoho projektu. Díky tomu jsme mohli mít všechny anotace uloženy v jednom JSON souboru. Každý pacient byl identifikován svým názvem ve formátu Tx***Cislo***D_Ven, kde ***Cislo*** je číslo pacienta, kterému odpovídá maska, a Ven je označení pro kontrastní žilní fázi tomogramu, kterou jsme využívali během tohoto experimentu. Pro správné trénování a vyhodnocení jsme však potřebovali rozdělit soubor na trénovací, validační a testovací část. K tomuto účelu jsme vytvořili Python skript, který na základě názvu pacienta a názvu vyžadovaného pro detekci orgánu odděloval informace z původního JSON souboru do nových souborů, čímž jsme vytvářeli potřebné datasety.

Další výhodou využití nástroje CVAT byla možnost stahovat příslušné PNG soubory se snímky, které byly roztříděny podle struktury odkazované v JSON souboru. Kromě toho jsme také vytvořili PNG masky pro všechny tři datasety, abychom měli vhodnou reprezentaci informací od studentů. Pro vyhodnocení finálních výsledků jsme využili PNG masek testovacího datasetu a pomocí metody Intersection over Union jsme je porovnali s výsledky segmentace neuronové sítě. Rozměr těchto maskových obrázků odpovídal obrázkům použitým pro anotace, konkrétně 512 x 512 pixelů. Pro každého pacienta jsme vytvořili masky zvlášť pro každý orgán. V případě, že řez obsahoval jeden orgán, přiřadili jsme hodnotu pixelu jeho anotaci 1 a pozadí 0. Tento postup jsme opakovali pro každý řez a orgán. Finální výstup naší segmentace obsahoval masku ve 2D prostoru, kterou jsme museli normalizovat na 1. Příklad takového formátu interpretace masky je zobrazen na obrázku 4.9.



Obrázek 4.9: Maska levé ledviny.

Jak již bylo zmíněno v této kapitole, při trénování frameworku Detektron2 jsme se setkali s problémem v podobě nedostatku informací, který může způsobit, že síť nebude správně rozlišovat mezi jednotlivými orgány. Abychom tento problém minimalizovali, rozhodli jsme se přidat kontext k jednotlivým řezům, které posíláme jako vstup do neuronové sítě. V rámci počítačové RGB reprezentace 2D snímku máme k dispozici tři barevné kanály. Tuto možnost využíváme k přidání obrázku s širším kontextem na vstup neuronové sítě. Aktuální řez jsme uložili do zeleného barevného kanálu. Řez před ním byl uložen do červeného kanálu a následující řez do modrého kanálu. V případě, že se jedná o první nebo poslední řez pacienta, který nebyl k dispozici pro kontext, byl příslušný kanál ponechán prázdný. Díky shodnému pojmenování řezů a stejné struktuře souborů jako u původních 2D snímků můžeme odkazovat přímo na nové 2D snímky s RGB kontextem pomocí JSON souboru z masky. Příklad takového 2D obrázku s RGB kontextem je zobrazen na obrázku 4.10.



Obrázek 4.10: Řez s kontextem předchozího a následného řezů.

4.3 Experimenty

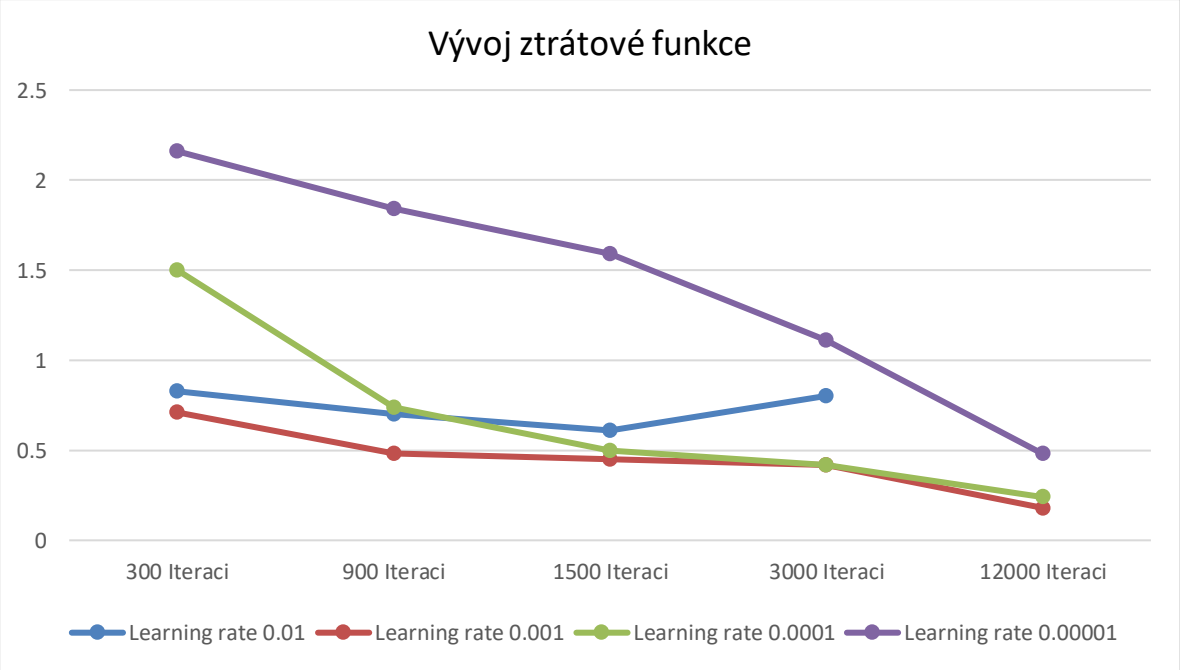
Po dokončení přípravy datasetu jsme postupně zahájili trénování naší neuronové sítě. Kvůli významnému rozsahu paměti a výpočetním nárokům, které neuronová síť vyžaduje, jsme se rozhodli využít výpočetní infrastrukturu poskytovanou organizací MetaCentrum. Tato organizace nabízí akademickému prostředí v České republice přístup ke vzdálenému výpočetnímu a úložnému systému pro potřeby akademického výzkumu. Před samotným spuštěním trénování jsme nahráli náš připravený dataset a příslušný kód

na úložiště poskytované MetaCentrem. Všechny použité skripty a implementace, které jsme vytvořily v rámci této bakalářské práce, jsou veřejně dostupné na platformě GitHub. URL adresa kódového repozitáře je uvedena na konci naší práce. Díky této strategii jsme získaly přístup k dostatečným výpočetním kapacitám a umožnily efektivní trénování naší neuronové sítě, což je zásadní pro dosažení požadovaných výsledků a splnění cílů naší studie.

Pro sledování učení neuronové sítě jsme zvolili systematický přístup místo náhodného nastavení learning rate (LR) a počtu iterací. Na začátku jsme zvolili nejvyšší hodnotu LR, konkrétně 0,01, a nejmenší počet iterací, který jsme nastavili na 300. Postupně jsme poté měnili hodnotu LR podle vzorce $LR \times 10^{-1}$. Když jsme dosáhli hodnoty LR 0,00001, považovali jsme ji za minimální rozumnou a pokračovali jsme v měnění počtu iterací, přičemž jsme opět začínali s LR hodnotou 0,01.

Tento iterativní postup jsme opakovali pro různé počty iterací, konkrétně 300, 900, 1 500, 3 000. Pro počet iterací 12 000 jsme začali s LR hodnotou 0,001, protože předchozí výsledky s LR hodnotou 0,01 ukázaly, že pro naši úlohu je tato hodnota příliš vysoká jak z našeho pohledu, tak i z pohledu neuronové sítě.

Na obrázku 4.11, který je přiložen, můžeme sledovat, jak naše neuronová síť vnímá a hodnotí průběh trénování. Tento obrázek poskytuje přehled o vývoji LR a počtu iterací během trénování sítě.

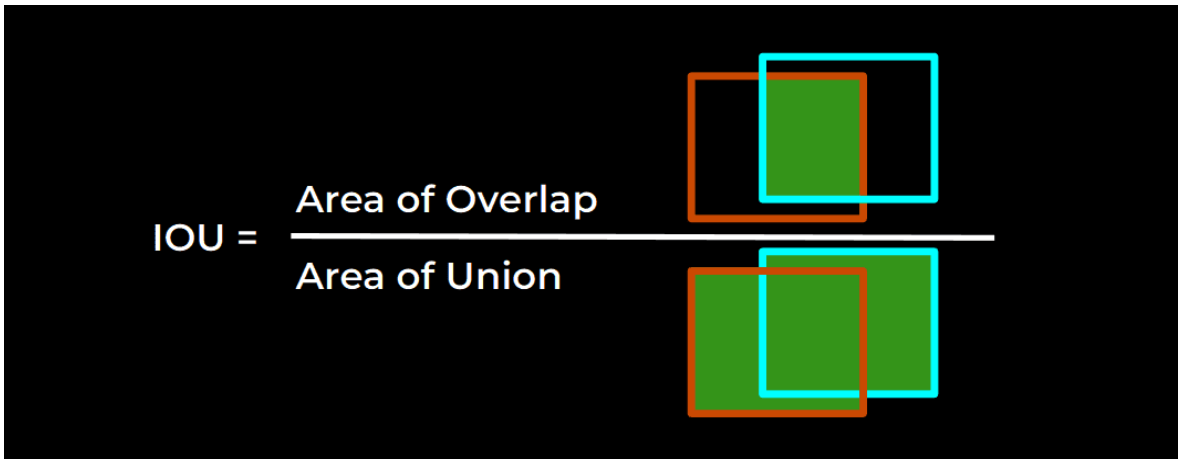


Obrázek 4.11: Vývoj ztrátové funkce (osa y) v Detectron2 během trénování s různými nastavení learning rate (popisuje barvy) a počtem iteraci (osa x).

Na základě obrázku 4.10 je patrné, že neuronová síť hodnotí pozitivní dynamiku ve svém učení v závislosti na počtu iterací, a to s výjimkou případu LP 0,01. Nicméně toto ocenění není dostatečným měřítkem kvality, a proto jsme se rozhodli hodnotit výsledky pomocí efektivní metriky. Pro tuto metriku jsme zvolili klasickou metodu Intersection Over

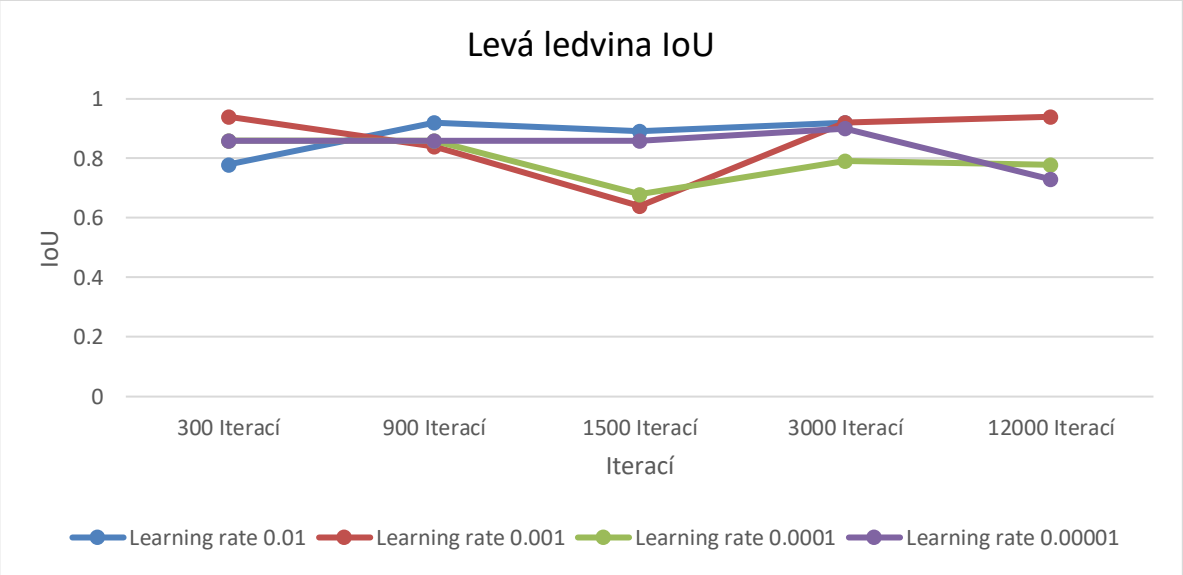
Union (IoU), která porovnává podobnost dvou obrázků. Pro správnou implementaci této metody jsme normalizovali informace od učitele, konkrétně masky, na hodnotu 1, jak jsme popsali v kapitole 4.2. Jako výstup neuronové sítě jsme uložili segmentační masky získané aplikací natrénované neuronové sítě na testovací dataset. Výsledkem jsou masky řezů, které jsme také normalizovali na hodnotu 1, podobně jako původní segmentace od učitele. Je důležité poznamenat, že soubory jsou tříděny nejen podle případů trénování, ale také podle orgánů, které jsou v maskách uvedeny.

Princip metriky Intersection over Union (IoU) je odvozen z jejího názvu. IoU se vypočítává jako podíl průniku dvou masek a jejich sjednocení. Myšlenkovým předpokladem je, že pokud jsou dvě masky identické, jejich průnik je stejný jako jejich sjednocení, což znamená, že mají stejný počet pixelů. V takovém případě je IoU rovno 1, což je ideální výsledek pro segmentaci, kterého se snažíme dosáhnout. Na druhou stranu, pokud jsou dvě masky zcela odlišné, jejich průnik je 0, a tedy hodnota IoU je také 0. Celkově se IoU pohybuje v rozmezí od 0 do 1, přičemž vyšší hodnota značí lepší shodu mezi predikovanou maskou a referenční maskou od učitele. Matematický zápis pro IoU je: (Průnik masek) / (Sjednocení masek), kde průnik masek představuje počet pixelů, které jsou společné oběma maskám, a sjednocení masek zahrnuje všechny pixely, které jsou obsaženy alespoň v jedné z masek. Obrázek 4.12 vizualizuje fungování IoU a způsob jeho výpočtu.

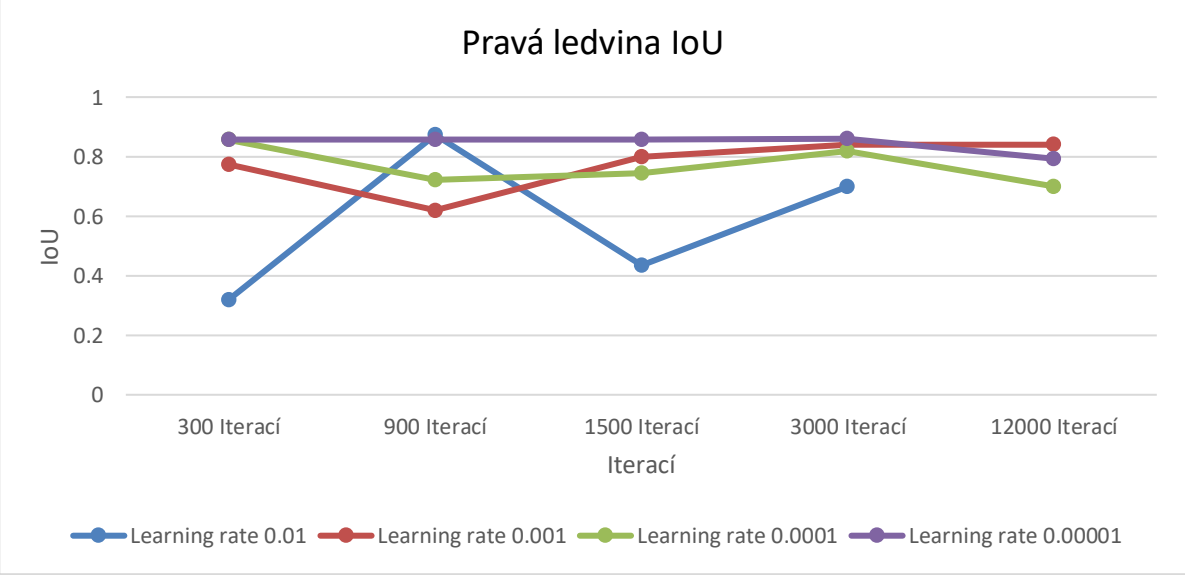


Obrázek 4.12: Princip Intersection over Union.

Výše popsaným způsobem jsme provedli hodnocení našich výsledků. Obrázky 4.13, 4.14, 4.15 nám ukazují hodnoty IoU podle nastavení sítě.

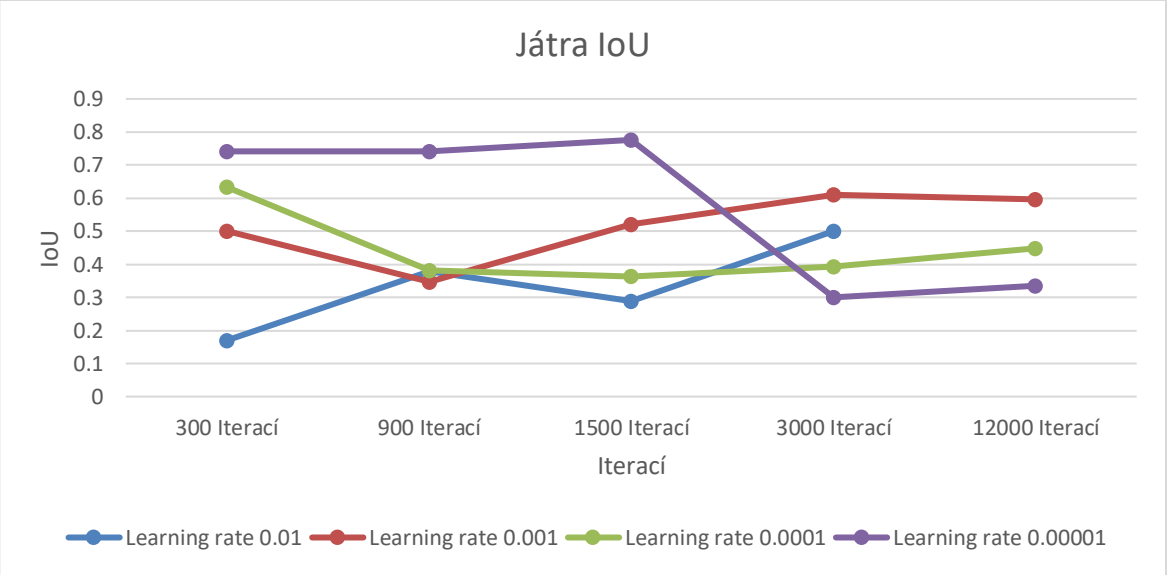


Obrázek 4.13: Dynamika IoU pro **levou ledvinu**.



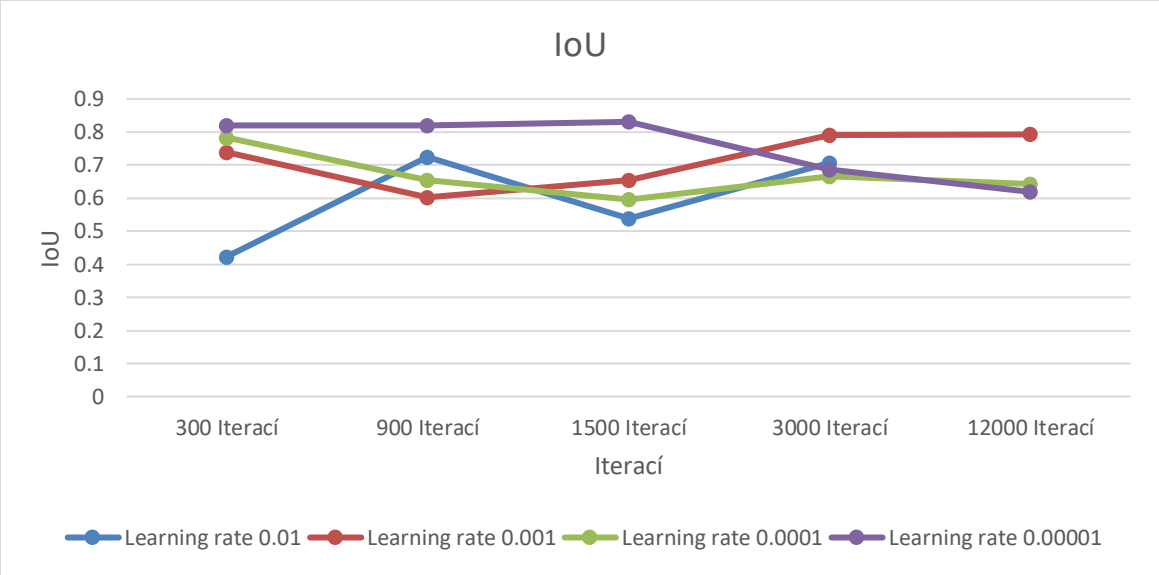
S

Obrázek 4.14: Dynamika IoU pro **pravou ledvinu**.



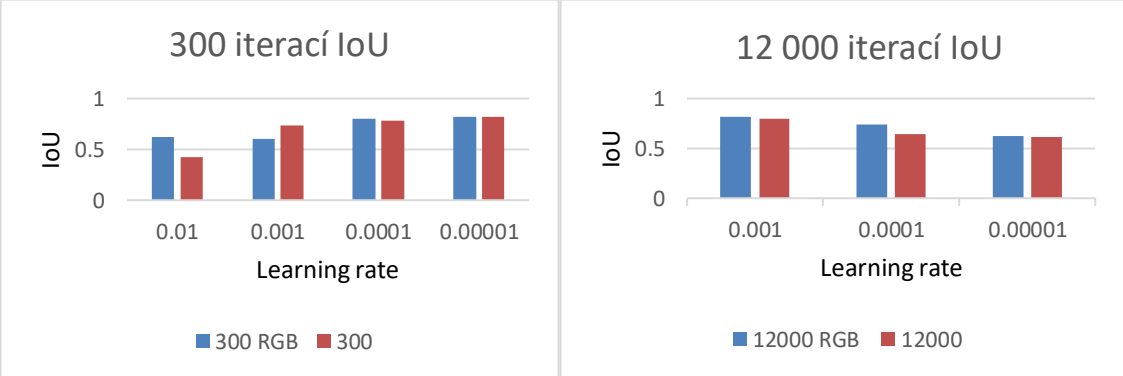
Obrázek 4.15: Dynamika IoU pro **játra**.

Na základě hodnot IoU jsme provedli analýzu výsledků. Zjistili jsme, že pro všechny tři orgány je nejlepším nastavením LR hodnota 0,01. Z grafu je nicméně patrné, že pro nastavení LR na hodnotu 0,00001 je v případě 1500 iterací dosaženo lepších výsledků. Je však důležité si uvědomit, že náš dataset obsahuje mnoho řezů bez přítomnosti zkoumaných orgánů, jak jsme zmínili v kapitole 4.1. V případě nastavení LR na hodnotu 0,00001 vrací neuronová síť při 300 a 900 iteracích prázdné masky. Tento jev je způsoben tím, že poměr řezů obsahujících zkoumané orgány vůči prázdným řezům je velmi nízký, jak je vidět v tabulce 4.7. Pro nastavení LR na hodnotu 0,00001 při 1500 iteracích se síť začíná zaměřovat pouze na játra, ale opět na velmi malý počet řezů (konkrétně 101 řezů s detekovaným orgánem oproti požadovaným 420). Tato situace vede k vysoké hodnotě IoU, která je ovlivněna prostorem pacientova těla, který neobsahuje játra. Z tohoto důvodu nemůžeme považovat tyto výsledky za nejlepší. Pro srovnání výsledků ze tří orgánů jsme provedli průměr hodnocení IoU. Tento průměr je zobrazen na obrázku 4.16. Na základě tohoto obrázku jsme došli k závěru, že optimální hodnotou LR je 0,001. Tato hodnota zohledňuje výsledky všech tří orgánů a je nejlepším kompromisem mezi jejich detekcí a počtem správně identifikovaných řezů.



Obrázek 4.16: Dynamika IoU zprůměrovaná podle IoU ledvin a jater.

Před dalším experimentem s větším počtem iterací pro nastavení LR na hodnotu 0,001 jsme provedli porovnání výsledků s využitím RGB kontextu. Cílem tohoto porovnání bylo zjistit, zda přidání RGB informace vylepší výsledky segmentace. Porovnání jsme provedli na všech hodnotách LR, neboť RGB kontext nám poskytuje třikrát více informací a je důležité zjistit, jak se trénování projeví při různých hodnotách LR. Tato další sada experimentů je však časově náročná. Z tohoto důvodu jsme nejprve provedli porovnání při 300 iteracích a následně při 12 000 iteracích. Tím jsme získali předběžné výsledky, které nám pomohou rozhodnout, zda je vhodné provádět další trénování s vyšším počtem iterací. Výsledky tohoto porovnání jsou zobrazeny v obrázku 4.17.

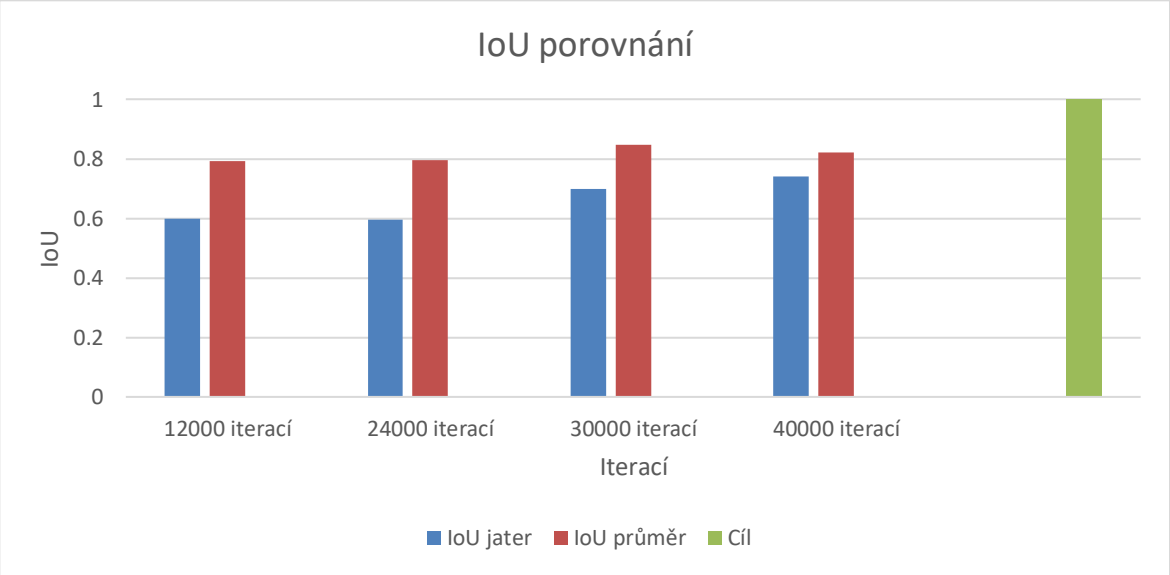


Obrázek 4.17: Porovnání výsledků segmentace při různých hodnotách LR a počtech iterací s rozšířeným kontextovým kanálem (RGB).

Obrázek 4.17 potvrzuje naši hypotézu, že přidání RGB kontextu vylepšuje schopnost detekce naší sítě. Je zřejmé, že hodnoty IoU v neuronové síti s RGB kontextem jsou proporcionální hodnotám bez něj. Přestože trénování s RGB kontextem vyžaduje o 10 % více časových prostředků, provedli jsme další experimenty bez využití kontextu. Tím jsme získali informaci o vývoji IoU pouze na základě vlastního objektu zájmu. S ohledem na

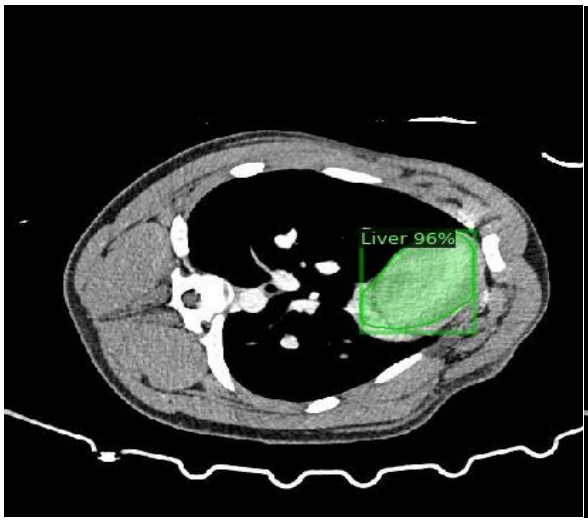
podobnou dynamiku vývoje IoU jsme se pro finální nastavení neuronové sítě rozhodli použít trénování s RGB kontextem.

Další část našeho experimentu spočívala v aplikaci vyššího počtu iterací trénování na LP 0,001. Kvůli časovým omezením MetaCentra jsme však postupně zvyšovali počet iterací na hodnoty 24 000, 30 000 a 40 000. Celkový průběh programu při 40 000 iteracích trénování se blížil maximální dovolené době 24 hodin, a proto jsme tuto hodnotu považovali za nejvyšší možný počet iterací na LP 0,001. Výsledky těchto experimentů jsou prezentovány v obrázku 4.18.

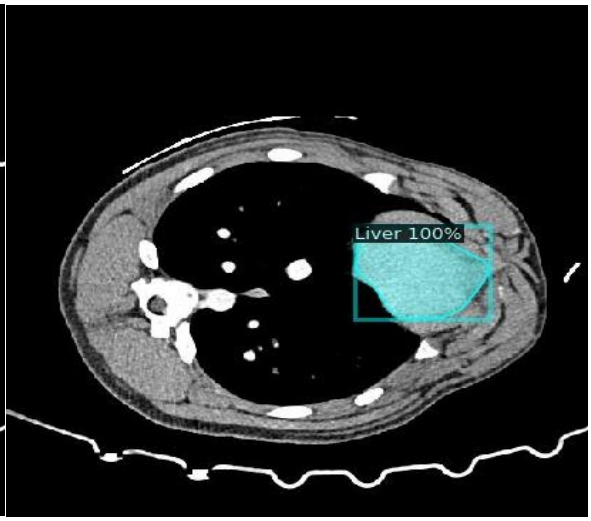


Obrázek 4.18: Výsledné IoU při 12 000, 24 000, 30 000, 40 000 iteracích pro játra a průměr orgánů.
Oba od 0.

Jaterní tkáně jsou klíčovým orgánem pro naše výzkumné úsilí. V souvislosti s vývojem metriky Intersection over Union (IoU) pro segmentaci jater pomocí maskování jsme zjistili, že nejlepší výsledky poskytuje nastavení s 40 000 iteracemi. Nicméně hodnota IoU = 0,73999 není příznivá pro aplikace v oblasti medicíny, protože se zde vyskytuje značné množství chyb. Hlavním nedostatkem je nesprávné rozpoznávání jater a srdce. Tato situace je způsobena absencí anotací pro srdce a omezeným tréninkem v 2D prostoru. Tento jev ukazuje na omezenou schopnost detekční sítě rozlišovat 2D objekty s vyšší mírou podobnosti. Pro ilustraci jsme připojili obrázky 4.19 a 4.20, na kterých je zřejmé, že i pro nás je bez kontextu celého pacienta obtížné rozlišit srdce a játra.

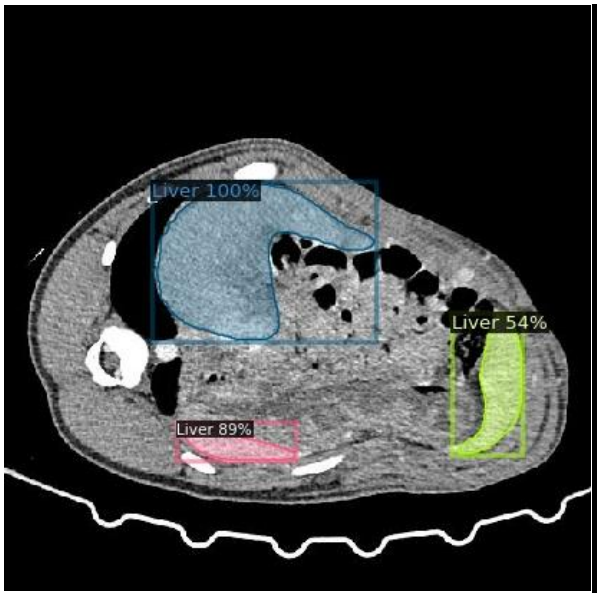


Obrázek 4.19: Srdce
špatný label do třídy ‚játra'.



Obrázek 4.20: Játra
správný label do třídy ‚játra'.

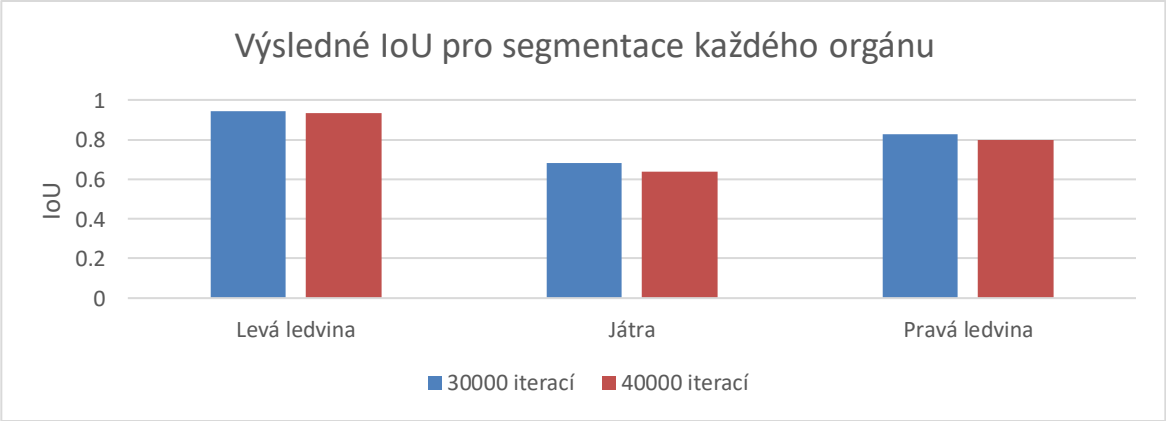
Avšak odborníci dokážou rozlišit řez srdce a řez jater i bez kontextuálních informací o pacientovi. Výsledky naší studie naznačují, že přidání anotace srdce by mohlo vylepšit výsledky. To by nám otevřelo prostor pro další výzkum. Nicméně je stále patrná nedokonalost detekce u každého orgánu samostatně. Na obrázku 4.20 je například vidět velká část orgánu, která je dobře odlišitelná od pozadí, ale při detekci není správně identifikována jako játra. I když jsme schopni poskytnout pouze omezenou míru expertní anotace, protože lékaři nemohou anotovat všechny pacienty v této fázi výzkumu, velikost trénovacího datasetu by měla kompenzovat nedokonalosti anotace. Z toho plyne, že denzita jater ve srovnání se svalovinou kolem ní není v tomto případě natolik výrazná, aby byla neuronovou sítí považována za prioritní příznak pro detekci. Dále je třeba poznamenat, že kvůli výraznému rozdílu ve formě mezi řezy nemůže síť pouze vyhledávat objekty se stejnou formou. To ukazuje obrázek 4.21, kde byla provedena detekce s přesností IoU = 0,7898. Obrázek 4.20 dosáhl přesnosti IoU = 0,8226.



Obrázek 4.21: Výsledná segmentace jater pro 1 řez.

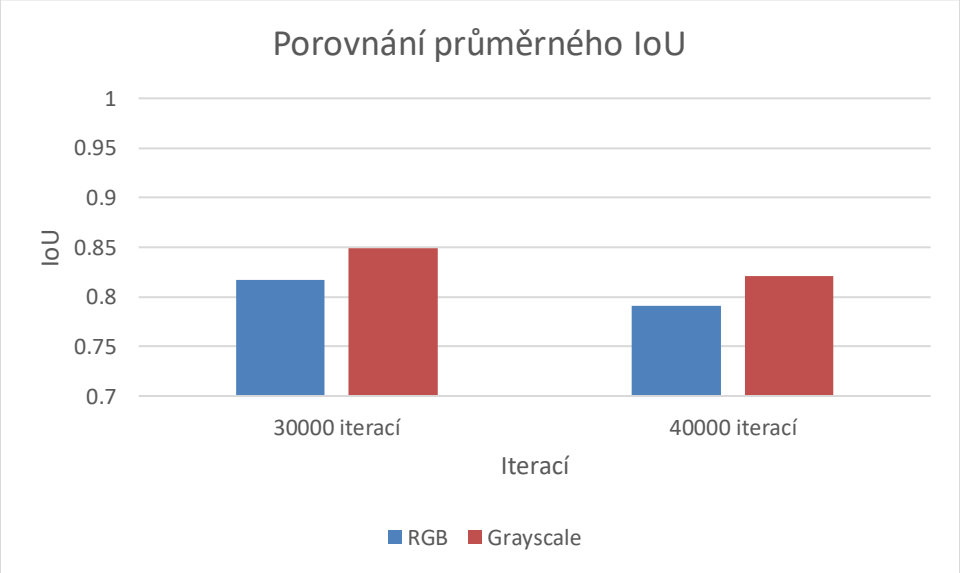
Obrázek 4.22: Výsledná maska pro řez na obrázku 4.21.

Je zřejmé, že existuje významná odlišnost ve formě mezi tímto obrázkem a obrázkem 4.20. Tato odlišnost také ovlivňuje chybu detekce v 2D prostoru, protože, jak jsme již zmínili v kapitole 4.1, detekční síť nedokáže rozpoznat, že se jedná o jeden orgán pacienta. Obrázek 4.22 nám ukazuje, jak by měla vypadat maska pro tento řez. Poslední experiment, který jsme provedli s použitím nástroje Detectron2, zahrnoval aplikaci kontextového datasetu při konfiguraci s nejlepším IoU pro játra samostatně a celkově. Jak jsme již zmínili v předchozí kapitole, tyto konfigurace dosáhly IoU hodnot 0,001 při trénování s 40 000 a 30 000 iteracemi. Nejprve porovnáme tyto dvě konfigurace, abychom mohli zhodnotit vliv tříobrazového kontextu na proces trénování. Tato porovnání nám poskytnou informace o tom, zda další experimenty s větším počtem iterací trénování povedou ke zlepšení výsledků. V případě průměrného IoU dosáhla konfigurace s 30 000 iteracemi nejlepších výsledků. Porovnání IoU detekce konkrétních orgánů pro náš experiment s kontextovým datasetem je zobrazeno na obrázku 4.23.



Obrázek 4.23: Výsledné IoU pro segmentace každého orgánu (levá a pravá ledvina a játra) po 30 000 a 40 000 iteracích.

Z obrázku 4.23 je patrné, že po 30 000 iteracích trénování již nelze výsledek dále zlepšovat zvýšením počtu iterací. V porovnání s trénováním bez kontextu se v případě jater výsledek IoU během 40 000 iterací zhoršil. Játra jsou pro nás nejdůležitějším objektem detekce, proto můžeme jednoznačně říct, že další experimenty s nástrojem Detectron2 nepřinesou vylepšení našich výsledků. Na obrázku 4.24 jsme zobrazili rozdíl v průměrném IoU mezi trénováním bez kontextu a trénováním s kontextem.



Obrázek 4.24: Porovnání průměrného IoU mezi trénováním s kontextem předchozího a následujícího řezu a bez kontextu při 30 000 a 40 000 iteracích.

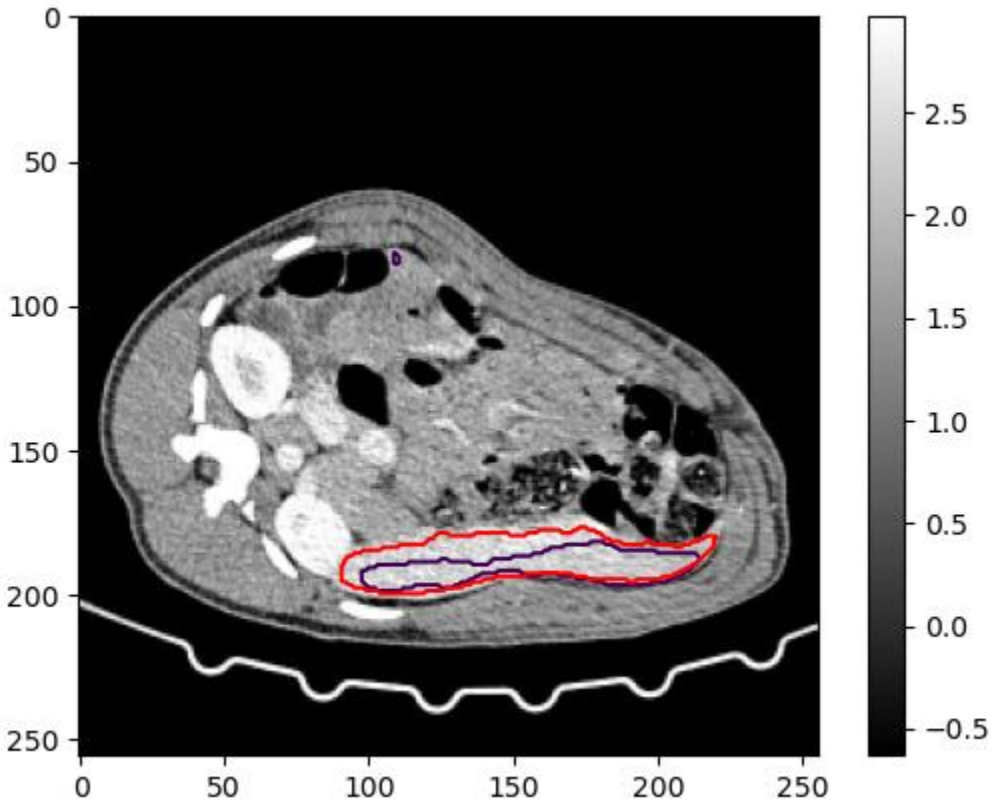
Z tohoto obrázku vidíme neočekávané výsledky. Výsledky nás vedou k myšlence, že přidání kontextu sítí nepřináší lepší výsledky než bez něj. To naznačuje, že tak malý kontext není dostatečný pro dosažení vhodných výsledků pro praktické využití sítě. Horší hodnocení IoU také naznačuje, že k přetrénování sítě dochází dochází dříve než při trénování bez kontextu, protože experimenty s nižším počtem iterací ukazují, že trénování s kontextem by mělo vést k lepším výsledkům. Podle teoretického předpokladu by také mělo dojít k lepším výsledným maskám. Nicméně vidíme, že rychlost zhoršování IoU mezi 30 000 a 40 000 iteracemi není výrazná, a proto další experimenty nejsou pravděpodobně zásadní pro výrazné zlepšení našeho nejlepšího výsledku IoU = 0,8489.

4.4 Porovnání s U-NET

Pro porovnání jsme vybrali segmentační neuronovou síť U-NET. Ta je jednou z nejvíce využívaných architektur pro segmentaci obrazových dat v oblasti medicíny. Pro maximálně konzistentní porovnání jsme využili variantu, kde vstup i výstup jsou 2D datasety. Z toho důvodu jsme využili stejné trénovací, validační a testovací datasety. Pro trénování U-NET jsme bohužel museli použít Google Colaboratory, která má řadu omezení,

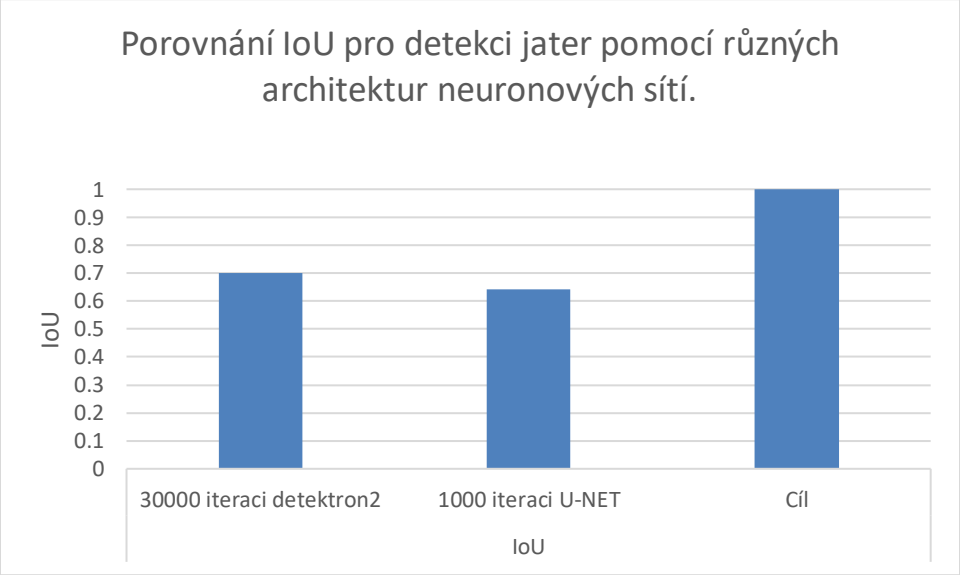
která způsobují komplikace. Hlavní komplikací je paměťové omezení. Umožňuje využít výpočetní zdroj s RAM o velikosti 12 GB, což není málo, proto jsme museli rozdělit dataset na pacienty a při každé iteraci trénování posílat na vstup sítě jednoho z pacientů. Toto způsobuje nerovnoměrné učení z hlediska celého trénovacího datasetu. Stejně však je přidání náhodně se měnících pacientů z datasetu jednou z užívaných možností v případě trénování na medicínských datech. Rovněž omezení náhodnosti pouze na výběr pacienta oproti náhodnému výběru obrázků nám pomáhá tím, že si budeme jistý, že při každé iteraci bude vstup obsahovat řezy s maskami. Také jsme měli omezení na 1 spuštení trvající 12 hodin, což při výpočetní kapacitě zdrojů dávalo málo možností pro experimenty. Proto, jsme mohli dosáhnout počtu iterací 1000. Taky jsme měli celkové omezení 30 hodin za týden, což také omezilo počet našich experimentů.

Nejlepšího výsledků jsme dosáhli při iteraci 1000, což jsme mohli předpokládat kvůli tomu, že při tak malém počtu iterací by nemělo dojít k přetrénování sítě. Při sestavování modelu jsme využili optimizer SGD, který je defaultním optimizerem pro architekturu detektron2. V případě U-NET jsme optimizeru povolili samostatně zvolit learning rate, což nám dalo nejlepší výsledky. Na základě výpisu IoU po jednotlivých iteracích trénování byla dynamika trénování na learning rate = 0,001 horší,. Výsledná IoU = 0,641, což je velmi dobrý výsledek pro počet iterací 1000. Segmentaci na jednom z řezů můžeme vidět na obrázku 4.25.



Obrázek 4.25: Příklad segmentace jater v řezu z testovacího datasetu v porovnání se segmentaci od učitele po 1000 iteracích. Fialovou barvou je označena segmentace od neuronové sítě, červenou barvou je označena segmentace od učitele.

Porovnání IoU pro játra, která jsou naším cílovým orgánem, je zobrazeno na obrázku 4.26.



Obrázek 4.26: Porovnání IoU pro detekci jater pomocí neuronových sítí založených na architektuře detectron2 a na architektuře U-NET mezi sebou a s cílovou IoU.

Můžeme vidět, že náš výsledný IoU je poněkud horší, než jsme dosáhli při trénování detectron2, ale dosáhli jsme ho na značně menším počtu iteraci. Proto můžeme říct, že segmentační síť s architekturou U-NET vykazuje pro 2D dataset lepší dynamiku trénování. Může to být způsobeno tím, že U-NET bere v potaz pozici objektů uvnitř obrázku, zatímco detectron2 jako detekční síť předpokládá, že objekt může být kdekoliv na obrázku. Rovněž věnuje méně pozornosti samotné segmentaci, protože jeho prioritou je detekce třídy objektu. Proto můžeme říci, že detekční síť s architekturou detectron2 není nejvhodnější volbou při úloze segmentace orgánu.

5 Závěr

V naší práce jsme se věnovali segmentaci parenchymatózních orgánů na CT snímcích. V teoretické části jsme věnovali pozornost metodám počítačového vidění, které jsou vhodné pro segmentaci objemových dat z výpočetní tomografie. Zaměřili jsme se na neuronové sítě s různou architekturou, které jsme následně využili v experimentální části. Velkou částí práce bylo vytváření datasetu, zejména segmentací využitých jako informace od učitele.

Porovnání jsme prováděli na segmentaci jater. Při detekci jater se setkáváme s komplikacemi způsobenými dvěma faktory. Prvním faktorem je umístění jater v těle pacienta. Velká část jater se nachází v blízkosti žebrových oblouků, které mají podobnou

denzitu jako játra. Pro síť to znamená menší kontrastnost, což ovlivňuje vážení tohoto příznaku, a výsledkem je nadměrná detekce jater u pacienta. Druhým faktorem je složitost a proměnlivost tvaru jater ve 2D kontextu. Každý řez je sítí považován za jednoznačný kontext a maximální dosažitelnou hloubkou je tříobrazový kontext vzhledem k počtu dostupných barevných kanálů. Lze taky jasně vidět rozdíl, který jsme již zmínili. Pro detektron2 jsou to dva různé objekty a z této informace vyplývá, že játra mohou mít různý tvar. Síť je vnímá jako různé případy, nikoli jako různé části stejného objektu. Proto není schopna označit konkrétní tvar jako správný. Síť také považuje tyto tvary za tři různé objekty a snaží se vnímat jejich kontextový tvar ve vstupních řezech zvlášť. Experiment s detekční sítí detectron2 nám ukázal, že detekční síť nedosahuje dokonalých výsledků a funguje hůře než U-NET.

Naopak detekce ledvin je dosahuje dobrých výsledků. Je patrné, jak se v případě 30 000 iterací zlepšuje průměrné IoU pro ledviny. Problémem jsou však případy nalezení objektů podobného tvaru u pacientů, zejména kostí končetin. Bohužel tvar a denzita jater po řezech nejsou dostatečně unikátní v kontextu celého pacienta, což vede k takovým chybovým detekcím. Nejednotná poloha každého pacienta během snímání CT snímků v tomografu způsobuje, že pozice orgánů ve 2D obraze není spolehlivým znakem pro síť.

Experiment se segmentační sítí U-NET nám ukazuje, že neuronová síť s touto architekturou je pro segmentační úlohy vhodnější než detekční síť s architekturou detectron2. Bohužel jsme museli využít jiný zdroj, což zmenšilo výpočetní kapacitu a omezilo experiment. Nicméně dynamika trénování potvrzuje medicínské zaměření této architektury a je postačující pro evaluační porovnání s výsledkem detectronů2

Jako výsledek všeho již napsaného by tato práce měla ukazovat, že při detekci 3D objektů není vhodnou volbou detekční síť zaměřená na vnímání objektů ve 2D prostoru, neboť 3D kontext je klíčovým zdrojem informací pro správnou detekci. V případě orgánů s podobnou řezovou strukturou není možné, aby 2D detekční síť správně odlišila složitější orgány, jako jsou játra, která byla preferovaným cílem experimentů. Naopak ledviny, které mají dostatečně strukturálně jednotné vlastnosti, dosáhly uspokojivých výsledků evaluace, ačkoli se stále vyskytovaly chyby v detekci případů, které nelze přičítat nedostatečné kvalitě datového souboru. Proto je v rámci dalšího vývoje v této oblasti vhodné zaměřit se na trénování detekční sítě na 3D datasetu.

Jedním ze způsobů, jak dále zlepšit naše výsledky je zvýšení počtu popsaných orgánů na vstupu, což by mohlo poskytnout dodatečné informace pro detectron2. Tím bychom mohli řešit problém, kdy je srdce považováno za játra nebo kdy jsou kosti zaměňovány za ledviny. Nicméně tento přístup vyžaduje rozsáhlou anotaci dat, což zvyšuje jak časové, tak finanční nároky na projekt. Alternativní cestou pro další výzkum je využití sítě pracující s vnímáním 3D kontextu.

SEZNAM POUŽITÉ LITERATURY:

[1] Brooks, S. L. (1993). Computed tomography. Dental Clinics of North America, 37(4), 575-590.

[2] Buzug, T. M. (2011). Computed tomography (pp. 311-342). Springer Berlin Heidelberg.

[3] ČIHÁK, R. (2002). Anatomie 2. GRIM. M.

[4] VLACH, Jaroslav. Strojové zpracování obrazu: Využití neuronových sítí I [online]. Matematicko-fyzikální fakulta Univerzity Karlovy., 20. ledna 2021 [cit. 2023-06-19]. Dostupné z: https://www.matfyz.cz/clanky/strojove-zpracovani-obrazu-vyuziti-neuronovych-siti-i

[5] Hinton, G. E., Krizhevsky, A., & Sutskever, I. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25(1106-1114), 1.

[6] O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. arXiv preprint arXiv:1511.08458.

[7] Cunningham, P., Cord, M., & Delany, S. J. (2008). Supervised learning. Machine learning techniques for multimedia: case studies on organization and retrieval, 21-49.

[8] Haykin, S. (1998). Neural networks: a comprehensive foundation. Prentice Hall PTR [9] Becker, S. (1991). Unsupervised Learning Procedures for Neural Networks. Int. J. Neural Syst., 2, 17-33.

[9] Becker, S. (1991). Unsupervised learning procedures for neural networks. International Journal of Neural Systems, 2(01n02), 17-33.

[10] Girshick, R. (2015). Fast r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 1440-1448).

[11] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).

[12] Felzenszwalb, P. F., Girshick, R. B., & McAllester, D. (2010, June). Cascade object detection with deformable part models. In 2010 IEEE Computer society conference on computer vision and pattern recognition (pp. 2241-2248). Ieee.

[13] Jia, Y. (2013). Caffe: An open source convolutional architecture for fast feature embedding. h ttp. caffe. berkeleyvision. org, 4.

[14] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).

[15] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[16] Honda, H. (2020). Digging into Detectron 2. the link https://medium. com/@ hirotoschwert/digging-into-detectron-2-47b2e794fabd.

[17] *London, W. T., & MCGLYNN, K. A. (2006). Liver cancer. Cancer epidemiology and prevention, 763-786.*

[18] Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3431-3440).

[19] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18 (pp. 234-241). Springer International Publishing.

[20] Siddique, Nahian, et al. "U-net and its variants for medical image segmentation: A review of theory and applications." Ieee Access 9 (2021): 82031-82057.

Reference na GitHub s kódem přací:
(https://github.com/BohdanYeremenko/Segmentation_parenchymatous_pig-.git)