Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

FIIT-182905-110766

Bc. Matúš Bojko

Segmentácia medicínskych obrazových dát metódami hlbokého učenia

Diplomová práca

Študijný program: Inteligentné softvérové systémy

Študijný odbor: Informatika

Miesto vypracovania: Ústav počítačového inžinierstva a aplikovanej informatiky, FIIT STU, Bratislava

Vedúci práce: Ing. Marek Jakab, PhD.

Máj 2025

Slovenská technická univerzita v Bratislave
Ústav počítačového inžinierstva a aplikovanej informatiky

Fakulta informatiky a informačných technológií
Akademický rok: 2024/2025
Evidenčné číslo: FIIT-182905-110766

ZADANIE DIPLOMOVEJ PRÁCE

Študent:	**Bc. Matúš Bojko**
ID študenta:	110766
Študijný program:	inteligentné softvérové systémy
Študijný odbor:	informatika
Vedúci práce:	Ing. Marek Jakab, PhD.
Vedúci pracoviska:	Ing. Katarína Jelemenská, PhD.

Názov práce: **Segmentácia medicínskych obrazových dát metódami hlbokého učenia**

Jazyk, v ktorom sa práca vypracuje: slovenský jazyk

Špecifikácia zadania:

Pre pomoc lekárom pri diagnostike a rozhodovaní, je potrebné segmentovať niektoré kľúčové objekty na lekárskych snímkach a extrahovať vlastnosti zo segmetovaných oblastí. Nasadenie metód hlbokého učenia sa v posledných rokoch ukázali byť efektívne vo viacerých úlohách v praxi, vrátane medicínskeho zobrazovania a segmentácie, kde dokážu lekárom zvýrazniť a ohraničiť oblasti záujmu a tak sprehľadniť zmeny anatomických alebo patologických štruktúr na snímkach. Segmentačné metódy medicínskych modalít s využitím hlbokého učenia majú však stále priestor na zlepšenie kvôli mnohým nedostatkom, ktoré sťažujú ich použitie v rôznych medicínskych doménach. Analyzujte súčasný stav problematiky využitia hlbokého učenia za účelom segmentácie obrazových dát v medicíne. Zamerajte sa na aktuálne metódy publikovaných riešení z posledných rokov a zvoľte si konkrétny problém v danej oblasti. Navrhnite vlastnú metódu alebo úpravu existujúcich riešení segmentácie medicínskych snímok, ktorú s jeho modelom implementujte využitím dostupných knižníc a rámcov. S riešením experimentujte a overte nad reálnymi dátami z medicínskych modalít. Optimalizujte parametre modelu riešenia pre dosiahnutie čo najlepšej presnosti. Výsledky vyhodnoťte prostredníctvom relevantných metrík a porovnajte s existujúcimi riešeniami v tejto oblasti.

Termín odovzdania diplomovej práce:	11. 05. 2025
Dátum schválenia zadania diplomovej práce:	15. 04. 2025
Zadanie diplomovej práce schválil:	prof. Ing. Vanda Benešová, PhD. – garantka študijného programu

Čestne vyhlasujem, že som túto prácu vypracoval samostatne, na základe konzultácií a s použitím uvedenej literatúry.

V Bratislave, 10.05.2025 Bc. Matúš Bojko

Poďakovanie

Chcel by som sa poďakovať vedúcemu Ing. Marekovi Jakabovi, PhD za skvelé vedenie záverečnej práce, samostatný prístup a efektívnu komunikáciu v akomkoľvek čase a počas celej doby práce na projekte.

Nezabudnem ani na neustálu podporu od mojej priateľky v akýkoľvek chvíľach a počas celého štúdia, za čo jej ďakujem.

Anotácia

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Študijný program: Inteligentné softvérové systémy

Autor: Bc. Matúš Bojko
Diplomová práca: Segmentácia medicínskych obrazových dát metódami hlbokého učenia
Vedúci diplomovej práce: Ing. Marek Jakab, PhD.
Máj 2025

Techniky spracovania obrazu sú dôležitou súčasťou oblasti analýzy medicínskych obrazových dát, kde môžu významne napomôcť lekárom pri rozhodovaní, identifikácii a diagnostike chorôb. V tejto oblasti použitie hlbokých neurónových sietí urýchlilo pokrok v klasifikácii, detekcii a segmentácii analyzovaných modalít. Práve metódy segmentácie dokážu lekárom zvýrazniť a ohraničiť oblasti záujmu, čo sa ukazuje byť užitočné v rôznych doménach vrátane rádiológie, dermatológie či histológie. Existujúce metódy založené na hlbokom učení, však vyžadujú veľké množstvo anotovaných dát, ktoré sú nákladné na získanie.

Táto práca je preto zameraná na segmentáciu medicínskych obrazových dát s využitím hlbokého učenia. Vrámci analýzy sme zhrnuli hlboké učenie a bližšie sa zamerali na state-of-the-art prístupy využívajúce neoznačené dáta. Zo získaných znalostí sme navrhli U-Net model s 2 dekódermi a pomocnými hlavami segmentácie. Pre model sme navrhli hybridný učiaci rámec a stratovú funkciu využívajúcu konzistentnú regularizáciu medzi pyramídovými výstupmi pre učenie z označených aj neoznačených dát. So stratovou funkciou sme ďalej experimentovali a najlepší z prototypov porovnali s existujúcimi riešeniami na verejnom ACDC datasete.

Anotation

Slovak University of Technology Bratislava
Faculty of Informatics and Information Technologies
Degree Course: Intelligent Software Systems

Author: Bc. Matúš Bojko
Master's Thesis: Medical image segmentation using deep learning methods
Supervisor: Ing. Marek Jakab, PhD.
2025, May

Image processing techniques are an important part of medical image analysis, where they can significantly assist physicians in decision making, identification and diagnosis of diseases. In this field, the use of deep neural networks has accelerated advances in the classification, detection and segmentation of analyzed modalities. Segmentation methods especially, can highlight and delineate regions of interest to physicians, which is proving useful in a variety of domains including radiology, dermatology, and histology. Existing methods based on deep learning, however, require large amounts of annotated data, which are expensive to obtain.

Therefore, this work focuses on medical image segmentation using deep learning. As part of the analysis, we summarize deep learning and take a closer look at state-of-the-art approaches using unlabeled data. From the knowledge gained, we proposed U-Net model with 2 decoders and auxiliary segmentation heads. For the model, we proposed a hybrid learning framework and a loss function using consistency regularization between pyramid predictions for learning from both labeled and unlabeled data. We further experimented with the loss function and compared the best prototype with existing approches on the public ACDC dataset.

Obsah

1 Úvod **1**

2 Hlboké učenie **3**

2.1 Umelé neurónové siete 3

2.1.1 Diskrétny perceptron 4

2.1.2 Viacvrstvový perceptron 4

2.1.3 Aktivačné funkcie 5

2.1.4 Tensory 7

2.2 Optimalizácia 7

2.2.1 Stratové funkcie 8

2.2.2 Gradient descent a algoritmus spätného prechodu 8

2.3 Konvolučné neurónové siete 10

2.3.1 Konvolúcia 10

2.3.1.1 Vlastnosti konvolúcie 11

2.3.1.2 Padding a stride 11

2.3.2 Združovanie 11

2.3.3 Architektúry konvolučných sietí 12

2.4 Autoenkóder 13

2.4.1 Regularizované autoenkódery 14

2.4.2 Konvolučný autoenkóder 15

3 Segmentácia obrazových dát **17**

3.1 Medicínske obrazové modality a datasety 18

3.2 Metódy segmentácie 19

3.2.1 Metódy riadenej segmentácie 19

3.2.2 Metódy slabo riadenej segmentácie 21

4 Analýza súvisiacich prác **23**

4.1 Uncertainty Rectified Pyramid Consistency 23

4.1.1 Nastavenie stratových funkcií 24

4.1.2 Datasety 26

4.1.3 Architektúra a nastavenie trénovania 26

4.1.4 Experimenty 27

4.2 Cross Teaching CNN & Transformer 28

4.2.1 Krížové učenie a nastavenie stratových funkcií 29

4.2.2 Dataset 30

4.2.3 Architektúra a nastavenie trénovania 30

4.2.4 Experimenty 31

4.3 Cross Distillation of Multiple Attentions 32

4.3.1 Architektúra MTNet 33

4.3.2 Destilácia znalostí dekóderov a minimalizácia neistoty 33

4.3.3 Dataset a nastavenie trénovania 34

4.3.4 Experimenty 35

4.3.5 Zhrnutie riešení a možné vylepšenia 36

5 Návrh **37**

5.1 Návrh učenia regularizáciou konzistencie 37

5.2 Architektúra 40

5.3 Návrh stratovej funkcie 42
5.4 Metodológia experimentovania 45
5.4.1 Dataset 46
5.4.2 Nastavenie trénovania a implementácia 47
5.4.2.1 Predspracovanie a augmentácie 47
5.4.2.2 Hyperparametre 48
5.4.3 Metriky 48
5.4.3.1 Oblastne založené metriky 49
5.4.3.2 Hranične založené metriky 49
5.4.3.3 Logovanie a vyhodnocovanie metrík 50
5.5 Porovnanie základu so súvisiacimi prácami 50
5.6 Experimenty čiastočného riadenia 52
5.6.1 Čiastočné riadenie na základe MSE 53
5.6.2 Čiastočné riadenie na základe KL 55
5.6.3 Prototypy kombinácií váh a vzdialenostných metrík 58
5.6.4 Prototypy s dodatočnou minimalizáciou entropie 59
5.6.4.1 Zhrnutie experimentovania s čiastočným riadením . 61
5.7 Experimenty plného riadenia 61
5.7.1 Pridanie krížovej entropie alebo fokálnej straty 62
5.7.2 Nastavenie plného riadenia na pomocných výstupoch 64
5.8 Experiment metód zväčšovania rozlíšenia 65

6 Výsledky **67**
6.1 Kvantitatívne výsledky 67
6.1.1 Experimenty čiastočného riadenia 67
6.1.2 Experimenty plného riadenia 70
6.1.3 Experiment metód zväčšovania rozlíšenia 71
6.1.4 Porovnanie s existujúcimi riešeniami 71

6.2 Kvalitatívne výsledky 73

7 Zhodnotenie **75**

7.0.1 Možné vylepšenia 76

Literatúra **77**

Príloha A: Technická dokumentácia **A-1**

A.1 Požiadavky a závislosti A-1

A.1.1 Softvérové knižnice A-2

A.1.2 Hardvérové požiadavky A-2

A.2 Dataset A-3

A.3 Štruktúra projektu A-3

A.4 Funkcionality a použitie riešenia A-4

Príloha B: Plán práce **B-1**

B.1 Harmonogram práce DP1 . . . . . B-1

B.2 Harmonogram práce DP2 . . . . . B-1

B.3 Harmonogram práce DP3 . . . . . B-2

Príloha C: Digitálna časť práce **C-1**

Príloha D: IITSRC Poster **D-1**

1. Úvod

Úlohy spracovania obrazu majú svoje uplatnenie v rôznych oblastiach vrátane autonómneho riadenia, robotiky či medicíny. Dôležitou súčasťou sú v oblasti analýzy medicínskych obrazových dát, kde môžu významne napomôcť lekárom pri rozhodovaní, identifikácii a diagnostike chorôb. V tejto oblasti použitie hlbokých neurónových sietí urýchlilo pokrok v klasifikácii, detekcii a segmentácii medicínskych modalít. Práve metódy segmentácie dokážu lekárom zvýrazniť a ohraničiť oblasti záujmu, čo sa ukazuje byť užitočné v rôznych doménach vrátane rádiológie, oftalmológie, dermatológie či histológie.[47] Je tak možné napomôcť pri včasnej diagnostike rôznych typov rakovín a iných ochorení.[61]

Pokroky v hlbokom učení ako konvolučné neuronové siete (CNN) [64] a najnovšie tzv. Vision Transformery (ViT) [14] dosahujú pozoruhodný výkon v širokom spektre medicínskych úloh vrátane segmentácie obrazu. Tieto metódy v plne riadenom nastavení často vyžadujú veľké množstvo anotovaných dát. Problémom je, že získanie veľkého množstva pixelovo presných anotácií pre trénovanie segmentačných modelov je časovo náročný proces pre doménových expertov a lekárov. Ako výsledok, metódy, ktoré vedia pre tréning využiť malé množstvo dostupných anotácií a často veľký počet ne-anotovaných dát dostávajú čoraz väčšiu pozornosť výskumu. Preto sa v našej práci zameriame práve na podobné metódy segmentácie medicínskych obrazových dát.

2. Hlboké učenie

Hlboké učenie, ako pod-oblasť strojového učenia, sa snaží budovať matematické modely učiace sa užitočné reprezentácie prostredníctvom hierarchickej štruktúry postavenej z množstva výpočtových vrstiev. Prostredníctvom naučených transformácií na každej vrstve, sú schopné z neznámych štruktúr vstupných dát extrahovať rôzne vlastnosti viacerých úrovní abstrakcie. [29] Dostatočným počtom takých vrstiev je možné budovať komplexné funkcie, ktoré v posledných rokoch prekonávajú štandardné metódy strojového učenia nasadené v oblastiach počítačového videnia, spracovania prirodzeného jazyka ako aj medicíny. [16] V úlohách postavených na ich dátach to nie je výnimkou ani v súčasných prístupoch segmentácie.

V rámci tejto kapitoly preto rozoberieme základné aspekty neurónových sietí ako súčasť hlbokého učenia, na ktorých sú postavené aj aktuálne metódy segmentácie medicínskych obrazových dát, ktoré výrazne benefitujú z ich schopností.

2.1 Umelé neurónové siete

Umelé neurónové siete (UNN) predstavujú základný koncept algoritmov hlbokého učenia, ktoré simulujú mechanizmy učenia sa biologických neurónových sietí. Nervový systém mozgu je postavený na nervových bunkách (neurónoch) a prepojeniach medzi ich dendridmi a axónmi. Regióny prepojenia (synapsy) prenášajúce elektrické impulzy kontrolujú ich silu, ktorá je premenlivá na podnet

vonkajších impulzov, čo je základom schopnosti učenia sa. [2] Na danej myšlienke stavujú aj dnešné UNN, s prvým konceptom umelého neurónu z roku 1943 ako matematického modelu od McCulloch a Pitts. [54]

2.1.1 Diskrétny perceptron

V nadväznosti aj na ich prácu, Rosenblatt v roku 1958 skonštruoval tzv. perceptron [50], fyzický stroj umelého neurónu, ktorý rozširoval známy model o váhy ako upraviteľné parametre. Operáciu umelého neurónu je možné modelovať ako lineárnu funkciu:

$$\hat{y} = f(x, w) = g(\sum_{j=1}^{n} w_j x_j + b) \tag{2.1}$$

Kde x je vektor nezávislých premenných, w je vektor váh, b je posunutie (tzv. *bias*) a g je aktivačná funkcia (v prípade perceptronu kroková funkcia). Výstupom je potom binárna predikcia $\hat{y}$. Ak potom pre všetky x považujeme ich príslušné y ako očakávané závislé premenné, môže byť model optimalizovaný úpravou váh prostredníctvom jednoduchého kritéria: $w \Leftarrow w + \alpha(y - \hat{y})x$, kde α je predom nastaviteľná rýchlosť učenia a $(y - \hat{y})$ predstavuje tzv. *stratu* (chybu) v predikcii. [2] Tento prístup optimalizácie je dnes nahradený komplexnejšími algoritmami zastupujúceho gradientu straty v podobe diferencovateľnej funkcie (viac kap. 2.2).

2.1.2 Viacvrstvový perceptron

Paralelné zapojenie perceptronov do tzv. *jedno-vrstvového perceptronu* umožňuje riešiť viac-triednu klasifikáciu, avšak stále len relatívne jednoduchej úlohy.

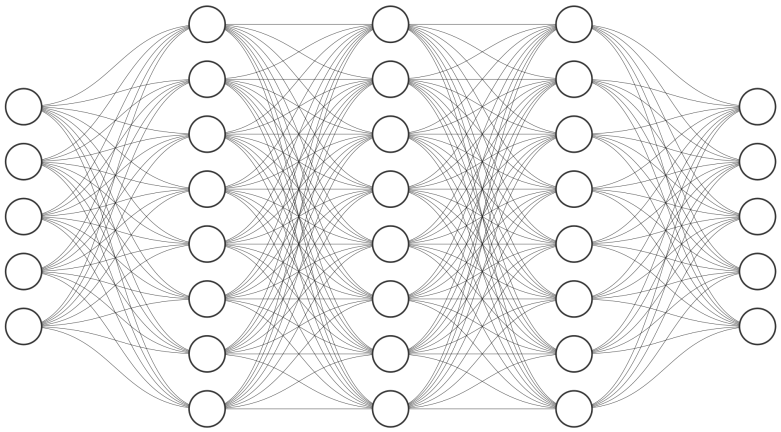
Viac-vrstvové perceptrony (ang. MLP), taktiež dopredné UNN, na druhej strane stavajú na myšlienke sekvenčných výpočtov viacerých vrstiev. Pre toto nastavenie je charakteristické plné prepojenie, kedy výstup každého neurónu vrstvy je váhovo prepojený so všetkými neurónmi nasledujúcej vrstvy. Vstupný vektor x,

ako dátová vzorka reprezentovaná vstupnou vrstvou, je potom postupne prešírená ďalšími tzv. skrytými vrstvami až na výstupnú vrstvu. Na tejto vrstve každý neurón vyhodnocuje odhad pravdepodobnosti triedy, alebo odhad spojitej hodnoty v závislosti od toho, či sieť rieši klasifikačnú alebo regresnú úlohu.

Dopredné spracovanie celou sieťou na jednotlivých vrstvách má potom tvar: [44]:

$$A^{[l]} = g(W^{[l]}A^{[l-1]} + b^{[l]})^{[l]} \tag{2.2}$$

Kde l je číslo vrstvy, $g^{[l]}$ je aktivačná funkcia a $W^{[l]}$ je matica váh s rozmermi $n^l \times n^{l-1}$ kde n je počet neurónov. Potom $b^{[l]}$ je vektor posunutí pre každý neurón vo vrstve l a A je matica aktivácií teda výstup l-tej vrstvy. V prípade vstupnej vrstvy ($l = 0$) platí $A = X$, kde X je matica dávky (tzv. batch) viacerých vstupných vzoriek. Touto maticovou reprezentáciou môže byť dávka vzoriek spracovaná naraz v 1 dopredanej fáze čo zefektívňuje samotný výpočet.



Obr. 2.1: Viac-vrstvový perpceptron.

2.1.3 Aktivačné funkcie

Zapájaním veľkého množstva skrytých vrstiev je snaha budovať tzv. *hlboké neuronové siete (ang. DNN)*. Aby boli schopné modelovať zložitejšie funkcie, je na výstupy skrytých vrstiev aplikovaná ne-linearita v podobe spojitých *aktivačných*

funkcií čo umožňuje riešiť ne-lineárne separovateľné problémy. Spolu s ich použitím a dodatočnou hĺbkou množstva vrstiev sú DNN schopné učenia sa komplexnejších vzorov z dát [13]. Medzi najvýznamnejšie aktivačné funkcie patria:

Sigmoid: je aktivačná funkcia dnes používaná na výstupnej vrstve sietí binárnej klasifikácie v podobe odhadu pravdepodobnosti pozitívnej triedy v rozsahu (0,1). Jej použitie v skrytých vrstvách je nahradené funkciami ako ReLU, kvôli saturácii a problému miznúceho gradientu, ktorý spomaľuje učenie sietí.

$$g(x) = \frac{1}{1 + e^{-x}} \tag{2.3}$$

Hyperbolický Tangent: je taktiež monotónna funkcia, symetrická okolo 0 s oborom hodnôt (-1,1). V okolí 0 má gradient 4-krát väčší než sigmoida, čo prispieva k rýchlejšiemu učeniu, preto je preferovanejšou voľbou v skrytých vrstvách, avšak je spojená s rovnakými problémami. [13]

$$g(x) = tahn(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.4}$$

ReLU: je dnes štandard pre aktivácie na skrytých vrstvách UNN vďaka jednoduchšiemu výpočtu a rýchlejšej konvergencii než predošlé funkcie. Pre $x \geq 0$ je gradient funkcie konštantný v bode 1, vďaka čomu je menej náchylná na uviaznutie v lokálnych minimách a rieši miznúce gradienty.[13] Sieť s ReLU aktiváciami však môže trpieť problémom "mŕtveho reluä jeho neurónov pri častých hodnotách $x < 0$.

$$g(x) = ReLU(x) = max(0, x) = \begin{cases} x & \text{ak } x \geq 0, \\ 0 & \text{ak } x < 0 \end{cases} \tag{2.5}$$

Problém mŕtveho relu rieši funkcia **lReLU** prostredníctvom konštanty $\alpha = 0.01$ tak aby hodnoty x < 0 a ich gradienty boli nenulové. Ďalším vylepšením je funkcia **PReLU**, ktorá navyše pridáva parameter α medzi tréningom naučiteľné

zložky.

$$g(x) = lReLU(x) = max(ax, x) = \begin{cases} x & \text{ak } x \geq 0, \\ ax & \text{ak } x < 0 \end{cases} \tag{2.6}$$

Použitie aktivačných funkcií bolo intenzívne skúmanou oblasťou dnes s aktuálnejšími funkciami ako napr. Swish [48], GELU [19] a podobné, pričom ich výber spravidla zavisí od špecifických potrieb a architektúry siete.

2.1.4 Tensory

Parametre DNN sietí ako aj obrazové dáta sa pre potreby efektívneho spracovania reprezentujú v podobe matíc vyšších rádov tzv. *tensorov*.[64]. Ich implementácia je optimalizovaná pre maticové operácie, ktoré môžu byť plne obslúžené grafickými kartami.

Pre klasické obrázky sú tensory štandardne v 4D tvare: $\mathsf{A} \in \mathbb{R}^{B \times C \times H \times W}$ kde H, W sú výška a šírka obrazu v pixel-och, C je počet kanálov a B je veľkosť dávky. RGB obrázok v podobe tensoru s 3 farbenými kanálmi (R - červená, G - zelená, B - modrá) pre každý pixel má potom rozmer $1 \times 3 \times H \times W$.

2.2 Optimalizácia

Úlohou optimalizácie (trénovania) DNN modelov je využiť dostupné dáta na hľadanie optimálnych parametrov siete pre riešenie úlohy. Rovnako ako pri odlišných metódach strojového učenia, sú pre tieto účely cieľové dáta rozdelené do trénovacej, testovacej a validačnej sady alebo do viacerých rozdelení prostredníctvom stratégií krížovej validácie. Formálne je cieľom nájsť také parametre θ, ktoré redukujú nákladovú funkciu $J(\theta)$, ktorá typicky vyhodnocuje výkonnosť/chybovosť modelu v požadovanej úlohe na základe celej trénovanej sady. [15]

2.2.1 Stratové funkcie

Na výpočet chyby modelu v jej predikcii $f(x)$ (tiež $\hat{y}$) pre vstupné pozorovanie x voči očakávanej hodnote y, sa používa stratová funkcia L. Nákladová funkcia je potom priemerná strata naprieč všetkým trénovacím pozorovaniam m v tvare: $J(\theta) = \frac{1}{m} \sum_i^m L(y_i, f(x_i))$.

Výber stratovej funkcie závisí od špecifickej úlohy a architektúry siete. V prípade regresných úloh sú najznámejšie L1, L2 alebo Huberova strata ktorá kombinuje výhody dvoch spomenutých. [60] Pri klasifikačných úlohách sa najčastejšie používa krížová entropia, pre binárnu klasifikáciu v tvare:

$$L(y, \hat{y}) = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y}) \tag{2.7}$$

Kde y je očakávaná trieda $(0/1)$ a $\hat{y}$ je pravdepodobnosť pozitívnej triedy.

2.2.2 Gradient descent a algoritmus spätného prechodu

Hľadanie globálneho minima nie je priamočiare vzhľadom na množstvo parametrov a výpočtovú zložitosť, preto DNN využívajú metódy založené na zostupujúcom gradiente tzv. **gradient descent**. Ide o algoritmy, ktoré iteratíve minimalizujú nákladovú funkciu úpravou parametrov podľa pravidla [2]:

$$\overline{W} \Leftarrow \overline{W} - \alpha \nabla L; \quad \nabla L = \frac{\partial L}{\partial \overline{W}} = \left[\frac{\partial L}{\partial w_1} \dots \frac{\partial L}{\partial w_d} \right] \tag{2.8}$$

Kde W je vektor parametrov (váh) a L je gradient vektor nákladovej funkcie, ktorého zložky sú parciálne derivácie voči jednotlivým parametrom. Jeho veľkosť a smer udáva naj-strmšie stúpanie funkcie v danom bode, preto aplikáciou jeho záporu na parametre je v opačnom smere strata minimalizovaná. Hyperparameter α potom kontroluje veľkosť kroku (*rýchlosť učenia*). Výpočet v maticovej podobe je obdobný a analogický je postup aj v prípade *bias* posunutí.

Pri štandardnom nastavení algoritmu ide o *batch* gradient descent kedy je výpočet nákladovej funkcie a jeden učiaci krok vzhľadom na celú trénovaciu množinu. Opačnou variantou je *stochastic* verzia, ktorá aktualizuje parametre po každom spracovanom pozorovaní. Strednou variantov je *mini-batch* gradient descent s učiacim krokom po každej dávke pozorovaní ako kompromis medzi stabilitou učenia, rýchlosťou a pamäťovými nárokmi predchádzajúcich prístupov. [2]

Prelomovým v učení DNN bolo predstavenie algoritmu **spätného šírenia chyby** [52], ktorý na efektívny výpočet gradientov váh a biasov pre každú vrstvu využíva fakt, že celá sieť aj s jej aktivačnými funkciami je zloženou diferencovateľnou funkciou, ktorú je možné reprezentovať ako výpočtový graf. Algoritmus potom využíva pravidlo derivovania zloženej funkcie (chain rule) a má 3 fázy:

- 1. fáza - dopredný prechod (*forward pass*), v ktorom je celou sieťou spracovaný mini-batch pozorovaní a vyhodnotená nákladová funkcia. Počas spracovania sú ukladané medzi-výsledky spracovania potrebné pre výpočet gradientov - v prípade lineárnej vrstvy i bez aktivácie je to $\bar{z}_{i+1} = W\bar{z}_i$ [1]

- 2. fáza - spätný prechod (*backward pass*), v ktorom sú spätne od nákladovej funkcie počítané gradienty voči váham a biasom vrstiev a aktivačných funkcií, ktoré sú sekvenčne propagované do predchádzajúcich vrstiev podľa reťazového pravidla. Pre príklad siete len lineárnych vrstiev, je spätne propagovaný gradient vektor $\bar{g}_i$ vrstvy i v tvare $\bar{g}_i = W^T\bar{g}_{i+1}$. Gradient pre aktualizáciu váh i-tej vrstvy je potom $\frac{\partial L}{\partial W} = \bar{g}_i\bar{z}_{i-1}^T$ [1].

- 3. fáza - krok gradient descent, v ktorom sa aktualizujú parametre vrstiev podľa vypočítaných gradientov na jednotlivých vrstvách.

Na základe štandardného gradient descentu existujú optimizátory ako napr. **SGD s momentom**, ktorý sa snaží riešiť stabilitu gradientov a uviaznutia v lokálnych

minimách, **RMSProp**, ktorý rieši konštantnú mieru učenia na rôznych vrstvách alebo **Adam** [26], ako kombinácia výhod oboch prístupov.

2.3 Konvolučné neurónové siete

Z pohľadu moderných sietí boli a naďalej sú pokrokové **konvolučné neurónové siete (CNN)**, ako samostatná kategória architektúr DNN, ktoré využívajú ako primárny výpočtový prostriedok operáciu konvolúcie.

Použitím konvolúcií na jednotlivých vrstvách, sieť počas inferencie a tréningu zohľadňuje priestorové rozloženie dát a vlastnosti vychádzajúce z ich vzájomnej pozície. Z tejto schopnosti benefitujú predovšetkým úlohy spracovania obrazových dát, kde pozícia, rotácia, škála a rôzne farebné vlastnosti objektov uložené v pixeloch/voxeloch, nesú významné kontextové informácie pre riešenie úlohy.

2.3.1 Konvolúcia

Konvolučná operácia je špecializovaný druh lineárnej operácie využívajúca tzv. kernel matice na transformáciu dát. Z hľadiska efektivity výpočtu sa pre DNN implementuje v podobe cross-korelácie. Ak I je matica pixelov 1-kanálového obrazu a K je matica kernelu (štandardne rozmerov 3x3 alebo 5x5), ich konvolúcia je potom operácia $(K * I)$, kde výstup S je *mapa príznakov (feature map)* [63]:

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i+m, j+n) K(m,n). \tag{2.9}$$

Ako zobrazuje obr. 2.2, kernely sa správajú ako posuvné okná, ktoré na každej z možných pozícií vstupnej mapy vykonávajú hadamardov súčin. Výsledné elementy každého súčinu sú sčítané do jedného elementu na výstupnej mape príznakov. Platí, že pre viac-kanálové obrázky/mapy musí mať kernel vrstvy rovnakú hĺbku. Počet kernelov konv. vrstvy potom zodpovedá hĺbke jej výstupnej mapy príznakov.

2.3.1.1 Vlastnosti konvolúcie

Kernely sú optimalizované gradient descent prístupmi ako štandardné váhy. Ide o efektívne zdieľanie tzv. **riedkych váh**[15], pretože ich znovupoužívaním je možné detekovať príznaky na rôznych miestach obrázka.

Konvolúcie majú vlastnosť **recepčných polí** kedy každý prvok vstupnej matice vrstvy je výsledkom susedných prvkov vstupných matíc predošlých vrstiev. To umožňuje konvolučným vrstvám postupne extrahovať elementárne príznaky (hrany, rohy objektov) až po príznaky vyšších rádov v ďalších vrstvách. [59]

Konvolúcia je **invariantná k translácii** objektov [16], no nie je invariantná k ich škálovaniu a rotácii, preto je časté použitie augmentačných techník (náhodné rotácie, výrezy a pod.) pre zaradenie hraničných prípadov do trénovanej množiny.

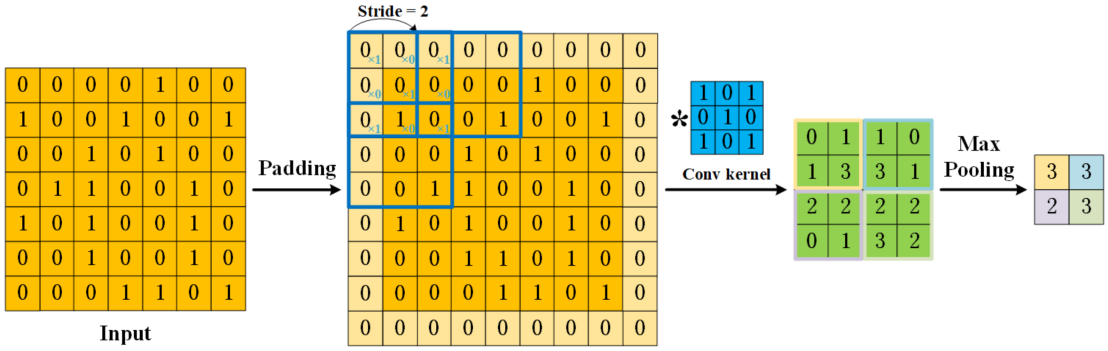
2.3.1.2 Padding a stride

Konvolúcia pri klasickom nastavení zmenšuje priestorové rozmery (rozlíšenie) vstupnej mapy príznakov. Kontrolu nad veľkosťou zmenšenia nastavuje hyperparameter kroku kernelu (tzv. *stride*) štandardne veľkosti 1. Pre zachovanie rozlíšenia je na okolie vstupnej mapy pridávaná dodatočná výplň tzv. *padding* v podobe konštanty (núl), zrkadlenia, kópie krajných pixelov a pod.

2.3.2 Združovanie

Prehlbovanie siete ďalšími konv. vrstvami zväčšuje recepčné pole výstupnej mapy siete, avšak nie v takej miere ako by bolo potrebné pre obrázky vysokých rozlíšení. Riešením sú združovacie vrstvy, ktoré redukujú priestorové rozmery vstupnej mapy (výšku a šírku) nahrádzaním susediacich elementov na rôznych pozíciách sumárnou štatistikou (napr. maximom, priemerom) (viď obr. 2.2). [73, 15] Zredukované rozlíšenie (so zachovanou hĺbkou) vedie k menšej výpočtovej réžii a zväčše-

niu recepčného poľa bez dodatočných parametrov a straty kľúčových informácií. Operácia zároveň podporuje invariantnosť k transláciám a zlepšuje generalizáciu modelu. [59] Použitie rôznych združovacích techník je skúmanou oblasťou pričom najpoužívanejšími sú maximálne a priemerné združovanie.



Obr. 2.2: Vizualizácia konvolúcie a združovania.[34] Na vstup je aplikovaný 1x1 padding a vykonaná konvolúcia 3x3 kernelom s posunom stride 2. Na výstup konvolúcie (zelená mapa) je aplikované max. združovanie kernelom 2x2 so stride 2.

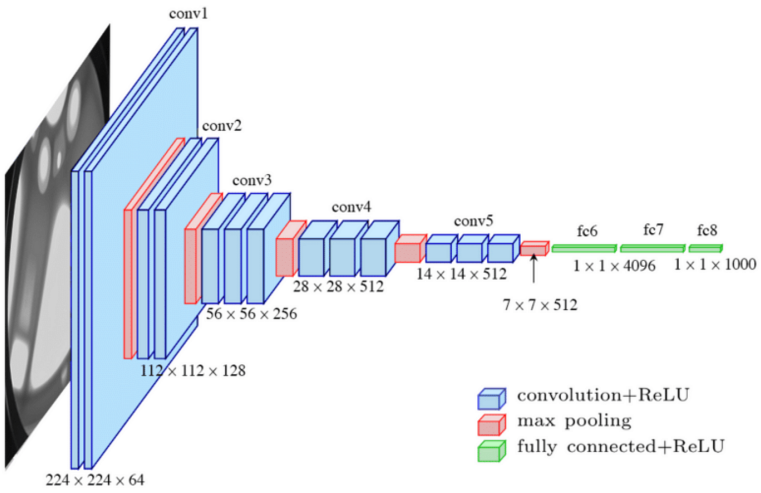
2.3.3 Architektúry konvolučných sietí

Základným prvkom architektúr CNN sú **konvolučné bloky** kde jeden štandardne obsahuje 1 alebo viac sekvenčných konv. vrstiev, ktorým nasleduje dodatočná nelinearita v podobe aktivačnej funkcie ako ReLU. Mapy príznakov medzi konvolúciami a aktiváciami sú často normalizované napr. dávkovou (batch) [21] alebo skupinovou (group) [67] normalizáciou ako dodatočná regularizácia pre lepšiu konvergenciu a stabilitu počas tréningu.

Hlavná časť CNN tzv. **extraktor príznakov** (*ang. feature extractor*), je typicky zložený zo zasebou zapájaných konv. blokov, medzi ktorými môže byť združovacou vrstvou redukované rozlíšenie. Počet kernelov sa potom často postupne po blokoch navyšuje, pre kompenzáciu informácií, ktoré by sa mohli redukciou stratiť. Za extraktorom štandardne nasleduje **hlava**, ktorá prispôsobuje výstup siete pre konkrétnu úlohu. Pre viac-triednu klasifikáciu je hlavou napr. globálne priemerne

združovanie spolu s MLP, kde výstupná vrstva má neurón pre každú triedu.

Známa architektúra začínajúca rozmach CNN je AlexNet (2012), ktorá významne prekonala výsledky predchodcov. Na podobnej štruktúre je doteraz stavaných množstvo známych CNN riešení (viď obr. 2.3) vylepšujúc rôzne aspekty architektúry ako napr. VGGNet [55], ResNet [17] alebo EfficientNet [56].



Obr. 2.3: VGG-16 architektúra [55] s 13 konvolučnými blokmi (modré) 5 združovacími vrstvami (červené) a MLP hlavou pre klasifikáciu 1000 tried. [25]

2.4 Autoenkóder

Autoenkóder je pristúp v architektúre neurónových sietí, pre učenie jednoduchších reprezentácií redukciou dimenzionality vstupných dát a ich spätnou rekonštrukciou do pôvodnej alebo požadovanej podoby.

Skladá sa z 2 hlavných častí: **enkóder** - ktorý transformuje vstup X do latentnej reprezentácie nižšej dimenzionality za účelom extrakcie len dôležitých príznakov a **dekóder** - ktorý zostavuje komprimované dáta späť do pôvodného priestoru. V štandardnom nastavení ide o prístup učenia bez učiteľa, to znamená, že úlohou je čo najviac priblížiť vstup siete X, k zostavenému výstupu dekódera X^d mini-

malizáciou rekonštrukčnej straty $L(X, X^d)$, ktorou môže byť L1, krížová entropia alebo podobné funkcie v prípade klasifikácie.

Autoenkódery boli tradične používané na redukciu dimenzionality a učenie reprezentácií, no v rámci ďalšieho pokroku majú ich princípy uplatnenie pri detekcii anomálií, odporúčacích systémoch, jazykových modeloch a úloh spracovania a generovania obrazu (klasifikácia, segmentácia, detekcia a pod.). [31, 4]

2.4.1 Regularizované autoenkódery

Častým prístupom zmenšenia dimenzie reprezentácií je **tzv. bottleneck-om** siete kedy skryté vrstvy enkódera sú postupne užšie (s menším počtom neurónov) až po najužšiu (bottleneck) vrstvu a naopak v dekodéri siete postupne širšie až do tvaru vstupu. V tomto nastavení je však generalizácia sťažená pri vysoko kapacitných nastaveniach. Naopak ak vrstvy zachovávajú alebo zväčšujú šírku, môže byť pre sieť jednoduché vykonávať len funkciu identity. [15] Vylepšené verzie originálneho autoenkódera riešia tento kompromis zavádzaním **regularizačných techník**.

Riedky (sparse) autoenkóder pridáva k trénovaciemu kritériu tzv. *sparse penalty* $\Omega(h)$ ako ďalšiu zložku rekonštrukčnej straty $L(X, X^d) + \Omega(h)$ kde $f(X) = h$ je výstup z enkódera. Zložka $\Omega(h)$ môže byť L1 normalizácia alebo KL divergencia, ktorá penalizuje aktivácie tak, že pre určité neuróny sú blízke 0. To umožňuje zachovať šírku siete bez použitia *bottleneck* vrstvy. [15, 31]

Odšumovací (denoising) autoenkóder namiesto penalizačnej zložky, upravuje rekonštrukčnú stratu minimalizáciou $L(X, g(f(\tilde{X})))$, kde $g()$ je dekóder siete a $\tilde{X}$ je vstup s pridaným šumom (napr. Gaussovým). [4] Autoenkóder je týmto nastavením nútený zostaviť pôvodnú (čistú) verziu vstupu, preto sa často používa na korekciu a odstraňovanie šumu, rozmazaní a podobných chýb.

Odlišnou stratégiou regularizácie, ktoré používajú tzv. **kontrastívne (contrac-**

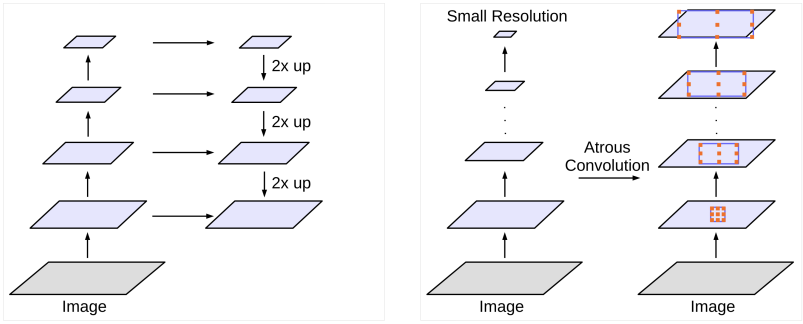
tive) autoenkódery je penalizácia gradientov prostredníctvom zložky v tvare $\Omega(h, x) = \sum_i \|\nabla_x h_i\|^2$ kde ∇_x je gradient vzhľadom na x. Táto penalizácia núti sieť učiť sa funkciu, ktorá sa veľmi nemení pri malej zmene vo vstupe X. [15]

2.4.2 Konvolučný autoenkóder

S príchodom CNN boli vyvinuté aj konvolučné autoenkódery pre obrazové dáta, pri ktorých by sa použitím lineárnych vrstiev strácala priestorová informácia.

Tieto modely využívajú konvolučné a združovacie vrstvy v enkódery pre zmenšenie rozlíšenia a extrakciu príznakov. V dekodéry je potom používaná transponovaná konvolúcia ako opačná operácia konvolúcie alebo alternatívy v podobe kombinácie 1x1 konvolúcie pre nastavenie hĺbky aktivačných máp a tzv. *upsampling* vrstvy na zväčšenie rozlíšenia bilineárnou interpoláciou alebo algoritmom najbližšieho suseda. Odlišný prístup popularizovaný modelom DeepLabv3 [9] je atriálna konvolúcia špecifická medzerami medzi elementami kernel filtrov, čo zachováva rozlíšenie (nestráca priestorové informácie) a zväčšuje recepčné pole (viď obr. 2.4).

Konvolučné autoenkódery majú veľké uplatnenie v segmentácii obrazu kde prelomovým riešením bola sieť U-Net [49] so skokovými prepojeniami, na ktorej základoch stavajú dnešné architektúry nielen segmentačných sietí (viac v kap. 3).



Obr. 2.4: Alternatívne prístupy zachytávania viac-škálového kontextu [9]: vľavo štandardný enkóder-dekóder, vpravo atriálna konvolúcia.

3. Segmentácia obrazových dát

Narastajúci počet medicínskych dát vrátane 2D a 3D modalít, ktoré je potrebné lekármi interpretovať, podporil rozvoj počítačových nástrojov a algoritmov analýzy medicínskych obrazov pre vykonanie rýchlych, presných a objektívnych meraní. Zároveň pri analýze obrazových dát, je jedna z potrieb správne ohraničiť kľúčové objekty a regióny pre stanovenie presných diagnóz.

Segmentácia snímok resp. obrazových dát, ako jedna z úloh počítačového videnia preto zohráva ústrednú rolu v medicínskom zobrazovaní a analýzy ich modalít. V tejto oblasti je snahou segmentácie jasnejšie zvýrazniť anatomické štruktúry a patologické zmeny na snímkach. Ich presná identifikácia môže poskytnúť cenné obsahové a tvarové informácie napr. pre extrakciu tumorov, meranie objemu tkaniva alebo ďalšie klinické aplikácie ako plánovanie zákrokov či diagnostika chorôb.

Segmentácia obrazu, ako úloha spracovania obrazu, je proces rozdelenia 2D alebo 3D obrazu na viacero segmentov na základe podobností alebo odlišností medzi regiónmi. Vzhľadom na klasifikáciu rôznych typov sú známe 2 kategórie segmentácie [29]:

- Sémantická segmentácia (*semantic segmentation*)
- Inštančná segmentácia (*instance segmentation*)

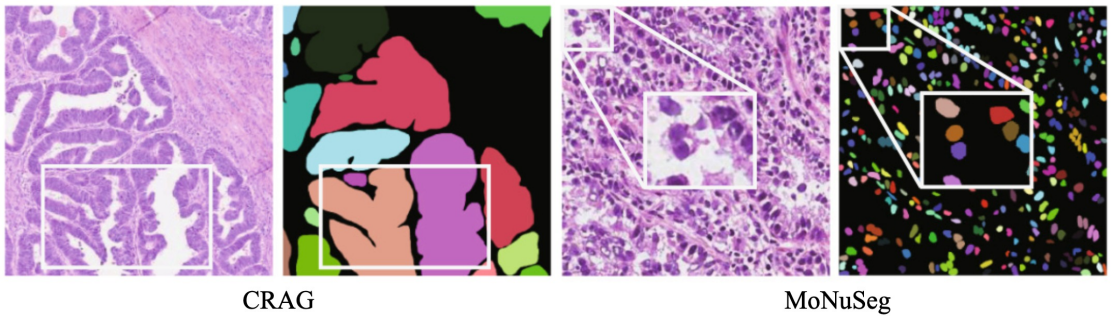
Sémantická segmentácia je klasifikačná úloha, ktorá priradzuje triedu z dostup-

ného rozsahu každému pixelu obrazu tak, aby spolu závislé pixely patrili do spoločnej oblasti. Tento typ je podmnožinou **segmentácie inštancií**, ktorá je doplnená o identifikáciu inštancií objektov z dostupných kategórií. V posledných rokoch sa výskum v segmentácii medicínskych obrazov zameriava viac na sémantické metódy, pretože každý orgán/tkanivo je odlišné, čo sťažuje identifikáciu inštancií.

3.1 Medicínske obrazové modality a datasety

Dnešné segmentačné metódy sa typicky prispôsobujú rôznym medicínskym modalitám, s ktorými pracujú lekári ako napríklad: *MRI, CT, röntgen, RGB* ale aj *histologické, mikroskopické a fundoskopické snímky*. Pozorujeme však v posledných rokoch aj multi-modálne metódy, ktoré sa snažia do svojho algoritmu kombinovať znalosti z viacerých modalít (napr. CT a MRI).

Verejné datasety: Pre podporu výskumu segmentačných techník vychádza veľké množstvo datasetov v rámci rôznych súťaží ako napr. BRATS [42], ISLES [22], KiTS [18], LiTS [6] konkrétne s anotovanými vzorkami mozgu alebo orgánov brušnej dutiny. Iné zdroje, sa zasa sústreďujú na archiváciu datasetov viacerých modalít prevažne nádorových ochorení. [3, 58]. V našej práci sa zameriame na 2D dataset ako napr. ACDC (cine MRI srdca) [5], MoNuSeg [28], CRAG [28] (histologické/patologické snímky) vid obr. 3.1



Obr. 3.1: Ukážka vzoriek a ich príslušných segmentačných masiek z CRAG a MoNuSeg datasetu

3.2 Metódy segmentácie

Prvé segmentačné metódy boli zamerané predovšetkým na prahovanie (*thresholding*), hranové (*edge-based*) alebo *regiónové* (region-based) metódy ako aktívne kontúry či metódy strojového učenia. V súčastnosti sa však presunul výskum k návrhu sietí hlbokého učenia resp. konvolučných sietí, ktoré na medicínskych modalitách dosahujú vyššiu mieru presnosti a kvality výstupu. [16, 25]

Segmentačné riešenia hlbokých sietí, na úkor vysokej presnosti, vyžadujú pre svoje trénovanie veľké množstvo kvalitne anotovaných dát. Oblasť medicínskeho zobrazovania je však obzvlášť problémová, kde rovnako dáta aj kvalitné anotácie sú nákladné na získanie. Preto pozorujeme vo výskume 2 smery návrhu sietí vzhľadom na učenie resp. predpoklad o dátach a to [47, 61]:

- Riadená segmentácia obrazu (*Supervised image segmentation*)
- Slabo riadená segmentácia obrazu (*Weakly-supervised image segmentation*)

3.2.1 Metódy riadenej segmentácie

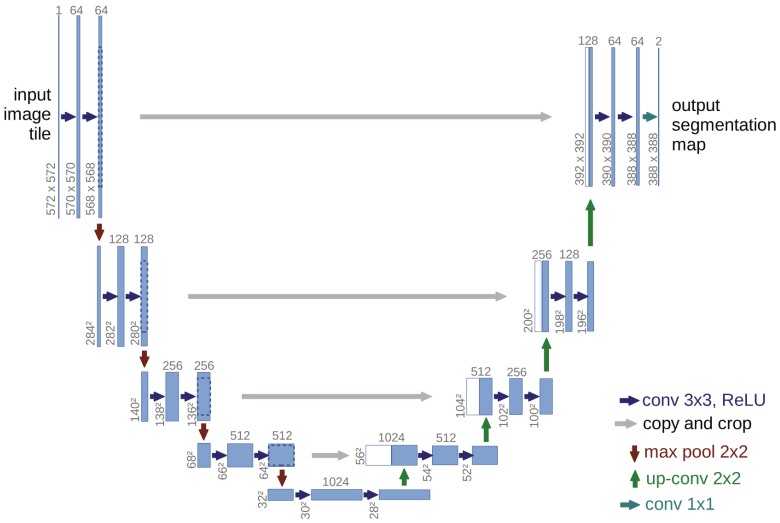
Modely riadenej sémantickej segmentácie pracujú s predpokladom, že všetky trénovacie dáta sú anotované, čo im zabezpečuje kvalitnejšie výsledky. Ich výskum sa v súčastnosti zaoberá predovšetkým zlepšovaním architektúr, ich sieťových blokov a stratových funkcií. [40]

Enkóder-dekóder: Spomedzi existujúcich riešení je väčšina postavená na konvolučných enkóder-dekóder sieťach, vďaka predstaveniu U-Net [49] (obr. 3.2) a V-Net [43] pre 3D modality, využívajúce symetrickú štruktúru a skokové prepojenia. Štúdie sa doteraz zakladajú na ich architektúre s návrhmi rôznych zmien v ich štruktúre ako napr. U-Net++ [76] s dodatočnými skokovými prepojeniami alebo MultiResUNet [20] s reziduálnych blokmi. 3D verzie podobných sietí však

trpia vysokým výpočtovým nákladom kvôli veľkému počtu parametrov. Populárno alternatívou sú spomínané DeepLabV3 modely [9, 8] s atriálnou konvolúciou.

Celkovo je ich cieľom efektívnejšia extrakcia vlastností z enkódera a rekonštrukcia dekóderom pre čo najlepšiu presnosť segmentácie. Na výstupe je typicky hlava 1x1 konvolúcie pre nastavenie hĺbky výstupnej mapy podľa počtu požadovaných tried danej úlohy. *Softmax* a *argmax* funkcia je použitá pre stanovanie výslednej triedy pixelu/voxelu a vytvorenie masky segmentovaných regiónov.

Medicínske transformery: V súčastnosti dosahujú transformer založené siete SOTA výsledky predovšetkým vďaka riešeniu Vision Transformer (ViT) [14]. Tieto riešenia nahradzujú konvolučné operácie za tzv. moduly seba-pozornosti (*self-attention*), ktorými zostavujú celú enkóder-dekóder štruktúru. Oproti plne konvolučným riešeniam majú schopnosť učenia vzdialených závislostí medzi pixelmi, no na úkor výpočtovej zložitosti. Viaceré riešenia sa to snažia minimalizovať ako napr. Swin-UNet [7] použitím swin-transformer blokov s mechanizmom pozornosti a posuvným oknom. Najnovšie riešenia sú rôzne hybridné prístupy ako E-TUNet [30] kombinujúce konvolúciu s transformer blokmi v enkóderi aj dekóderi.



Obr. 3.2: Architektúra U-Net pre segmentáciu biomedicínskych snímok [49]

3.2.2 Metódy slabo riadenej segmentácie

Riešenia slabo riadenej segmentácie pracujú pri návrhu s predpokladom, že počet anotovaných dát je limitovaný resp. iba malá časť je označená a väčšina dát je neoznačená. Preto je v tomto smere snaha navrhovať metódy, ktoré dosahujú kvalitné výsledky segmentácie aj s týmto obmedzením.

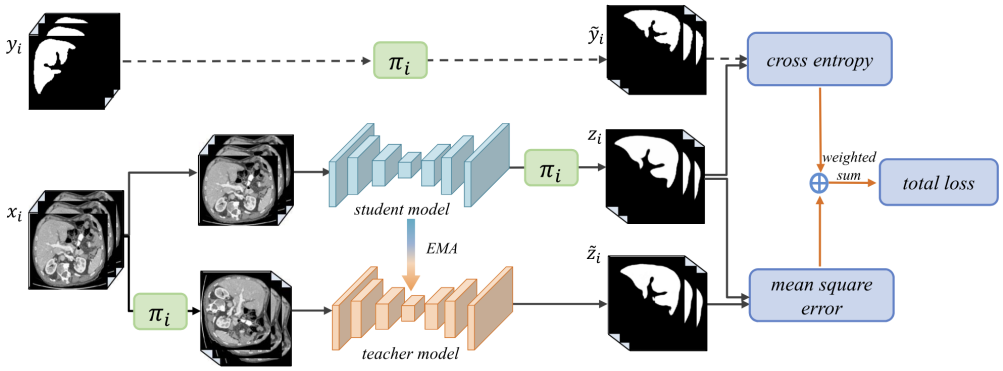
Aktívne učenie (*active learning*) je technika učenia, pri ktorej je model natrénovaný na malej počiatočnej množine anotovaných dát a následne iteratívne vyberá vzorku z množiny neoznačených dát pre anotáciu lekárom a pridanie do trénovanej množiny, až kým nie je dosiahnutá potrebná hranica kvalitne segmentovať. V tomto procese je kľúčový návrh akvizičných funkcií pre ohodnotenie a výber vzoriek umožňujúcich zlepšenie modelov. Snaha pri výbere je chodné stanovenie reprezentatívnosti a neurčitosti vzorky, čo ostáva byť naďalej náročné. Shen et. al. [53] navrhol akvizičné funkcie, ktoré berú do úvahy oba spomenuté meratele. Yan et al. [68] využil dávkový (*batch*) prístup výberu vzoriek pomocou ich skórovacej funkcie, ktorá berie do úvahy 2 rôzne pohľady na 3D snímku.

Čiastočne-riadené učenie (*semi-supervised learning*) a jeho metódy využívajú všetky dostupné ne-anotované snímky pre podporu učenia z limitovaného množstva anotovaných snímok. V tomto smere sa metódy rozlišujú zhruba podľa 2 konceptov a to: *pseudo-anotácie* a *regularizácia konzistencie*.

Pri *pseudo-anotačných metódach* sa najprv trénuje model na limitovanom počte anotovaných dát a následne sa použije na generovanie pseudo-anotačných masiek z neoznačených dát. Vytvorené umelé anotácie sa s ich príslušnými dátami zaradia do označenej sady, a trénovanie je striedavo prepínané medzi týmito fázami. Tieto metódy majú problém s pre-trénovaním na chybných umelých anotáciách, preto sa predovšetkým líšia v tom, ako s nimi zaobchádzajú. V súčasných riešeniach sa skôr

kombinuje koncept umelých anotácií s inými prístupmi ako napr. *cross-teaching* [39], co-training [46] alebo *cross-pseudo-supervision* [10].

Pri koncepte *regularizácie konzistencie* ide o učenie zo všetkých dát v jednotnom rámci, kedy je minimalizovaná strata pre anotované snímky (*supervised loss*) a aj ne-anotované snímky prostredníctvom ďalšej zložky (*consistency loss*) [40]. Tieto metódy využívajú predpoklad, že dva blízke body na vstupe by mali mať rovnaké anotácie. Na základe toho využívajú ne-anotované údaje tým, že aplikujú na ne tzv. *perturbácie* (poruchy, augmentácie, tranformácie,..) a trénujú model alebo jeho časť, ktorá nie je nimi ovplyvnená. To sa dosahuje pridaním regularizácie v podobe konzistentnej straty, ktorá meria vzdialenosť medzi pôvodnými a perturbovanými predikciami. Ich hlavným predstaviteľom sú Mean Teacher [57] modely a jej vylepšenia [24], pri ktorých sa udržuje konzistencia v predikciách medzi učiteľ a študent modelom. Riešenia sa navzájom často odlišujú spôsobom akým aplikujú perturbácie: vrámci vstupných dát, architektúry alebo výstupných predikcií. Obr. 3.3 znázorňuje Mean Teacher model od Li et al. [33] s konzistentnými augmentáciami v rámci vstupu učiteľ modelu a výstupu študent modelu.



Obr. 3.3: Transformačne konzistentný Mean Teacher model od Li et. al [33]

Okrem spomenutých pozorujeme riešenia využívajúce súperiace (*adversarial*) učenie [74] alebo kontrastívne učenie. [27]

4. Analýza súvisiacich prác

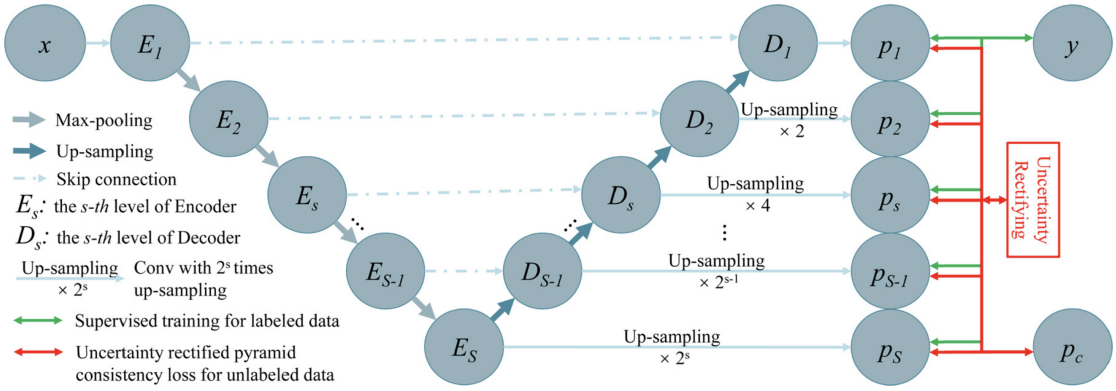
V tejto kapitole bližšie analyzujeme 3 vybrané SOTA riešenia čiastočne riadenej segmentácie medicínskych snímok. Zamerali sme sa na riešenia, ktoré rôzne pristupujú k regularizácii konzistencie, prípadne spolu s využitím konceptu pseudo-anotácií a to: *URPC* [40], *Cross-teaching CNN&Transformer* [39] a *Cross-distilation* [75]. Pri každom rozoberáme navrhnutý koncept učenia, použitú architektúru, dataset a nakoniec vykonané experimenty s ich výsledkami.

4.1 Uncertainty Rectified Pyramid Consistency

Autori Luo et al. vybraného článku [40] predstavili vlastný prístup k regularizácii konzistencie na úrovni architektúry a výstupu siete, nazývaný *Uncertainty Rectified Pyramid Consistency* (URPC) pre čiastočne riadenú segmentáciu 3D snímok.

Základom prístupu je pyramídová enkóder-dekóder sieť (viď obr. 4.1), ktorá vytvára pre anotované aj ne-anotované snímky množinu segmentačných predikcií (máp) na viacerých úrovniach (škálach) dekódera. Pre potreby učenia siete, sú tieto predikcie *upsampling* vrstvami škálované do pôvodných rozmerov vstupu.

URPC na využitie ne-anotovaných snímok pracuje s predpokladom, že predikcie rovnakého objektu/regiónu na rôznych úrovniach, by mali byť navzájom blízke. Pre tento účel autori zavádzajú tzv. *pyramídovú konzistenciu*, ktorá minimalizuje



Obr. 4.1: Prehľad prístupu URPC [40] z pohľadu architektúry.

rozdielnosť (varianciu) medzi predikčnými mapami danej úrovne a priemernou predikciou všetkých úrovní dekóderov. Vzhľadom na nedostatok anotácií a chybám, ktoré môžu nastať pri *upsampling-u* predikčných máp, môže byť ich konzistencia regularizovaná smerom k vychýleným hodnotám (outlierom). Preto k pyramídovej konzistencii pridávajú korekčnú zložku tzv. *opravu neistoty*. Okrem toho, prostredníctvom *minimalizácie neistoty*, ako ďalšej zložky učenia, dodatočne podporujú robustnosť modelu a confidenciu výsledných predikcií.

4.1.1 Nastavenie stratových funkcií

Pre štandardný anotovaný snímok x, sieť produkuje množinu viac-úrovňových pravdepodobnostných máp $[p_1, p_2, ..., p_s]$ škálovaných do rovnakých rozmerov vstupu x. Nad týmito mapami je sieť učená pod dohľadom cez stratu v tvare:

$$\mathcal{L}_{sup} = \sum_{s=1}^{S} \alpha_s \mathcal{L}(p_s, y) \tag{4.1}$$

Kde S je počet úrovní/škál dekódera a $\mathcal{L}$ stratová funkcia Dice koeficientu.

Z pohľadu ne-anotovaného snímku, sieť rovnako produkuje sadu pravdepodobnostných máp pre každú úroveň a s kanálom pre každú triedu. Pre ďalšie zložky stratovej funkcie sú vytvorené tzv. *mapy neistoty* $\mathcal{D}$ každej úrovne teda $[\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_S]$,

definované ako:

$$\mathcal{D}_s^i \approx \mathbf{KL}(p_s^i, p_{avg}^i) \tag{4.2}$$

Kde $\mathcal{D}_i^s$ je neistota pixelu/voxelu i na úrovni s a p_{avg}^i je element priemernej pravdepodobnostnej mapy zo všetkých úrovní definovanej ako: $p_{avg} = \frac{1}{S}\sum_{s=1}^{s} p_s$. Funkcia $\mathbf{KL}()$ je potom Kullbackova–Leiblerova divergencia, ktorá meria odlišnosť 2 pravdep. distribúcií. Pre interpretáciu, veľká hodnota $\mathcal{D}_s^i$ indikuje, že predikcia pre pixel/voxel i v danej škále s je veľmi rozdielna od ostatných škál, teda *"neistá"*. Stratovú funkciu *minimalizácie neistoty* potom definujú ako:

$$\mathcal{L}_{umc} = \frac{1}{S}\sum_{s=1}^{S} \mathcal{D}_s \tag{4.3}$$

V prípade *pyramídovej konzistencie* je snaha udržiavať konzistentné predikcie naprieč škálam pomocou L_2 straty, ktorú prispôsobili do tvaru:

$$\mathcal{L}_{urc} = \frac{1}{S}\sum_{s=1}^{S} \frac{\sum_i ||p_s^i - p_{avg}^i||_2 \cdot w_s^i}{\sum_i w_s^i} \tag{4.4}$$

Kde je w_s^i je spomínaná *oprava neistoty* definovaná ako $w_s^i = e^{-\mathcal{D}_s^i}$, ktorá potláča stratovú funkciu pre outlier pixely s výrazne odlišnými predikciami než priemer a sústreďuje stratu na spoľahlivé pixely. Celková strata pre ne-anotované snímky je potom kombináciou *minimalizácie neistoty* a *pyramídovej konzistencie*:

$$\mathcal{L}_{unsup} = \beta\mathcal{L}_{urc} + (1-\beta)\mathcal{L}_{umc} \tag{4.5}$$

Kde β je váha zložiek. Takto nastavená strata pomáha produkcii konzistentnejších predikcií naprieč úrovňami dekodéra. Celková stratová funkcia je potom:

$$\mathcal{L}_{total} = \mathcal{L}_{sup} + \lambda\mathcal{L}_{unsup} \tag{4.6}$$

Kde λ tzv. gausova zohrievacia funkcia, ktorá počas tréningu postupne navyšuje váhu zložky pre ne-anotované dáta.

4.1.2 Datasety

Návrh siete s URPC bol evaluovaný na 1 vlastnom a 2 verejných datasetoch:

- NPC-MRI dataset - pozostáva z MRI skenov 258 pacientov s NPC (nádorom nosohltana a lymfatických uzlín), ktorých prípady boli anotované onkológom. Dataset bol rozdelený na 180, 20, 58 prípadov pre trénovanie, validáciu a testovanie. Autormi bol normalizovaný na priemer 0 a varianciu 1.
- BraTS2019 (Menze et al. 2014) [41] - obsahuje anotované snímky nádorov mozgu z viacerých modalít. Autori z nich prevzali sadu FLAIR s 335 skenmi izometrického rozlíšenia $1mm^3$. Skeny boli náhodne rozdelené do 250, 25, 60 skenov pre trénovanie, validáciu, testovanie a škálované do rozsahu [0,1].
- Pancreas-NIH (Roth et al. 2015) [51] - obsahuje 82 abdominálnych CT snímok s anotáciami pankreasu a rozdelením 50 pre trénovanie, 12 validáciu a 20 pre testovanie. Autori Luo et al. škálovali snímky na rozlíšenie $1mm^3$ a rozsah [0, 1] s výrezmi pre vy-centrovanie pankreasu.

Ako augmentačné techniky boli vykonávané náhodné výrezy 112x112x112px pre NPC-MRI a 96x96x96px pri ostatných datasetoch. Počas tréningu bolo vykonávané náhodné prevracanie a náhodné rotácie.

4.1.3 Architektúra a nastavenie trénovania

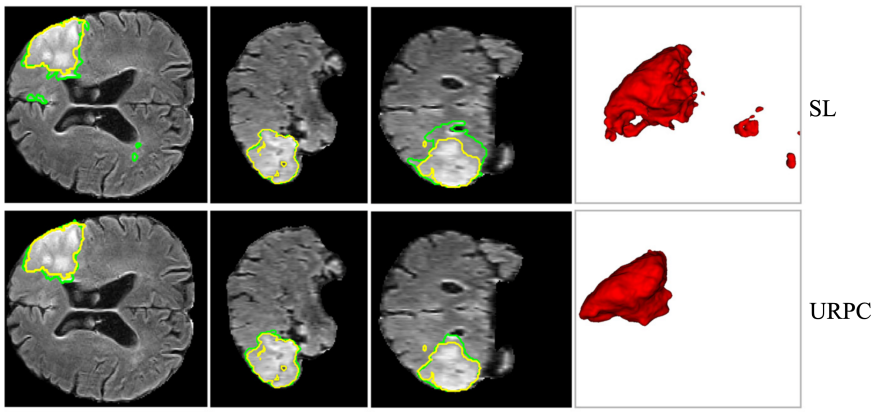
Základom siete je 3D-UNet (2016) [11], ktorú Luo et al. upravili pre pyramídové predikcie prostredníctvom pomocných segmentačných hláv za každým upsampling blokom dekódera. Pomocné hlavy obsahujú 1x1 konvolúciu a softmax.

Experimenty boli vykonávané na Ubuntu PC s GTX1080TI GPU. Pre trénovanie bol zvolený SGD optimizátor (weight decay $=1e^4$, m = 0.9) a predstavená stratová funkcia s hodnotami $\alpha_s = 1$, $\beta = 0.5$. Mieru učenia postupne znižovali z 0.1

stratégiou *poly learning rate*. Pre čiastočne riadené učenie nastavili trénovaciu sadu NPC datasetu na 18 anotovaných a 162 ne-anotovaných vzoriek, Pancreas-NIH datasetu na 40 anotovaných, 10 ne-anotovaných vzoriek a v podobnom pomere pre BraTS. Veľkosť dávok bola 4, s polovicou ne-anotovaných vzoriek. Počas inferencie je výsledná segmentačná maska zostavená stratégiou posuvného okna.

4.1.4 Experimenty

Návrh bol vyhodnocovaný metrikami Dice koeficient (DSC↑) a 95% Hausdorff Distance (HD_{95} ↓) a Average Surface Distance (ASD ↑). Na overenie riešenia využili NPC a Pancreas-NIH, s ktorými testovali rôzne počty úrovní (S = 1 až 5) a kombinácie zložiek stratovej funkcie. Najlepšie v metrikách bolo predstavené nastavenie s počtom škál 4. V rámci SOTA sa porovnali s 5 súčasnými čiastočne riadenými riešeniami na všetkých datasetoch (pri 10% alebo 20% anotáciách), kde dosiahli najlepšie výsledky na DSC a porovnateľné výsledky v ostatných metrikách. Napríklad pri 10% anotáciach Pancreas-NIH dosiahli 74.89 DSC v porovnaní s populárnym Mean Teacher [57] riešením s 70.89 DSC. Pre demonštráciu schopnosti URPC využiť ne-anotované snímky v učení, obr. 4.2 znázorňuje výsledky segmentácie plne riadenej siete v porovnaní s čiastočným riadením (obe s 10% anotáciami).

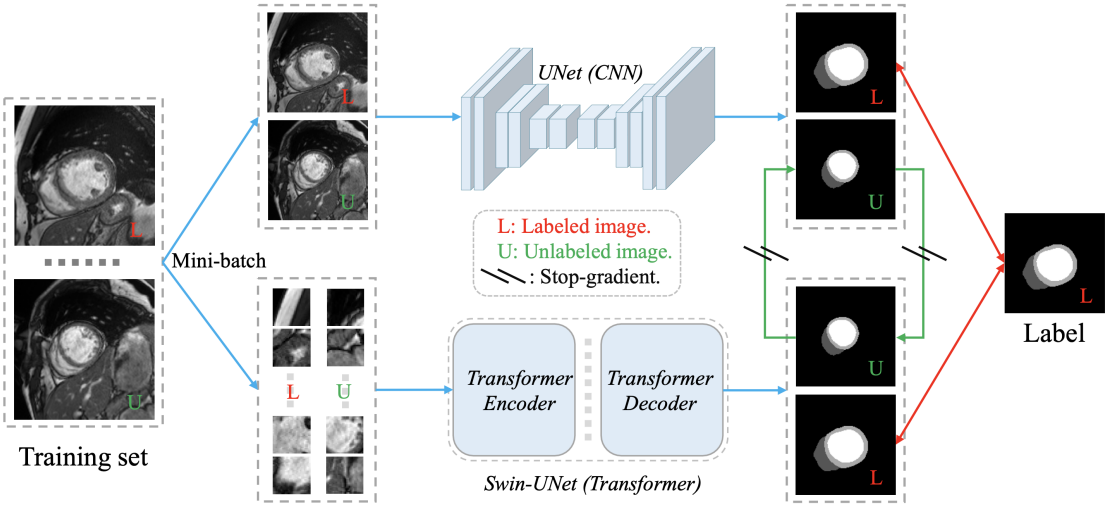


Obr. 4.2: Plné riadenie (1. rad - SL) vs čiastočné riadenie (2. rad - URPC) na BraTS2019 [41] z rôznych pohľadov - zelená=predikovaná, žltá=očakávaná maska.

4.2 Cross Teaching CNN & Transformer

Rovnaký autori Luo et al. predstavili v tomto článku [39] z oblasti čiastočne-riadených metód segmentácie, novú stratégiu krížového učenia 2 sietí tzv. **cross-teaching**. Ide o učiace sa nastavenie, ktoré zjednodušuje co-training prístup regularizácie konzistencie a kombinuje ho s využitím pseudo-anotácií na učenie druhej siete super-vised spôsobom (pod dohľadom).

Viac konkrétne, ide o efektívnu regularizačnú schému medzi **CNN a Transformer** enkóder-dekodér sieťou (viď obr. 4.3), ktorá využíva anotované aj neanotované snímky. V prípade anotovaných snímok, je vstupný snímok spracovaný rovnako CNN aj Transformer sieťou, pričom ich parametre sú optimalizované klasickým učením pod dohľadom, príslušnými anotačnými maskami. V prípade neanotovaných snímok, sú predikcie CNN aj Transformer siete použité ako pseudo-anotácie na vzájomnú aktualizáciu parametrov pod dohľadom druhej siete.



Obr. 4.3: Prehľad architektúry frameworku krížového učenia medzi CNN a Transformerom [39]

Oproti ostatným riešeniam, kombinuje predstavená schéma 2 paradigmy - CNN

siete, ktoré sa zameriavajú na lokálne príznaky snímok a Transformery, ktoré veľmi dobre reprezentujú rozsiahle vlastnosti naprieč sekvencii snímok. Spolu s navrhnutou stratégiou je tak implicitne podporovaná konzistencia oboch sietí, ktoré sa vzájomne svojimi výhodami kompenzujú pre lepší segmentačný výkon. Autori tvrdia, že takéto riešenie produkuje navyše stabilnejšie a presnejšie pseudo-anotácie.

4.2.1 Krížové učenie a nastavenie stratových funkcií

Predstavená stratégia je inšpirovaná prístupmi: *co-training* [46] a *cross pseudo supervision* [10], kde ide taktiež o vzájomné učenie sietí ale s odlišnými perturbáciami na úrovni vstupných dát (*views*) alebo inicializácie parametrov architektúry.

V prípade krížového učenia sú pre ne-anotovaný snímok výstupom 2 predikcie: softmax predikcia CNN siete $p_i^c = f_\phi^c(x_i)$ a softmax predikcia Transformera $p_i^t = f_\phi^t(x_i)$. Vzájomná pseudo-anotácia pre dohľad nad CNN sieťou pl_i^c a pseudo-anotácia pre dohľad nad Transformerom pl_i^t sú potom definované ako:

$$pl_i^c = argmax(p_i^t); pl_i^t = argmax(p_i^c) \tag{4.7}$$

Pre pseudo-anotácie pl_i^c ani pl_i^c nie je žiadny spätný prechod gradientov cez siete, ktoré ich vytvorili. Namiesto toho je definovaná strata krížového učenia v tvare:

$$\mathcal{L}_{ctl} = \underbrace{\mathcal{L}_{dice}(p_i^c, pl_i^c)}_{\text{dohľad pre CNN}} + \underbrace{\mathcal{L}_{dice}(p_t^c, pl_t^c)}_{\text{dohľad pre Transformer}} \tag{4.8}$$

Kde $\mathcal{L}_{dice}$ je stratová funkcia Dice koeficientom. Pre anotované snímky je stratová funkcia oboch sietí definovaná ako:

$$\mathcal{L}_{sup} = \mathcal{L}_{ce}(p_i, y_i) + \mathcal{L}_{dice}(p_i, y_i) \tag{4.9}$$

teda ako kombinácia krížovej entropie $\mathcal{L}_{ce}$ a dice koeficient straty $\mathcal{L}_{dice}$, kde p_i je predikcia siete a y_i jej anotačná maska. Toto nastavenie je tak možné trénovať naraz

na všetkých dostupných snímkach s celkovou stratovou funkciou v podobe:

$$\mathcal{L}_{total} = \mathcal{L}_{sub} + \lambda \mathcal{L}_{ctl} \tag{4.10}$$

Parameter λ (*gausova zohrievacia funkcia*) v každom epochu navyšuje zložku krížového učenia, aby z počiatku neboli modely zahltené regularizáciou, ak nie sú dostatočne naučené z anotovaných snímok. Výsledná loss je potom použitá na aktualizáciu parametrov oboch sietí.

4.2.2 Dataset

Autori článku použili ACDC dataset obsahujúci 200 anotovaných cine-MRI snímok srdca (100 pacientov) so segmentačnými maskami myokardu (srdcovej svaloviny), ľavej a pravej komory. Rozdelených bolo 140 na 60 snímok pre trénovaciu a validačnú množinu.

Z pohľadu efektivity, obe siete pracovali namiesto celých 3D snímok s 2D rezmi, ktoré predspracovaním zmenšovali na veľkosť 256x256px. V rámci augmentácií boli 2D rezy škálované do rozsahu [0,1] a pre navýšenie sady a predchádzanie pre-trénovania boli vykonávané náhodné výrezy veľkosti 224x224px, náhodné prevracanie a náhodné rotácie medzi -25 a 25 stupňov. Pre výslednú inferenciu bola zvolená jedna zo sietí a jej predikcie z rezov skladané do celých 3D objemov.

4.2.3 Architektúra a nastavenie trénovania

Základom CNN a Transformer segmentačných sietí bola zvolená originálna U-Net [49] a populárna Swin-UNet [7] architektúra v podobe ich open-source implementácií. Pre všetky ostatné koncepty bola použitá knižnica PyTorch.

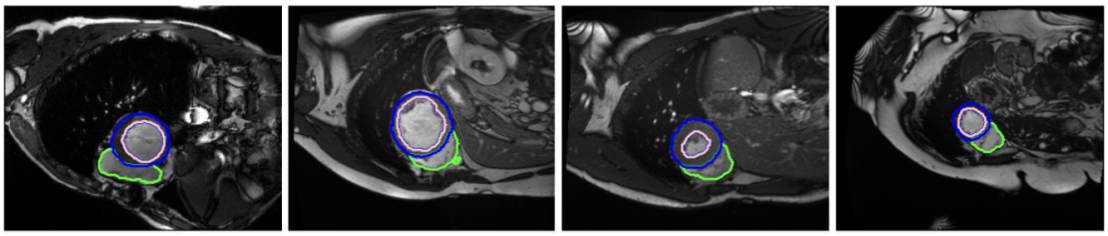
Všetky experimenty bežali na Ubuntu PC s GTX1080TI GPU. Celá schéma sa trénovala SGD optimizátorom s počiatočnou mierou učenia 0.01, postupne znižovanou stratégiou *poly learning rate*.

Pre demonštráciu čiastočne-riadeného učenia bolo náhodne zvolených 6 skenov (3 pacienti) alebo 14 skenov (7 pacientov) ako anotované vzorky (cca. 5% alebo 10% trénovanej sady). Zvyšok (134 alebo 126 skenov) tvorili ne-anotované vzorky. Veľkosť dávok bola 16, s polovicou anotovaných snímok.

4.2.4 Experimenty

Pre demonštráciu výkonu, autori porovnali výsledky riešenia s 8 nedanými čiastočne-riadenými prístupmi segmentácie. Z pohľadu kvantitatívnych metrík, vyhodnocovali Dice koeficient (DSC↑) a 95% Hausdorff Distance (HD_{95} ↓), na ktorých dosiahli najlepšie výsledky a to 0.656 DSC a 16.2 HD_{95} pri 5% anotáciách a 0.864 DSC a 8.60 HD_{95} pri 10% anotáciách. Pre porovnanie, populárne riešenie Mean Teacher [57] dosahovalo 0.810 DSC, 14.4 HD_{95} (na 10% anotáciách). Súčasné plne riadené riešenie nn-U-Net na rovnakom datasete 0.925 DSC a 7.9 HD_{95}.

Mimo SOTA, experimentovali s výberom stratových funkcií, kombináciami sietí (CNN&CNN, Transformery&Transformer,..) alebo s výberom siete pre inferenciu. CNN aj Transformer mali veľmi podobné výsledky, no inferenčný čas a výpočtová zložitosť Transfomera je značne väčšia (27.12M vs 1.81M parametrov), preto pri vyhodnocovaní používali U-Net. Kvalitatívne výsledky riešenia pri 5% anotáciách (2 obr. zľava) a pri 10% anotáciach (2 obr. zprava) zobrazuje obr. 4.4.

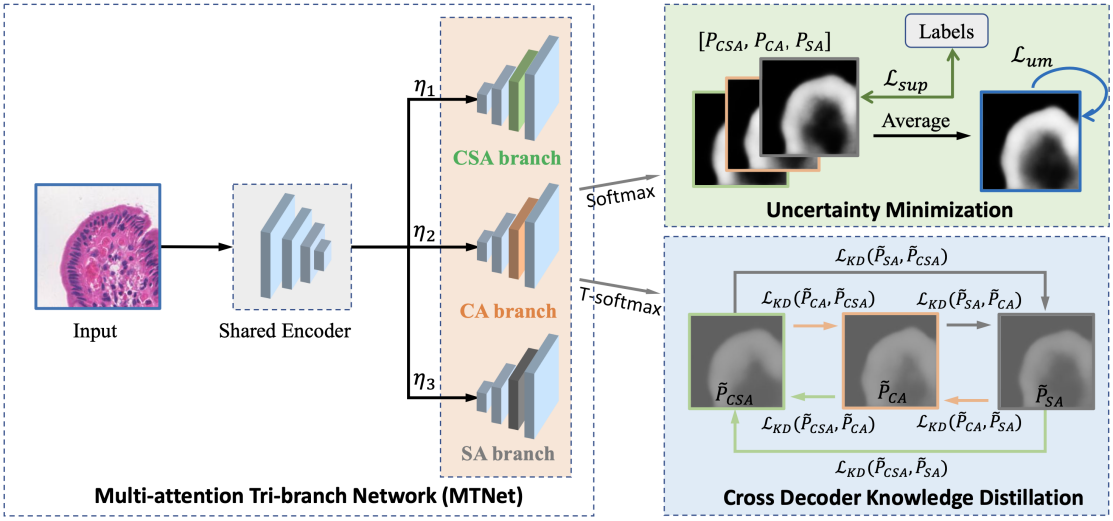


Obr. 4.4: Výsledky riešenia s predikovanými segmentačnými mapami pre 4 vybrané rezy ACDC datasetu.

4.3 Cross Distillation of Multiple Attentions

Autori tohto článku [75] navrhli nový, čiastočne riadený prístup segmentácie patologických snímok. Prvým z príspevkov je segmentačná sieť MTNet (*Multi-attention Tri-branch Network*) zložená z enkódera a 3-vetvového dekódera, kde každá vetva (dekóder) využíva rozdielny mechanizmus pozornosti (*attention*) na získanie rozdielnych ale navzájom komplementárnych predikcií.

Druhým príspevkom je stratégia učenia tejto siete tzv. *cross decoder knowledge distilation* (CDKD) na zapojenie predikcií z ne-anotovaných snímok do učenia (viď. obr. 4.5). V CDKD, každá vetva siete slúži ako učiteľ ostatných vetiev na základe dohľadu nad ich predikciami. Ide tak o vzájomné učenie dekóderov pomocou vlastných pseudo-anotácií, získaných pre vstupný snímok v jednej doprednej fáze. Oproti iným riešeniam, sú vytvárané mäkké (*T-softmax*) anotácie, čo znižuje vplyv šumu na učenie, ktorý môže spôsobovať nepresné *argmax* pseudo-anotácie.



Obr. 4.5: Prehľad architektúry článku krížovej destilácie viacerých pozorností [75]

Posledným príspevkom, ktorý dopĺňa CDKD, je vlastná verzia *minimalizácie neistoty* ako ďalšia zložka učenia. Jej účelom je regularizovať priemernú predikciu

naprieč dekódermi, čo podporuje ich vzájomnú konzistenciu v predikciách.

4.3.1 Architektúra MTNet

Autori postavili enkóder siete MTNet pomocou extraktora ResNet50 predtrénovaným na ImageNet. Dekóder siete implementovali rozšírením DeepLabv3+ [8] modelu do 3-vetvového dekódera, kde konkrétna vetva obsahuje odlišný mechanizmus pozornosti: kanálovú (CA), priestorovú (SA) alebo kombináciu oboch (CSA) v konvolučných blokoch na každej úrovni rozlíšenia. Rôzne mechanizmy pozornosti venujú pozornosť rôznym aspektom máp príznakov, čo vedie k rôznym výstupom vetiev. CA vetva s blokmi kanálovej pozornosti zvýrazňuje dôležité kanály, SA vetva sa zameriava na najviac relevantné priestorové regióny a potláča tie irelevantné, pričom CSA vetva kombinuje vlastnosti oboch mechanizmov.

4.3.2 Destilácia znalostí dekóderov a minimalizácia neistoty

Navrhnutá CDKD stratégia eliminuje vplyv zašumených pseudo-anotácií tak, že ich vytvára teplotne kalibrovaným softmax-om (T-Softmax) definovaným ako:

$$\tilde{\mathbf{p}}_c = \frac{exp(\mathbf{z}_c/T)}{\sum_c exp(\mathbf{z}_c/T)} \tag{4.11}$$

kde $\mathbf{z}_c$ je logit predikcia triedy c pre pixel, T je parameter pre kontrolu sily pravdepodobnosti a $\tilde{\mathbf{p}}_c$ je potom mäkká pravdep. predikcia triedy c. Ak $\tilde{P}_{CSA}, \tilde{P}_{SA}, \tilde{P}_{CA}$ sú pre danú snímku jej výstupné T-softmax predikčné mapy vetiev, tak potom je CSA vetva učená pod dohľadom ostatných vetiev pomocou KL divergencie:

$$\mathcal{L}_{kd}^{CSA} = \mathbf{KL}(\tilde{P}_{CSA}, \tilde{P}_{CA}) + \mathbf{KL}(\tilde{P}_{CSA}, \tilde{P}_{SA}) \tag{4.12}$$

Gradient je v prípade $\mathcal{L}_{kd}^{CSA}$ spätne šírený len pre CSA vetvu aby sa znalosti destilovali od učiteľ vetiev do študent vetvy. Obdobne je pod dohľadom učená SA a

CA vetva, kde si vymenia role. Výsledná strata pre CDKD je potom:

$$\mathcal{L}_{cdkd} = \frac{1}{3}(\mathcal{L}_{kd}^{CSA} + \mathcal{L}_{kd}^{CA} + \mathcal{L}_{kd}^{SA}) \tag{4.13}$$

V existujúcich riešeniach dodatočná zložka *minimalizácie neistoty* efektívne podporuje konfidenciu predikcií ne-anotovaných snímok. Aplikovaním jednotlivo na každú vetvu by však predikcie vetiev mohli byť vzájomne príliš nekonzistentné preto autori definujú vlastnú verziu v podobe:

$$\mathcal{L}_{um} = -\frac{1}{N}\sum_{i=0}^{N}\sum_{c=0}^{C}\bar{P}_i^c log(\bar{P}_i^c) \tag{4.14}$$

Kde C je číslo triedy, N je index pixelu a $\bar{P} = (P_{CSA} + P_{SA} + P_{CA})/3$ je priemerná pravdepodobnostná mapa vetiev. Pre príklad ak $\mathcal{L}_{um}$ daného pixelu je blízko 1, tak potom aj priemerná predikcia vetiev pre pixel je 1 čo ich núti byť navzájom podobné. Výsledná strata pre učenie MTNet siete je potom:

$$\mathcal{L} = \mathcal{L}_{sup} + \lambda_1\mathcal{L}_{cdkd} + \lambda_2\mathcal{L}_{um} \tag{4.15}$$

Pre anotované snímky je štandardná $\mathcal{L}_{sup}$ strata definovaná ako priemer krížovej entropie a Dice koeficientu naprieč vetvami a λ sú váhy zložiek. V tomto nastavení je $\mathcal{L}_{cdkd}$ aj $\mathcal{L}_{um}$ použitá pre anotované aj ne-anotované snímky.

4.3.3 Dataset a nastavenie trénovania

Na experimenty a evaluáciu MTNet s navrhnutou stratégiou bol použitý Digest-Path [12] dataset obsahujúci 130 WSI snímok kolonoskopických nádorových lézií s binárnymi anotačnými mapami. WSI snímky (5000x5000px), boli náhodne rozdelené do trénovanej, validačnej a testovacej množiny po 100, 10 a 20 snímok.

Ako aj podobné riešenia, náhodne bolo vybraných 5% alebo 10% anotácií pre trénovaciu množinu. V rámci predspracovania boli pre výpočtovú efektivitu vytvárané náhodné výrezy 256x256px. Počas inferencie je pre enkóder vybraný jeden

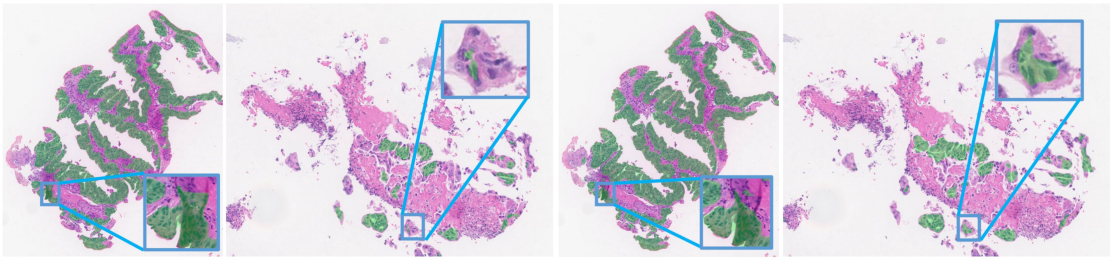
z dekodérov a celá WSI spracovaná oknom 256x256px a posunom 192x192px. Ako augmentačné techniky zvolili náhodné rotácie, prevracanie a Gaussov šum.

Riešenie je implementované v PyTorch a trénované na NVIDIA 2080TI GPU. Ako optimizátor zvolili SGD s počiatočnou mierou učenia 0.001, weight decay = 0.0005 a m = 0.9. Veľkosť dávky bola 16 s polovicou ne-anotovaných výrezov. Parametre $\lambda_1 = \lambda_0 = 0.1, T = 10$ nastavili podľa najlepších validačných výsledkov.

4.3.4 Experimenty

V rámci overenia testovali rôzne nastavenia vetiev (2 vetvy, bez pozornostných blokov, s rovnakými blokmi a pod.), nastavenia stratových funkcií alebo podobu predikcií z pohľadu T-softmaxu. Pre výslednú inferenciu bola zvolená CSA vetva.

Riešenie bolo evaluované na metrikách Dice koeficient (DSC↑) a Jaccard Index (JI↑), pri ktorých dosiahlo 69.72 DSC a 57.09 JI s 5% anotáciami a 72.24 DSC a 60.17 JI s 10% anotáciami. Výsledky porovnali s 8 čiastočne riadenými prístupmi vrátané analyzovaného CNN%Transformer-a, ktorý dosiahol 67.66 DSC a 55.74 JI pri 5% anotáciách. S rovnakým počtom anotácií testovali plné riadenú MTNet čo dosiahlo 64.74 DSC, čo značí schopnosť učenia návrhu aj z ne-anotovaných snímok. Obr. 4.6 kvantitatívne porovnáva toto nastavenie oproti CDKD stratégii.



Obr. 4.6: Porovnanie výsledkov segmentácie pri 5% všetkých anotovaných dát: SL - plne riadená MTNet, CDKD - čiastočné riadená MTNet s využitím dát bez anotácií [75]. Zelené regióny = predikované oblasti lézie

.

4.3.5 Zhrnutie riešení a možné vylepšenia

Na základe analýzy a výsledkov vybraných riešení sme dospeli k potvrdeniu, že efektívne využitie veľkého množstva dát bez anotácií pre podporu celkovej kvality výsledkov segmentačných metód, je stále otvoreným problémom, ktorý sa oplatí ďalej skúmať. Aktuálne riešenia tohto problému sa segmentačným výkonom a efektivitou sietí naďalej nedoťahujú súčasným plne riadeným prístupom s predpokladom všetkých anotácií. Preto sa v rámci praktickej časti ďalej venujeme návrhom vlastnej metódy položenej na základe SOTA čiastočne riadenej segmentácie.

Pri bližšie analyzovaných riešeniach je potenciál ďalšieho experimentovania. U všetkých 3 návrhov je možnosť vylepšenia architektúr za aktuálnejšie koncepty vyspelých CNN sietí ako napr. EfficientNet [56] alebo ConvNext [36]. To sa týka najmä modelu CNN&Transformer-a [39] s pomerne už dnes staršou U-Net.

V rámci priebežného návrhu sa nám ponúka zapojiť koncept pyramídovej konzistencie [40] alebo CDKD [75] viacerých dekodérov do krížoveho učenia [39] alebo Mean Teacher [57] založených sietí, kde rovnako študent aj učiteľ sieť by bol regularizovaný naprieč škálami alebo pomocnými dekódermi. Podobný prístup môžme sledovať od Jin et al. [24] kde sú predikcie regularizované naprieč dodatočnými segmentačnými hlavami, ktoré naopak vychádzajú z enkódera študent siete.

Výhodou MTNet [75] a pyramídovej-siete pre URPC [40] je z pohľadu výpočtovej zložitosti, schopnosť učenia v doprednej fáze cez jednu sieť, oproti prístupom so spracovaním druhou sieťou. Podľa výsledkov MTNet sme spozorovali nie moc značné zlepšenie medzi 2 a 3 dekódermi, preto je možnosť experimentovať zjednodušením siete na CA vetvu a SA vetvu a stratený výkon vykompenzovať zavedením pyramídovej konzistencie. Inou možnosťou je namiesto pozornosti regularizovať transformačnú konzistenciu (ako Li et al. [33]) ale naprieč 2 dekódermi.

5. Návrh

Na základe analýzy súčasných riešení segmentácie medicínskych modalít sme navrhli hybridný prístup čiastočne riadenej segmentácie 2D medicínskych snímok v podobe vlastného modelu **DBPNet** (*Dual Branch Pyramid Network*) a úprave učenia založenom na regularizácii konzistencie na úrovni architektúry. S týmto učením sme experimentovali a prispôsobovali modelu v podobe úprav a rozšírení stratovej funkcie pre zlepšenie segmentačného výkonu. Použité existujúce koncepty učenia sme rozšírili o vlastne navrhnutú zložku, ktorá za pomoci anotovaných aj ne-anotovaných dát regularizuje konzistenciu medzi pomocnými (pyramídovými) škálami naprieč 2 dekódermi, čo dodatočne zvyšuje výkon segmentácie.

V tejto časti bližšie rozoberieme návrh učenia (kap. 5.1), detaily architektúry (kap. 5.2), počiatočný návrh stratovej funkcie (kap. 5.3), metodológiu experimentov vrátane detailov ich nastavenia, použitého datasetu a metrík (kap. 5.4) a nakoniec samotný proces overenia a experimentovania s riešením (kap. 5.5 až 5.8).

5.1 Návrh učenia regularizáciou konzistencie

Navrhnutá DBPNet je konvolučný auto-enkóder so zdieľaným enkóderom a 2 dekódermi, ktorý je nastavením učenia hybridom medzi 2 súčasnými prístupmi:

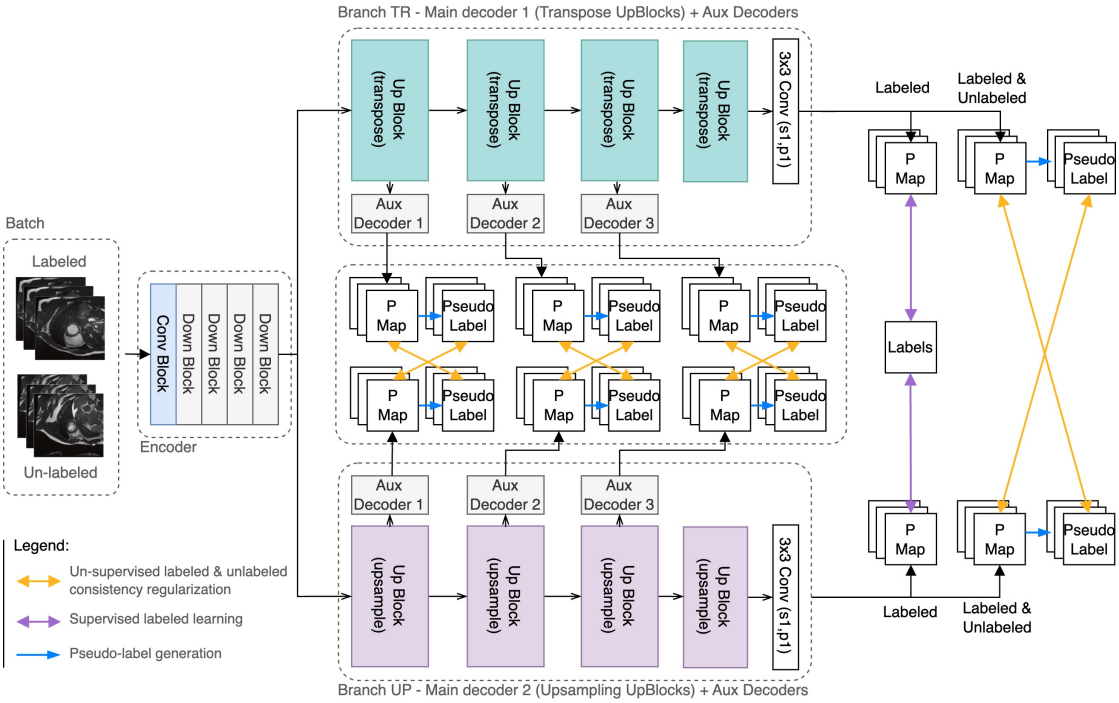
Prvým je riešenie MCNet+ [65], ktoré podobne ako analyzovaný MTNet (kap.

4.3), využíva spoločný enkóder a 3 rozdielne dekodéry, ktoré sa dodatočne učia zo všetkých dostupných dát regularizáciou konzistencie medzi vzájomnými predikciami. Predikcie (výstupy) dekodérov si navzájom slúžia ako pseudo-anotácie. Ideou je, že meranie štatistickej nezrovnalosti (ang. *discrepancy*) alebo rozdielnosti výstupov viacerých dekóderov, hovorí o neistote modelu v predikciách a indikuje ťažké regióny predovšetkým na ne-anotovaných snímkach. Aplikáciou konzistentného obmedzenia v podobe zložky stratovej funkcie medzi pravdepodobnostným výstupom jedného dekódera a výstupmi (pseudo-anotáciami) ostatných dekóderov, je možné minimalizovať ich nezrovnalosti (neistotu modelu), čo má efekt regularizácie tréningu a generalizácie modelu aj za pomoci ne-anotovaných dát. [65]

Druhým riešením je analyzovaný URPC [40] model (kapitola 4.1) s jedným enkóderom a jedným dekóderom, ktorý pristupuje k regularizácii konzistencie pomocou pyramídových predikcií na viacerých škálach dekódera. URPC model sa učí z ne-anotovaných dát minimalizáciou nezrovnalostí medzi pyramídovými predikciami a ich priemerom, čo má podobný efekt generalizácie modelu.

Naš navrhnutý model a nastavenie čiastočne riadeného učenia kombinuje koncepty oboch prístupov (viď vizualizáciu na obr. 5.1). Na rozdiel on MCNet+ [65] a MTNet [75] sme zjednodušili sieť na 2 hlavné dekódery resp. vetvy, pretože z výsledkov oboch prác sme usúdili, že pridanie tretieho nebolo až tak značné k celkovému výkonu. Obom hlavným dekodérom sme od URPC [40] pridali na každej škále pomocnú segmentačnú hlavu (*aux decoder*), pre dodatočné pyramídové predikcie. V jednej doprednej fáze tréningu je spracovávaný batch všetkých (anotovaných aj ne-anotovaných) dát. Z týchto dát je model učený nielen regularizáciou konzistencie medzi hlavnými výstupmi (ako pri MTNet, MCNet+) ale aj medzi dvojicami pyramídových predikcií naprieč prislúchajúcimi škálami dekóderov (viď. žlté šípky na obr. 5.1). Túto doplňujúcu zložku sme navrhli s ideou, že neistota

podľa nás nemusí byť modelovaná len v podobe hlavných výstupov dekóderov, ale aj priebežne v ich skrytých vrstvách nižšieho rozlíšenia, ktoré obsahujú príznaky a reprezentácie vyššej úrovne. Narozdiel od URPC, je myšlienkou podporiť minimalizáciu neistoty nie samostatne naprieč škálami rovnakého dekódera, ale naprieč susednými škálami 2 dekóderov pre dodatočnú podporu ich učenia sa navzájom. Pod dohľadom skutočných anotácií sú učené oba dekódery samostatne voči ich hlavných výstupom anotovaných dát (viď. fialové šípky na obr. 5.1). Počas inferencie je použitá len sieť s jednou vetvou a jeho hlavným výstupom, vďaka čomu nenarastá počet parametrov a zostáva rovnaký ako pri štandardnej U-Net.



Obr. 5.1: Návrh učiaceho rámca pre navrhnutý model DBPNet.

Umelé pseudo-anotácie (viď. modré šípky) sú pre čiastočne riadené učenie generované priamo z pravdepodobnostných predikcií, najjednoduchšie často pomocou fixnej prahovej hodnoty alebo *argmax* [10, 39]. Alternatívou sú funkcie *sharpening* [65] alebo teplotne kalibrovaný softmax (*T-softmax*) [75], ktoré sa lepšie vysporia-

dajú s nepresnými predikciami z ne-anotovaných dát, ktoré môžu obmedziť konvergenciu modelu. Ich voľbu pre náš model ďalej rozoberáme v návrhu stratovej funkcie (kapitola 5.3) a v samotných experimentoch.

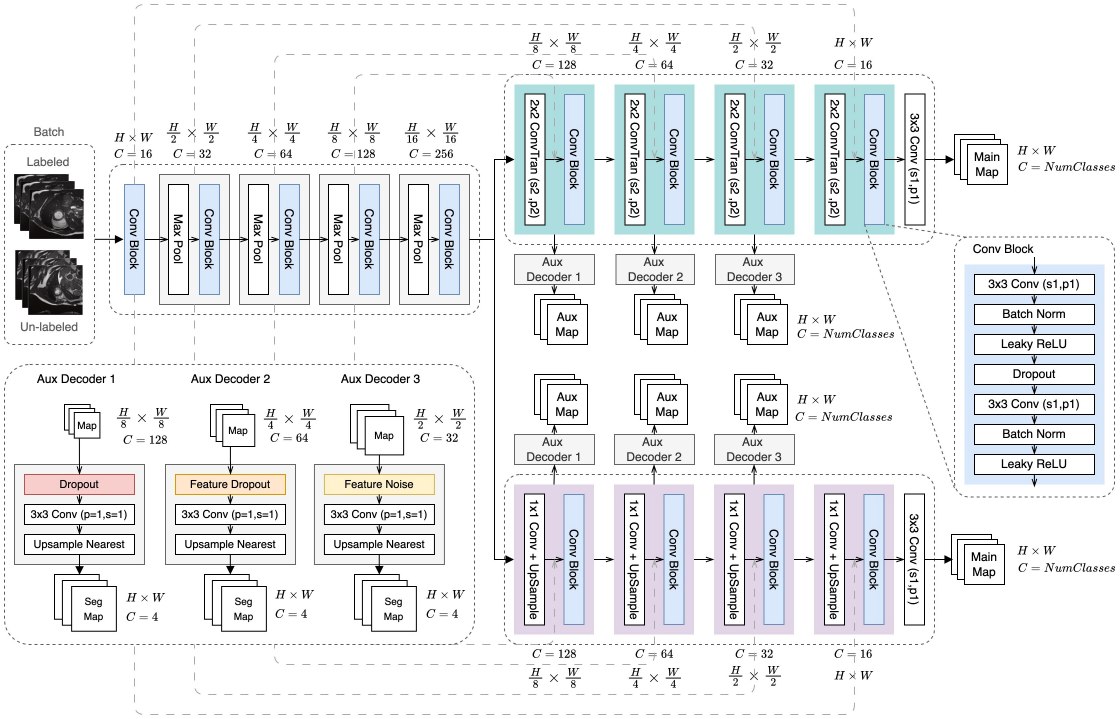
5.2 Architektúra

DBPNet sa v základe riadi architektúrou U-Net [49], ako ostatné metódy z tejto oblasti, pretože je stále hlavným referenčným modelom pre férové porovnávanie navrhovaných metód a učiacich rámcov segmentácie 2D obrazu s čiastočným riadením. Podrobnú štruktúru vrátane obsahu blokov sme znázornili na obr. 5.2

Základom sú **konvolučné bloky** zložené z dvoch 3x3 kernel konvolučných vrstiev. Oproti klasickej U-Net majú nastavené veľkosti stride a padding tak aby udržovali rovnaké rozlíšenie v rámci bloku na danej škále. Navyše sú použité štandardné koncepty zlepšujúce tréning a konvergenciu modelov v podobe *Leaky ReLU* aktivačnej funkcie a batch normalizácie po každej konvolučnej vrstve v bloku. Batch normalizácia [21] normalizuje aktivačné mapy na nulovú strednú hodnotu a jednotkový rozptyl, čo zmenšuje vnútorný posun kovariancie a efektívne pomáha voči explodujúcemu/ miznúcemu gradientu a vedie k plynulejšiemu toku gradientov. Medzi konvolučnými vrstvami je zavedený *dropout*, postupne po blokoch enkódera aj dekóderov s pravdepodobnosťou 0.05, 0.1, 0.2, 0.3 a 0.5.

Enkóder siete je nastavený ako extraktor príznakov zložený z 5 za sebou zapojených konvolučných blokov. Medzi blokmi je maximálne združovanie na zmenšenie rozlíšenia máp príznakov o polovicu a postupné zväčšovanie receptívneho poľa. Zmenšovanie rozlíšenia je kompenzované dvoj-násobným zväčšovaním počtu kanálov poslednou konvolúciou v každom konvolučnom bloku, aby boli reprezentácie priebežne zamerané na viac vysoko-úrovňové sémantické informácie. Konkrétne hodnoty rozmerov na výstupe blokov sú v obr. 5.2.

Dekódery siete využívajú rovnaké konvolučné bloky a nasledujú opačný prístup v nastavení počtu kanálov a rozlíšenia máp príznakov, kvôli spätnej rekonštrukcii do vstupných rozmerov. Na výstupoch dekodérov je zavedená 3x3 konvolúcia pre nastavenie počtu kanálov podľa počtu segmentačných tried. Podľa výsledkov experimentov URPC [40], v ktorých testovali vplyv počtu použitých škál na výkon, sme zvolili v oboch dekóderoch 4 konvolučné bloky, čo vyšlo ako najlepšie nastavenie pre ich pyramídovú konzistenciu. Dekodéry sa líšia v použití rozdielnych operácií zväčšovania rozlíšenia medzi blokmi za účelom modelovania neistoty a zavedenia perturbácií na úrovni architektúry. V prvom dekóderi sme zvolili transponovanú konvolúciu a v druhom kombináciu 1x1 konvolúcie a *upsampling* operácie s bilineárnou interpoláciou. Okrem toho je v rámci všetkých blokov umiestený *dropout*, čo má okrem predchádzania pre-trénovaniu aj dodatočný efekt perturbácií.



Obr. 5.2: Vizualizácia detailu architektúry DBPNet: Použité bloky, vrstvy a pomocné pyramídové hlavy/dekódery (*aux decoder*). Výstupné mapy sú v logitoch.

Podobne ako v U-Net [49] a iných enkóder-dekóder riešeniach [76, 11, 43, 20], sú medzi príslušnými blokmi rovnakej škály pomocou konkatenácie zavedené **skokové prepojenia** z enkódera do dekóderov. Prepojenia v tomto nastavení pomáhajú toku gradientov [76] a prenášajú príznaky vyššieho rozlíšenia, čo umožňuje ich využitie v dekóderoch pre detailnejšiu rekonštrukciu.

Pomocné segmentačné hlavy sme zaviedli za výstupom z každého bloku dekóderov, okrem poslednej škály. U všetkých hláv ide v základe o konvolučnú vrstvu (na nastavenie počtu kanálov na počet segmentovaných tried) a *upsampling* operáciu (na zväčšenie máp príznakov do pôvodných rozmerov vstupu). Spracovaniu hlavou predchádza operácia perturbácie, aby aj pyramídové predikcie vytvárali nezrovnalosti. Na rozdielnych škálach bol zvolený rozdielny prístup perturbácie: *dropout*, *feature dropout* alebo *feature noise*. Dropout náhodne s pravdepodobnosťou 0.5 deaktivuje elementy mapy príznakov (nastaví na 0). Feature dropout deaktivuje príznaky na základe ich priestorovej dôležitosti teda vypočíta mapu pozornosti spriemerovaním vstupu podľa kanálovej dimenzie. Následne vypočíta prahovú hodnotu ako náhodnú hodnotu z rozsahu 70% až 90% z maximálnej hodnoty mapy pozornosti. Prvky vstupnej mapy, ktoré sú v mape pozornosti pod prahom, sa nastavia na 0 pre všetky kanály. Feature noise pridáva do vstupu multiplikatívny šum vyberaný pre každý kanál z náhodného rovnomerného rozdelenia.

5.3 Návrh stratovej funkcie

V nasledujúcej časti definujeme počiatočný návrh celkovej stratovej funkcie, s ktorou sme začali proces experimentovania a ďalšieho iterovania jej podoby. Ak pre spracovávanú dávku (batch) definujeme Z ako jej výstupnú (logit) mapu z ľubovolného dekódera alebo pomocnej hlavy, tak mapa pravdepodobnostnej predikcie je $P = softmax(Z)$. Elementy v P hovoria o pravdepodobnosti, že pixel na danej

pozícii patrí do konkrétnej triedy, určenej podľa kanála pozície. Nech $\tilde{P}$ je potom mapa pseudo-anotácie vygenerovaná z P.

Zložku čiastočne riadeného učenia $\mathcal{L}_{\text{con}}$ konzistentnou regularizáciou dekóderov za pomoci anotovaných aj ne-anotovaných dát definujeme ako:

$$\mathcal{L}_{\text{con}} = \frac{1}{4}\sum_{s=1}^{4}[D(\tilde{P}_{TR}^{(s)}, P_{UP}^{(s)}) + D(\tilde{P}_{UP}^{(s)}, P_{TR}^{(s)})] \quad D = \text{MSE} \tag{5.1}$$

Kde subskript TR alebo UP označuje vetvu resp. jeden z 2 hlavných dekóderov a s označuje škálu vetvy. Napríklad $s = 4$ je škála hlavných výstupov. D je vzdialenostná metrika (strata) na meranie rozdielnosti medzi predikciou P a prislúchajúcou pseudo-anotáciou $\tilde{P}$ inej vetvy na škále s.

V prvých experimentoch sme ako D zvolili stratu strednej kvadratickej odchýlky (ang. *mean squared error*) definovanú ako:

$$\text{MSE}(A, B) = \frac{1}{N}\sum_{i=1}^{n}(A_i - B_i)^2 \tag{5.2}$$

Kde A a B sú tenzory rovnakých rozmerov, i je index elementov a N je počet elementov v tenzore. Vysoká hodnota tejto metriky hovorí o veľkej rozdielnosti medzi predikciami a v prípade hodnoty 0 o perfektnej zhode.

Pseudo-anotácie generujeme z hlavných aj pyramídových výstupov. Pre D v podobe MSE sme zvolili ako operáciu generovania pseudo-anotácií tzv. ostrenie (*sharpening*) od MCNet+ [65], pretože sa v ich riešení overilo ako účinné pre výstupy hlavných dekodérov v kombinácii s $D = \text{MSE}$. Ostrenie je definované ako:

$$\tilde{P} = \frac{P^{1/T}}{P^{1/T} + (1-P)^{1/T}} \quad T = 0.1 \quad P = softmax(z) \tag{5.3}$$

Kde T je hyperparameter kontroly teploty ostrenia. Podľa najlepších výsledkov ich experimentov s týmto parametrom, sme nastavili $T = 0.1$. Správne nastavenie môže okrem presadenia konzistentnej regularizácie zabezpečiť aby do učenia nebolo

zavádzaného priveľa šumu z nepresných anotácií, čo by rozhodilo model [65]. Je to preto potenciálna možnosť experimentovania.

Zložku plne riadeného učenia $\mathcal{L}_{sup}$ z limitovaného počtu skutočných anotácií definujeme v tvare:

$$\mathcal{L}_{sup} = \mathcal{L}_{dice}(P_{TR}^{(4)}, Y) + \mathcal{L}_{dice}(P_{UP}^{(4)}, Y) \tag{5.4}$$

Kde $P_{TR}^{(4)}$ a $P_{UP}^{(4)}$ je mapa pravdepodobnostnej predikcie poslednej škály na vetve TR a UP a Y je one-hot enkódovaný tenzor skutočnej anotácie rozmerov $B \times C \times H \times W$, kde jeden element určuje zaradenie pixelu do danej triedy. Zložka $\mathcal{L}_{dice}$ je strata dice koeficientom, bežne používaná v plne riadených riešeniach. [61] V súvisiacich prácach [40, 65] je definovaná ako priemerná strata pre 1 triedu:

$$\mathcal{L}_{dice} = \frac{1}{C}\sum_{c=1}^{C}\mathcal{L}_{dice}^{c} \quad \mathcal{L}_{dice}^{c} = 1 - \frac{2\sum_i P_{i,c} \cdot Y_{i,c} + \epsilon}{\sum_i P_{i,c}^2 + \sum_i Y_{i,c}^2 + \epsilon} \tag{5.5}$$

Kde C je počet všetkých tried, $P_{i,c}$ je predikovaná pravdepodobnosť, že pixel i patrí triede c a $y_{i,c}$ je skutočná binárna hodnota určujúca či pixel i patrí triede c. Táto strata je v 1 - dice koeficient, ktorý rozoberáme v kap. metrík (5.4.3)

Celkovú stratu nakoniec definujeme ako váhovaný súčet oboch zložiek v tvare:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{sup}} + \lambda \cdot \mathcal{L}_{\text{con}} \qquad \lambda(t) = w_{max} \cdot e^{(-5(1-\frac{t}{(t_{max}})^2)} \tag{5.6}$$

Parameter λ je gausova váhovacia funkcia [71] (ang. *warming up function*) v tvare sigmoidy, na kontrolu rovnováhy medzi riadenou a čiastočne riadenou zložkou straty. Funkcia v priebehu tréningu navyšuje váhu $\mathcal{L}_{\text{con}}$ približne od 0 do $w_{max} = 0.1$, aby z počiatku nebol model zahltený regularizáciou, ak nie je dostatočne naučený zo skutočných anotácií. Parameter t je krok navýšenia, ktorý sa v priebehu tréningu inkrementuje (napr. epocha, iterácia) a t_{max} je konečný krok. Poznámkou je, že $\mathcal{L}_{\text{sup}}$ je aplikovaná len na anotované vzorky výstupnej dávky a $\mathcal{L}_{\text{con}}$ na celú dávku (batch), aby bolo do regularizácie zapojených viac dát.

5.4 Metodológia experimentovania

Navrhnutý model DBPNet a jeho prvotné nastavenie učenia sme overili v porovnaní so súvisiacimi riešeniami, z ktorých vychádzame (kapitola 5.5). Výskum v čiastočne riadenej segmentácii je predovšetkým v návrhu učenia, ktorý využije veľké množstvo ne-anotovaných dát v prospech kvalitnejších výsledkov. S prvotným návrhom sme preto, za účelom zlepšenia segmentačného výkonu, ďalej experimentovali a prispôsobovali modelu hlavne v podobe úprav a rozšírení celkovej stratovej funkcie. K aktuálnej podobe návrhu sme sa dopracovali prostredníctvom viacerých prototypov celkovej straty, ktoré sme overovali a porovnávali v trénovacích experimentoch po stanovenú hranicu. Išlo o iteratívny proces, v ktorom sme sa na základe dosiahnutého výkonu rozhodovali, ktoré nastavenie/koncept bude ponechané ďalej. Prototypy sme navrhovali podľa úprav, ktoré nám dávali zmysel otestovať pre potenciálne zlepšenie pri obmedzených anotáciách alebo dopĺňali konceptmi, ktoré v podobných riešeniach preukázali zlepšenie výsledkov.

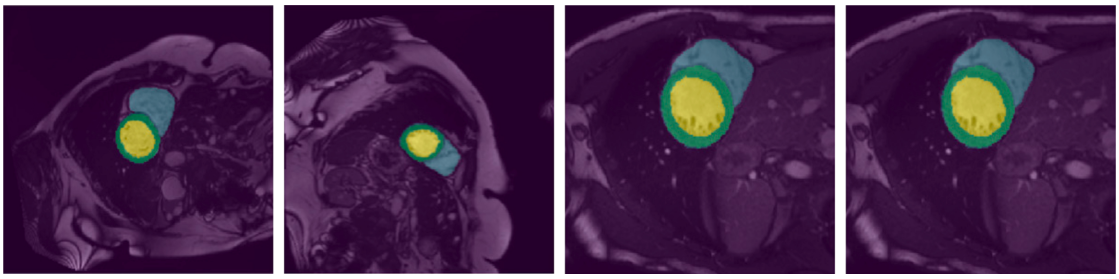
Celkovo sme tak experimenty zoskupili postupne do viacerých fáz/skupín podla konceptu, ktorý sme upravovali:

- Experimenty zložky čiastočne riadeného učenia založené na strednej kvadratickej odchýlke (MSE) - kapitola 5.6.1
- Experimenty zložky čiastočne riadeného učenia založené na Kullback Leibler divergencii (KL) - kapitola 5.6.2
- Prototypy kombinujúce rôzne nastavenia váh a vzdialenostných metrík (KL a MSE) čiastočne riadeného učenia - kapitola 5.6.3
- Prototypy s dodatočnou zložkou straty založenou na minimalizácii entropie - kapitola 5.6.4

- Experimenty nastavenia zložky plne riadeného učenia - kapitola 5.7.1 a 5.7.2
- Metódy zväčšovania rozlíšenia v rámci dekodérov - kapitola 5.8

5.4.1 Dataset

Na trénovanie v experimentoch a testovanie návrhu sme zvolili **ACDC dataset** (*Automated Cardiac Diagnosis Challenge*), často používaný v prácach slabo riadenej segmentácie ako benchmark dataset. Ide o 200 krátko-osích cine-MRI 3D skenov srdca od 100 pacientov, rovnomerne rozdelených do 5 podskupín (4 patologické a 1 zdravá skupina subjektov) podľa stavu: normálny (zdravý) stav, infarkt myokardu, dilatačná kardiomyopatia, hypertrofická kardiomyopatia, abnormálna pravá komora. Vzorky datasetu boli zozbierané zo skutočných klinických vyšetrení získaných v Univerzitnej nemocnici v Dijone, s typickým rozlíšením $1.8 \times 1.8 \times 10.0 mm^3$ na 1.5T a 3T systémoch. [5] Pre každý subjekt boli zaznamenané 2 snímky podľa rozličného časového okna a stavu srdca (koniec diastoly alebo systoly). Snímky pozostávajú zo 7-12 rezov a vzhľadom na veľký odstup medzi nimi (5mm) a možné posuny spôsobené dýchaním, je dataset braný ako úloha 2D segmentácie rezov krátkej osi. Segmentačné masky rezov boli manuálne anotované ich expertom a vymedzujú 4 oblasti záujmu (triedy): endokard ľavej komory (LV), pravej komory (RV), myokard ľavej komory (MV) a pozadie.



Obr. 5.3: Príklady rezov ACDC datasetu. [46] Ukážky sú prekryté segmentačnými maskami vyznačujúce ich triedy: endokard ľavej komory (žltá), endokard pravej komory (modrá), myokard ľavej komory (zelená) a pozadie (fialová).

5.4.2 Nastavenie trénovania a implementácia

Riešenie sme implementovali v jazyku Python 3.10.14 s použitím rámca PyTorch 2.5.1 a ďalších knižníc uvedených v *requirements.txt*. Vybrané časti a nastavenia nevyhnutné pre férovejšie porovnanie so súvisiacimi prácami MCNet+ a URPC (ako napr. metriky, predspracovanie, augmentácie, testovacia logika) vrátane nastavenia generátorov náhodných čísel, sme re-implementovali do svojej kódovej základne za pomoci ich verejných repozitárov od Wu et al. [65] a Luo et al. [40]. Druhý spomenutý, je používaný ako verejný benchmark pre čiastočne riadené učenie s medicínskymi dátami, pričom zoskupuje implementácie aj iných prác pre podporu výskumu.

Experimenty boli vykonávané na platforme Azure Machine Learning na systémoch s 6 jadrovým CPU, 112 RAM, V100 GPU alebo 40 jadrovým CPU, 320GB RAM a H100 GPU pre rýchlejší tréning. Prototypy boli trénované po hranicu 30 alebo 50 tisíc iterácií (v závislosti od konvergencie), počas ktorých bola každých 200 iterácií vykonávaná validačná fáza. Pre testovanie, boli priebežne ukladané parametre najlepšieho modelu z validácie podľa metriky DSC↑. Pri validácii aj testovaní sme zvolili sieť s vetvou UP, ktorá využíva *upsampling* operácie, pričom je využitý jej hlavný výstup (škála $s = 4$). V priemere trval tréning 6 až 7 hodín na V100 GPU a 2 hodiny na H100 GPU.

5.4.2.1 Predspracovanie a augmentácie

Dataset je autorom Luo et al. [40] fixne pred-rozdelený na 70, 10 a 20 trénovacích, validačných a testovacích skenov pacientov (dokopy 1902 rezov). Vzorky 2D rezov sú pre tréning načítavané ako 1-kanálové (*grayscale*) snímky, typicky v rozlíšení 256x224px. Vyskytujú sa aj iné rozmery (od 154 do 512 px), preto sú z rezov a ich anotácií počas trénovania extrahované náhodné výrezy 256x256px pomocou pri-

blíženia (*zoom*). Na ďalšie rozšírenie trénovacej sady a vyhnutie sa pre-trénovaniu, sú v rámci augmentácií vykonávané náhodné rotácie v rozsahu -20° až 20° a prevracanie s pravdepodobnosťou 50%. Počas validačnej a testovacej fázy sú skeny spracovávané ako celé objemy. Ich rezy sú priblížené na 256x256px ako vstup do modelu a po spracovaní je výstup inverzne zväčšený do pôvodných rozmerov. V tejto podobe sú výstupy skladané do 3D objemov pred vyhodnotením metrík.

5.4.2.2 Hyperparametre

Hyperparametre, spoločné a nemenené počas experimentov, boli nastavené ako MCNet+ [65]. Veľkosť dávky bola zvolená na 24, z ktorých je 12 anotovaných a 12 ne-anotovaných rezov. Výber dávok je implementovaný tak, aby 1 epoch náhodne iteroval cez všetky anotované dávky. Počas epochu môže byť viackrát náhodne iterované cez všetky ne-anotované dávky. Experimentovali sme v typickom nastavení, 10% anotovaných a zvyšok ne-anotovaných dát. Generátory náhodných čísel sme nastavili rovnako na hodnotu 1337, pričom výber vzoriek, augmentácie aj vyčlenenie ne-anotovaných vzoriek je deterministické. Ako optimizátor sme zvolili SGD s momentom 0.9, weight decay 0.0001 a mierou učenia 0.01. Parametre prvej podoby celkovej stratovej funkcie sme nastavili na $T = 0.1, w_{max} = 0.1, t_{max} = 200$ a t sme inkrementovali po každom 150-tom trénovanom kroku.

5.4.3 Metriky

Segmentačný výkon sme vyhodnocovali na základe 4 metrík používaných pri modeloch segmentácie medicínskych obrazových dát. Každá z týchto metrík je počítaná pre jednu triedu a potom spriemerovaná naprieč všetkými triedami. To znamená, že operujú nad binárnou segmentačnou maskou predikcie modelu a anotácie. Pre ľubovolnú triedu (región) definujme Y ako skutočnú (ground truth) binárnu masku a S ako predikovanú binárnu masku, ktorá je výstupom modelu.

5.4.3.1 Oblastne založené metriky

Oblastne založené (ang. *region-based*) metriky [23] hodnotia schopnosť segmentovať celý povrch oblasti záujmu. Z pomedzi nich vyhodnocujeme **Dice similarity coefficient** (**DSC**↑) a **Intersection over Union** (**IoU** ↑).

DSC meria prekryv (overlap) medzi regiónmi v S a Y [23]:

$$DSC = \frac{2|Y \cap S|}{|Y| + |S|} \tag{5.7}$$

IoU meria pomer medzi prienikom a zjednotením regiónov v S a Y: [23].

$$IoU = \frac{|Y \cap S|}{|Y \cup S|} \tag{5.8}$$

Hodnoty oboch metrík môžu nadobúdať rozsah 0 a 1: hodnota 1 perfektný prekryv, 0 žiadny prekryv. IoU, nazývaná tiež Jaccardov index, viac penalizuje nepresnosť segmentácie v malých regiónoch. Naopak DSC nie je v chybách odhadu regiónov tak citlivá, ale je viac senzitívna na ich správne zarovnanie.

5.4.3.2 Hranične založené metriky

Hranične založené (ang. *boundary-based*) metriky [23] hodnotia zarovnanie hraníc segmentovaného regiónu voči hraniciam skutočného regiónu, pre odhalenie chýb v okrajových častiach oblastí. Z pomedzi nich vyhodnocujeme **Average Surface Distance** (**ASD**↓) a **95th percentile Hausdorff Distance** (**HD**$_{95}$↓).

Pre vysvetlenie, definujme ∂S ako množinu odhadnutých hraničných resp. okrajových bodov (pixelov) regiónu z S a ∂Y ako množinu skutočných hraničných bodov regiónu z Y.

HD (housdorffová vzdialenosť) vyhodnocuje najväčšiu vzdialenosť od hraničného bodu v množine ∂S k najbližšiemu hraničnému bodu v množine ∂Y. HD$_{95}$ je úprava HD, ktorá zohľadňuje len 95 percentil zo vzdialeností kvôli zníženiu vplyvu

odľahlých hodnôt. Je definovaná ako [62]:

$$HD_{95} = \max\left(\max_{y\in\partial Y}\min_{s\in\partial S} d(y,s), \max_{s\in\partial S}\min_{y\in\partial Y} d(y,s)\right)_{95\%} \tag{5.9}$$

Kde $d(y,s)$ je euklidovská vzdialenosť medzi hraničnými bodmi y a s.

ASD meria priemernú najkratšiu vzdialenosť medzi hraničnými bodmi z ∂S a ich najbližšími hraničnými bodmi z ∂Y.

$$ASD = \frac{1}{|\partial Y| + |\partial S|}\left(\sum_{y\in\partial Y}\min_{s\in\partial S} d(y,s) + \sum_{s\in\partial S}\min_{y\in\partial Y} d(y,s)\right) \tag{5.10}$$

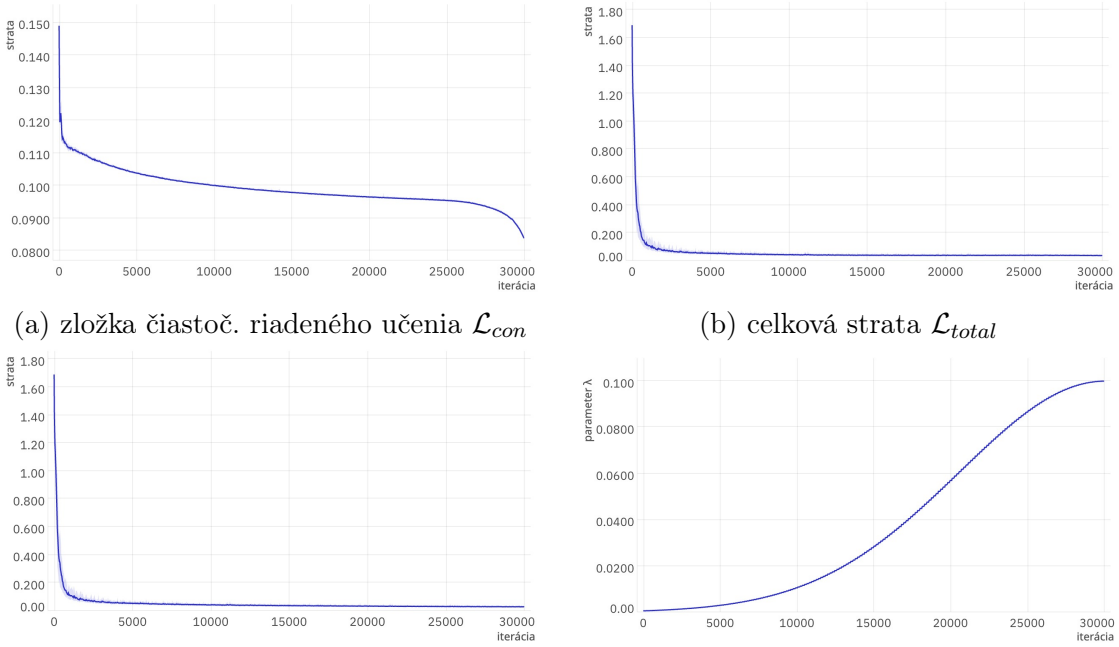
5.4.3.3 Logovanie a vyhodnocovanie metrík

Počas validačných fáz sme merali priemerné DSC↑ a HD_{95} ↓ zo všetkých vzoriek validačnej sady. Primárne sme sa pri vyhodnocovaní experimentov riadili podľa výsledkov DSC a ak boli podobné, zohľadňovali sme aj HD_{95}. Pri testovaní sme merali priemerné hodnoty aj ostatných metrík zo všetkých vzoriek testovacej sady. Priebeh experimentov sme zaznamenávali platformou neptune.ai.

5.5 Porovnanie základu so súvisiacimi prácami

Experimentom č.1.1 sme overili počiatočný návrh s opísanými nastaveniami. Priebeh tréningu resp. hodnôt celkovej straty a jej zložiek znázorňujú grafy na obr. 5.4. Je možné vidieť, že navrhnutá celková strata (graf b) a jej väčšinová zložka plne riadeného učenia (graf c) spočiatku nadobúdala hodnotu 1.7 a v počiatočnej fáze je rýchlo minimalizovaná s následnou konvergenciou po zbytok tréningu. To môže indikovať potenciálne učenie. Parameter λ (graf d) sme nastavili tak, aby dosiahol váhu 0.1 až na konci tréningu. Postupným zaradzovaním do tréningu bola minimalizovaná zložka čiastočne riadeného učenia (graf a), kde pozorujeme prudký pokles na konci tréningu. Myslíme si, že to bolo spôsobené tým, že pomocné

dekódery začnú produkovať lepšie a konzistentnejšie predikcie na konci tréningu kedy hlbšie reprezentácie sú postupne komplexnejšie a prispôsobené segmentácii než na začiatku tréningu. Toto pozorovanie bolo jedným z podnetov pre ďalšie experimentovanie so stratovou funkciou. To znamená napr. predĺženie tréningu, iné nastavenie parametra λ alebo rozdelenie zložky čiastočného riadenia $\mathcal{L}_{con}$ na 2 samostatné zložky: pre pyramídové predikcie a pre predikcie hlavných výstupov, za účelom preskúmania ich správania počas tréningu a osobitnú kontrolu.



(a) zložka čiastoč. riadeného učenia $\mathcal{L}_{con}$

(b) celková strata $\mathcal{L}_{total}$

(c) zložka plne riadeného učenia $\mathcal{L}_{sup}$

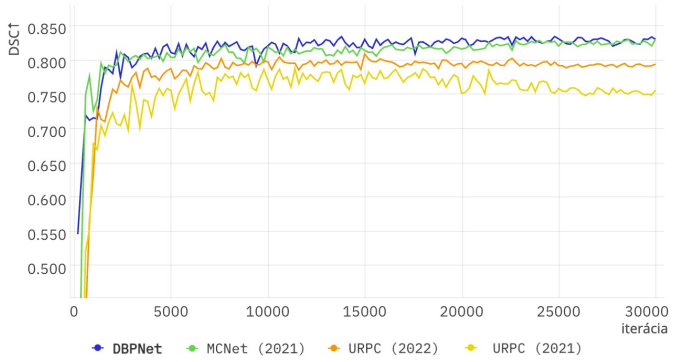
(d) parameter λ

Obr. 5.4: Priebeh stratovej funkcie a jej zložiek počas trénovacieho experimentu č.1.1 prvotného návrhu DBPNet.

Počiatočný prototyp sme overili porovnaním s modelmi, od ktorých sme sa inšpirovali pri návrhu. Ide o verziu riešenia MCNet [66] s dvoma dekódermi, pre férovejšie porovnanie s DBPNet a 2 verzie riešenia URPC, líšiace sa v nastavení celkovej straty. Novšia verzia URPC [40] r. 2022 (kap. 4.1) váhuje aj jednotlivé čiastkové zložky straty narozdiel od staršej verzie r. 2021 [37]. Riešenia sme re-

implementovali a trénovali s rovnakým nastavením spoločných hyperparametrov, okrem miery učenia, ktorú URPC znižuje stratégiou *poly learning rate*. Graf obr. 5.5 znázorňuje priebeh validačného DSC↑ všetkých modelov počas tréningu.

Ako je možné vidieť, tak URPC modely dosiahli nižší výkon (0.8095 DSC a 0.7880 DSC) oproti 2-dekóderovým modelom, pričom staršia verzia v polovici tréningu prechádzala do pre-trénovania. DBPNet model ukázal schopnosť učenia sa a konvergencie za pomoci počiatočného prototypu straty a dosiahol 0.8348 DSC v iterácii 24800, čo je podobný o niečo lepší výkon ako MTNet (0.8306 DSC v iterácii 28600). To môže naznačovať, že konzistentná regularizácia v našom nastavení na pomocných škálach môže potenciálne navyšovať výkon oproti regularizácii len hlavných výstupov ako v prípade MTNet, ale potrebuje ďalšie dolaďovanie.



Obr. 5.5: Graf priebehu vývoja DSC↑ validačných fáz počas trénovacích experimentov súvisiacich modelov v porovnaní s počiatočným návrhom DBPNet.

5.6 Experimenty čiastočného riadenia

Prvé 2 fázy experimentovania sme založili na rôznych vzdialenostných metrikách D. Vybrané 2 metriky (MSE a KL) sme pozorovali u viacerých riešeniach [65, 40, 71, 39, 24] ako súčasť konzistentnej regularizácie, preto bolo zámerom zistiť, ktorú z nich bude vhodné použiť a v akom nastavení pre náš navrhnutý rámec.

5.6.1 Čiastočné riadenie na základe MSE

V 1. fáze experimentov (skrátene exp) sme prototypovali so stratovou funkciou a jeho zložkou čiastočne riadeného učenia v podobe využívajúcej MSE ako metriku D. Graf obr. 5.6 znázorňuje priebeh validačného DSC↑ počas trénovania prototypov tejto skupiny, aj v porovnaní už s opísaným **exp č.1.1**.

Experiment č.1.2 overoval gausovú podobu parametra λ pre aktuálnu konfiguráciu straty, a to nastavením na konštantu 0.1. To sa ukázalo s mierne pomalšou konvergenciou než exp. č.1.1.

V **experimente č.1.3** sme na podnet pozorovania opísaného na začiatku kapitoly 5.5, rozhodli rozdeliť zložku $\mathcal{L}_{con}$ na 2 osobitne váhované zložky:

$$\mathcal{L}_{\text{con}}^{main} = MSE(\tilde{P}_{TR}^{(4)}, P_{UP}^{(4)}) + MSE(\tilde{P}_{UP}^{(4)}, P_{TR}^{(4)}) \tag{5.11}$$

$$\mathcal{L}_{\text{con}}^{aux} = \frac{1}{3}\sum_{s=1}^{3}[MSE(\tilde{P}_{TR}^{(s)}, P_{UP}^{(s)}) + MSE(\tilde{P}_{UP}^{(s)}, P_{TR}^{(s)})] \tag{5.12}$$

Kde $\mathcal{L}_{\text{con}}^{main}$ samostatne regularizuje výstupy hlavných dekóderov (škálu $s = 4$) a $\mathcal{L}_{\text{con}}^{aux}$ pomocné pyramídové predikcie v podobe priemeru ako v pôvodnej zložke. V celkovej strate ich potom váhujeme (ako pôvodne) gausovou funkciou v tvare:

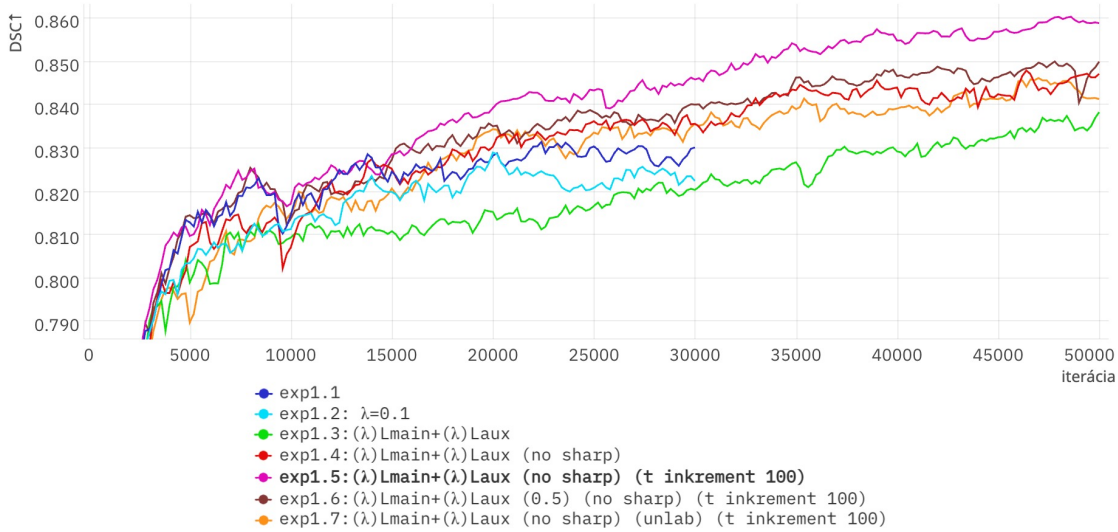
$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{sup}} + \lambda \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux} \qquad \lambda(t) = w_{max} \cdot e^{(-5(1-\frac{t}{(t_{max}}))^2)} \tag{5.13}$$

Nová konfigurácia konvergovala počas 30 tisíc iterácií pomalšie než predchádzajúce avšak vykazovala potenciálne ďalší rast (viď. graf 5.6), preto sme predĺžili tréning na 50 tisíc iterácií. Ako sme predpokladali tak zložka $\mathcal{L}_{aux}$ spôsobovala prudkú minimalizáciu straty na konci tréningu, ktorú sme opisovali (obr. 5.4 a). S novým tvarom straty a dĺžkou tréningu sme preto pokračovali v experimentovaní aby sme to vyriešili, aj vzhľadom na nárast vo výkone v porovnaní s predchádzajúcimi experimentmi.

Experimentom č.1.4 sme z novej konfigurácie straty (vzorec 5.13) odstránili generovanie pseudoanotácií z $\mathcal{L}_{\text{con}}^{aux}$ aby sme overili efektivitu operácie *ostrenia* na pyramídových predikciách. Zjednodušili sme tým zároveň tvar $\mathcal{L}_{\text{con}}^{aux}$ (viď 5.14), pretože v MSE nezáleží na poradí argumentov ak pracuje len s pravdep. mapami. $\mathcal{L}_{\text{con}}^{main}$ sme ponechali v pôvodnom tvare 5.11 so pseudo-anotáciami, vzhľadom na to, že je dokázaná ich efektivita na hlavných výstupoch dekóderov [65]

$$\mathcal{L}_{\text{con}}^{aux} = \frac{1}{3}\sum_{s=1}^{3} MSE(P_{TR}^{(s)}, P_{UP}^{(s)})) \tag{5.14}$$

Celková strata ostala v rovnakom tvare. Výsledkom novej podoby $\mathcal{L}_{\text{con}}^{aux}$ bolo vyriešenie jeho priebehu (ktorý je už rovnomerný) a nárast vo výkone oproti experimentu č.1.3, čo hovorí aj o nevhodnosti ostrenia na pomocných škálach.



Obr. 5.6: Graf priebehu vývoja DSC↑ validačných fáz počas trénovacích experimentov prototypov založených na $D = MSE$.

Experimentu č.1.5 sme preto nastavili v podobe exp č.1.4 (bez pseudo-anotácií pre $\mathcal{L}_{\text{con}}^{aux}$) a testovali sme vplyv rýchlosti nástupu parametra λ. Doteraz sme parameter t inkrementovali po každých 150 iteráciách a teda parameter λ nadobudol hodnotu 0.1 až v iterácii 30000. Ako vhodná možnosť sa nám zdala skúsiť zrýchliť

nástup inkrementovaním po 100 iteráciách. Výsledkom bol najlepší výkon v tejto skupine. Testovali sme aj nižšie inkrementovanie po 75 iteráciách ale to neprinieslo ďalšie zlepšenie. V posledných dvoch experimentoch založený na MSE sme preto pokračovali s nastavením tohto experimentu: parameter t inkrementovaný po 100 a konfigurácia straty v podobe vzorca 5.11, 5.13, 5.14.

Experiment č.1.6 mal zmenšenú váhu $\mathcal{L}_{\text{con}}^{aux}$ o polovicu teda $\mathcal{L}_{\text{con}}^{aux}/2$, s myšlienkou, že zložka $\mathcal{L}_{\text{con}}^{aux}$ môže oproti $\mathcal{L}_{\text{con}}^{main}$ potenciálne brzdiť učenie, pretože pomocné hlavy a hlboké vrstvy dekodérov môžu mať v skorých fázach tréningu ešte nie ideálne naučené reprezentácie.

Experiment č.1.7 testoval $\mathcal{L}_{\text{con}}^{aux}$ aplikovanú len na ne-anotované dáta inšpiráciou od URPC modelu [40], ktorý na pyramídové predikcie využíva len ne-anotované dáta z dávky. Oba posledné experimenty (1.6 aj 1.7) dosiahli horší výkon než experiment č.1.5, ktorý bol najlepším nastavením tejto skupiny z pohľadu dosiahnutého validačného DSC.

5.6.2 Čiastočné riadenie na základe KL

V nasledujúcej skupine experimentov sme s naším učiacim rámcom pre DBPNet prototypovali okolo inej vzdialenostnej metriky D namiesto MSE a to okolo Kullback–Leiblerovej divergencie (KL) definovanej ako:

$$KL(P||Q) = \sum_{x \in X} P(x) \log \left(\frac{P(x)}{Q(x)} \right) \tag{5.15}$$

Hodnota KL hovorí o tom, ako veľmi sa pravdep. rozdelenie Q líši od referenčného (skutočného) pravdep. rozdelenia P, preto je vhodnou metrikou na minimalizáciu rozdielnosti predikcií. Rozdelenia blízke $KL = 0$ sú si veľmi podobné.

Experiment č.1.8: Vzhľadom na to že, konzistentná regularizácia sa s odlišnou metrikou D môže pri rôznych úpravách správať inak, tak sme začali experimento-

vanie od počiatočne navrhnutej konfigurácie zložky $\mathcal{L}_{con}$, teda v tvare:

$$\mathcal{L}_{\text{con}} = \frac{1}{4}\sum_{s=1}^{4}[KL(\tilde{P}_{TR}^{(s)}, \tilde{P}_{UP}^{(s)}) + KL(\tilde{P}_{UP}^{(s)}, \tilde{P}_{TR}^{(s)})] \tag{5.16}$$

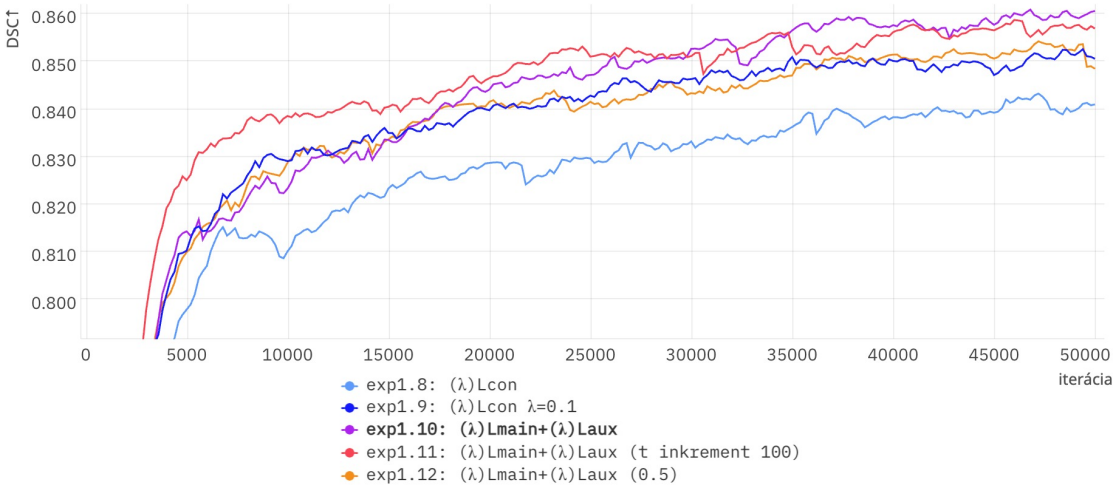
Nezmenené časti celkovej straty sme zachovali počiatočnému návrhu (vzorec 5.6):

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{sup}} + \lambda \cdot \mathcal{L}_{\text{con}} \qquad \lambda(t) = w_{max} \cdot e^{(-5(1-\frac{t}{(t_{max}})^2)}$$

Pred začatím trénovania v experimentoch sme vykonali aj ďalšie úpravy, ktoré boli už overené ako efektívne v kombinácii s KL. Z analyzovaného MTNet (kap. 4.3) sme na generovanie pseudo-anotácií namiesto ostrenia použili *T-softmax* [75]:

$$\tilde{\mathbf{p}}_c = \frac{exp(\mathbf{z}_c/T)}{\sum_c exp(\mathbf{z}_c/T)} \tag{5.17}$$

T-softmax má podobný efekt ako *ostrenie*, avšak parametrom T, teplotne kalibruje priamo *logit* predikcie namiesto pravdepodobnostných predikcií. Pri $T = 1$ sa funkcia redukuje na klasický *softmax*. Pri $T < 1$ sú predikcie škálované "hore", čo ich robí viac istejšími a pri $T > 1$ sú škálované "dole", teda viac rovnomernejšie.



Obr. 5.7: Graf priebehu vývoja DSC↑ validačných fáz počas trénovacích experimentov prototypov založených na $D = KL$.

Navyše v nasledovnosti MTNet, kalibrujeme oba argumenty KL nastavením $T =$

10 a zároveň druhému argumentu vypíname spätnú propagáciu gradientov, aby sa zachoval efekt učenia zo pseudo-anotácií spôsobom učiteľ a študent. Opísanú konfiguráciu sme overili a graf 5.7 ukázal stabilný tréning aj konvergenciu.

V ďalších experimentoch sme systematicky vykonávali podobné modifikácie ako pri skupine založenej na MSE. **Experimente č.1.9** mal nastavený parameter $\lambda = 0.1$, čo zlepšilo konvergenciu aj výkon oproti nastaveniu s gaussovou váhou.

Aj napriek tomu, že konštantná váha navýšila výkon, tak sme v **experimente č.1.10** uprednostnili gaussovu váhu a testovali, podobne ako pri skupine MSE, rozdelenie $\mathcal{L}_{con}$ na dve zložky straty pre ich lepšiu kontrolu:

$$\mathcal{L}_{\text{con}}^{main} = KL(\tilde{P}_{TR}^{(4)}, \tilde{P}_{UP}^{(4)}) + KL(\tilde{P}_{UP}^{(4)}, \tilde{P}_{TR}^{(4)}) \tag{5.18}$$

$$\mathcal{L}_{\text{con}}^{aux} = \frac{1}{3}\sum_{s=1}^{3}[KL(\tilde{P}_{TR}^{(s)}, \tilde{P}_{UP}^{(s)}) + KL(\tilde{P}_{UP}^{(s)}, \tilde{P}_{TR}^{(s)})] \tag{5.19}$$

Celkovú stratu sme nastavili rovnako ako vo vzorci 5.13. Táto konfigurácia s $\mathcal{L}_{\text{con}}^{main}$ a $\mathcal{L}_{\text{con}}^{aux}$ bola nakoniec výkonnostne ešte lepšia než exp. č.1.9 (s počiatočnou konfiguráciou a fixným $\lambda = 0.1$), preto sme ju uprednostnili v ďalšom prototypovaní.

Experimentom č. 1.11 sme testovali zníženie inkrementovania t a to po každých 100 iteráciách, čo vo výsledku malo z počiatku rýchlejšiu konvergenciu, ale veľmi nezmenilo výkon, preto sme ďalej nechali pôvodné nastavenie na 150. **Exp. č.1.12** testoval zníženie váhy $\mathcal{L}_{\text{con}}^{aux}$ o polovicu čo neprinieslo ďalšie zlepšenie.

Zhrnutie prototypovania s KL aj MSE: Z pohľadu DSC↑ aj HD_{95} ↓ v skupine KL dosiahol najvyšší výkon prototyp exp. č. 1.10 využívajúci *T-softmax* a konfiguráciu dvoch zložiek $\mathcal{L}_{\text{con}}^{main}$ a $\mathcal{L}_{\text{con}}^{aux}$ pre konzistentnú regularizáciu. Tento prototyp prekonal aj najlepší prototyp s MSE (exp. č.1.5), taktiež založený na 2 zložkách: bez *ostrenia* v $\mathcal{L}_{\text{con}}^{aux}$ a s *ostrením* v $\mathcal{L}_{\text{con}}^{main}$ a t navyšované po 100 iteráciách. Ďalej preto pracujeme s nastaveniami a konfiguráciou experimentu 1.10.

5.6.3 Prototypy kombinácií váh a vzdialenostných metrík

V nasledujúcich dvoch experimentoch (1.13 a 1.14) sme testovali kombinácie 2 rôznych nastavení.

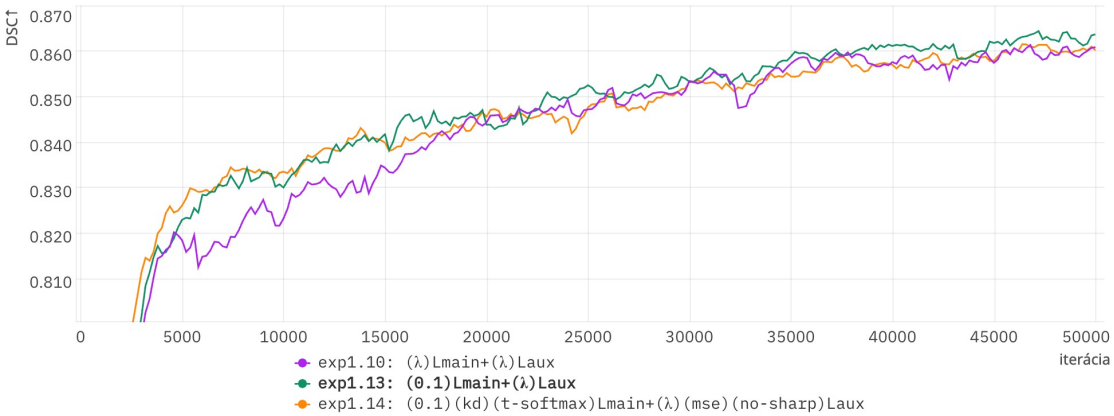
Experiment č.1.13 (kombinované váhy): V exp č.1.9 sa ukázalo, že konštantná váha $\lambda = 0.1$ pre $\mathcal{L}_{\text{con}}$ môže navýšiť výkon pri nastavení s KL. Aby sa ale nestalo, že konštantnou váhou v skorej fáze tréningu potenciálne pre-trénujeme model cez nepresné pseudo-anotácie z pomocných škál, tak sme testovali kombináciu konštantnej váhy 0.1 pre $\mathcal{L}_{\text{con}}^{main}$ a gaussovej váhy λ pre $\mathcal{L}_{\text{con}}^{aux}$. Celková strata je potom v tvare:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{sup}} + 0.1 \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux} \qquad \lambda(t) = w_{max} \cdot e^{(-5(1-\frac{t}{(t_{max}})^2)} \tag{5.20}$$

Zložky celkovej straty sa riadia podľa vzorcov 5.18 a 5.19:

$$\mathcal{L}_{\text{con}}^{main} = KL(\tilde{P}_{TR}^{(4)}, \tilde{P}_{UP}^{(4)}) + KL(\tilde{P}_{UP}^{(4)}, \tilde{P}_{TR}^{(4)})$$

$$\mathcal{L}_{\text{con}}^{aux} = \frac{1}{3}\sum_{s=1}^{3}[KL(\tilde{P}_{TR}^{(s)}, \tilde{P}_{UP}^{(s)}) + KL(\tilde{P}_{UP}^{(s)}, \tilde{P}_{TR}^{(s)})]$$



Obr. 5.8: Graf priebehu vývoja DSC↑ validačných fáz počas trénovacích experimentov prototypov kombinácií rôznych nastavení.

Priebeh trénovania tohto prototypu v porovnaní s najlepším prototypom KL skupiny (exp. č. 1.10) je možné vidieť na grafe obr. 5.8. Z pohľadu výkonu boli prototypy veľmi podobné, na konci bol experiment 1.13 výkonnejší v $DSC \uparrow$ ale naopak menej výkonný v $HD_{95} \downarrow$. My sme uprednostnili $DSC \uparrow$, preto pokračujeme do ďalšieho experimentu s kombinovaným nastavením váh.

Experiment č.1.14 (KL & MSE) kombinuje oba prístupy vzdialenostných metrík KL aj MSE. Konkrétne sme testovali nastavenie MSE bez *ostrenia* pre $\mathcal{L}_{\text{con}}^{aux}$, teda bez pseudo-anotácií (čo sa osvedčilo) a nastavenie KL a *T-softmax* pre $\mathcal{L}_{\text{con}}^{main}$. Celková strata a váhovanie nasleduje vzorec 5.20 prechádzajúceho experimentu. Pre úplnosť $\mathcal{L}_{\text{con}}^{main}$ sa riadi podľa vzorca 5.18 a $\mathcal{L}_{\text{con}}^{aux}$ podľa vzorca 5.14:

$$\mathcal{L}_{\text{con}}^{aux} = \frac{1}{3}\sum_{s=1}^{3} MSE(P_{TR}^{(s)}, P_{UP}^{(s)}))$$

$$\mathcal{L}_{\text{con}}^{main} = KL(\tilde{P}_{TR}^{(4)}, \tilde{P}_{UP}^{(4)}) + KL(\tilde{P}_{UP}^{(4)}, \tilde{P}_{TR}^{(4)})$$

Výsledný výkon prototypu bol oproti predchádzajúcemu veľmi podobný v HD_{95} a len o 0.004 horší v DSC. Rozhodli sme sa pokračovať do ďalších prototypov s nastavením predchádzajúceho experimentu 1.13, ktorý kombinuje váhovanie a využíva v oboch zložkách KL a *T-softmax*.

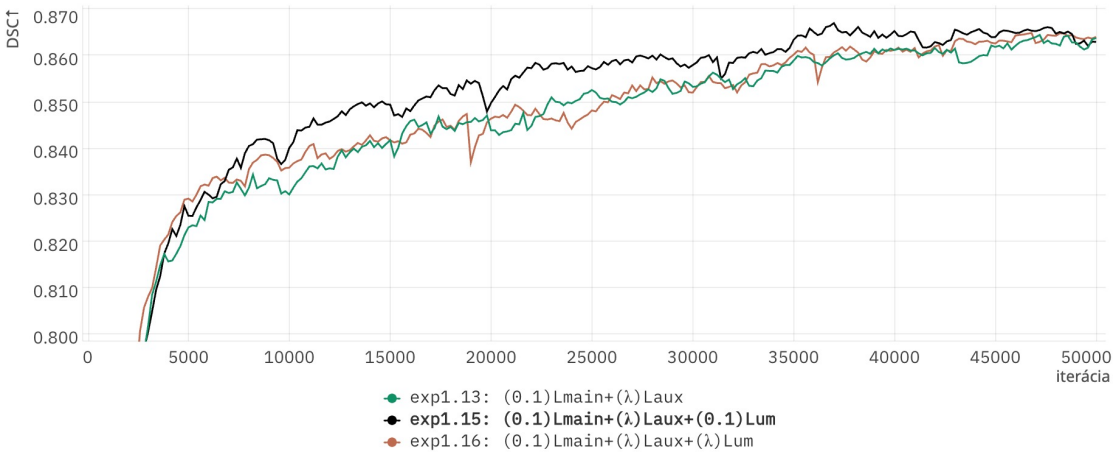
5.6.4 Prototypy s dodatočnou minimalizáciou entropie

Prototyp experimentu č.1.13 sme sa pre (potenciálne) získanie ďalšieho výkonu z čiastočného riadenia rozhodli v ďalších dvoch experimentoch (1.15 a 1.16) rozšíriť o dodatočnú zložku celkovej straty. Konkrétne ide o existujúcu zložku $\mathcal{L}_{um}$ (kap. 4.3 vzorec 4.14), ktorá kombinuje 2 koncepty, a to minimalizáciu entropie a konzistentnú regularizáciu medzi dekódermi. Definícia zložky je [75]:

$$\mathcal{L}_{um} = -\frac{1}{N}\sum_{i=0}^{N}\sum_{c=0}^{C} \bar{P}_i^c log(\bar{P}_i^c) \tag{5.21}$$

Entropia meria neistotu pravdepodobnostného rozdelenia. To znamená ak P je napr. výstupná pravdepodobnostná mapa z dekódera a na indexe i je pravdepodobnosť jednej triedy blízka hodnote 1 a u ostatných tried blízka 0, tak entropia je nízka, čo hovorí o vysokej istote v odhade. Naopak, ak majú predikcie na indexe i pre všetky triedy c navzájom podobnú pravdepodobnosť, entropia je vysoká, teda model si nie je istý v odhade triedy pixelu.

Minimalizáciou entropie je možné tlačiť model smerom k viac istým predikciám tried. Jej aplikácia samostatne na hlavné výstupy dekodérov ($s = 4$) by mohla spôsobiť, že by si dekódery pre daný pixel boli isté v odhade rozdielnej triedy. [75] Preto ju aplikujeme na priemernú pravdep. mapu $\bar{P}$ počítanú z hlavných výstupov dekodérov ($s = 4$), teda $\bar{P} = (P_{TR}^{(4)} + P_{UP}^{(4)})/2$, čo má aj efekt reg. konzistencie.



Obr. 5.9: Graf priebehu vývoja DSC↑ validačných fáz počas trénovacích experimentov prototypov s pridanou zložkou minimalizácie entropie.

Experiment č.1.15 pridáva $\mathcal{L}_{um}$ ako ďalšiu zložku do celkovej straty váhovanú konštantou 0.1. Aplikujeme ju na anotované aj ne-anotované dáta. Pre úplnosť je potom celková strata v tvare:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{sup}} + 0.1 \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux} + 0.1 \cdot \mathcal{L}_{\text{um}} \tag{5.22}$$

Experiment č.1.16 váhuje $\mathcal{L}_{\text{um}}$ gaussovou funkciou λ namiesto konštanty. Vo výsledku sme dodatočnou zložkou zlepšili výkon pri nastavení s konštantnou váhou (viď. graf priebehu trénovania na obr. 5.9).

5.6.4.1 Zhrnutie experimentovania s čiastočným riadením

Konfigurácia stratovej funkcie z experimentu 1.15 je doposiaľ najlepším nastavením čiastočného riadenia z pohľadu DSC↑, ktoré sme dosiahli pre navrhnutý DBPNet. Pre zhrnutie experimentu je jeho celková strata v tvare:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{sup}} + 0.1 \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux} + \lambda \cdot \mathcal{L}_{\text{um}} \qquad \lambda(t) = w_{max} \cdot e^{(-5(1-\frac{t}{(t_{max}}})^2)}$$

$\mathcal{L}_{\text{um}}$ sa nemenilo a nasleduje pôvodný vzorec 5.21. Navrhnuté zložky čiastočného riadenia sa riadia podľa vzorcov 5.18 a 5.19:

$$\mathcal{L}_{\text{con}}^{main} = KL(\tilde{P}_{TR}^{(4)}, \tilde{P}_{UP}^{(4)}) + KL(\tilde{P}_{UP}^{(4)}, \tilde{P}_{TR}^{(4)})$$

$$\mathcal{L}_{\text{con}}^{aux} = \frac{1}{3}\sum_{s=1}^{3}[KL(\tilde{P}_{TR}^{(s)}, \tilde{P}_{UP}^{(s)}) + KL(\tilde{P}_{UP}^{(s)}, \tilde{P}_{TR}^{(s)})]$$

Kde pseudo-anotácie sú generované operáciou *T-Softmax* a hyperparametre funkcií sú nastavené na hodnoty $T = 10, w_{max} = 0.1, t_{max} = 200$ a t sme inkrementovali po každom 150-tom trénovanom kroku (iterácii). Ostatné hyperparametre trénovania sa riadia podľa prvotného návrhu.

5.7 Experimenty plného riadenia

V nasledujúcich 2 skupinách sme experimentovali so zložkou plne riadeného učenia $\mathcal{L}_{\text{sup}}$. Doposiaľ bola zložka $\mathcal{L}_{\text{sup}}$ definovaná v podobe straty dice koeficientu medzi hlavnými výstupmi vetiev (dekodérov) a skutočnými anotáciami (vzorec 5.4). Zložku ďalej nastavujeme pre potenciálne získanie čo najväčšieho výkonu z obmedzeného počtu anotácií. Vo všetkých nasledujúcich experimentoch využívame čiastočné riadenie v najlepšom dosiahnutom nastavení (kap. 5.6.4.1).

5.7.1 Pridanie krížovej entropie alebo fokálnej straty

Experiment č.2.1 rozširuje $\mathcal{L}_{\text{sup}}$ o stratu krížovej entropie. V hlbokom učení ide o jednu z najpoužívanejších stratových funkcií, ktorá meria rozdiel medzi predikovaným a skutočným pravdepodobnostným rozdelením. Pre viac-triednu segmentáciu môže byť definovaná v tvare [69]:

$$\mathcal{L}_{ce}(p, y) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(p_{i,c}) \tag{5.23}$$

Kde N je celkový počet pixelov v snímke alebo dávke snímok, C je počet tried, $y_{i,c} \in \{0, 1\}$ je one-hot enkódovaná značka či pixel i patrí do triedy c a $p_{i,c} \in (0, 1)$ je predikovaná pravdepodobnosť, že pixel i patrí do triedy c. $\mathcal{L}_{\text{sup}}$ sme potom definovali v tvare nasledujúceho vzorca 5.24, kde aplikujeme $\mathcal{L}_{ce}$ rovnako ako $\mathcal{L}_{dice}$ na hlavný výstup (predikciu) z oboch vetiev po operácii softmax.

$$\mathcal{L}_{sup} = \mathcal{L}_{dice}(P^{(4)}_{TR}, Y) + \mathcal{L}_{dice}(P^{(4)}_{UP}, Y) + \mathcal{L}_{ce}(P^{(4)}_{TR}, Y) + \mathcal{L}_{ce}(P^{(4)}_{UP}, Y) \tag{5.24}$$

Experiment č.2.2 zamieňa stratu krížovej entropie v rámci $\mathcal{L}_{sup}$ experimentu 2.1 za tzv. fokálnu stratu (ang. *focal loss*). Ide o rozšírenie resp. variantu krížovej entropie, ktorá rieši problém nevyváženosti tried v dátach a znižuje váhu ľahko klasifikovateľných vzoriek a podporuje príspevok učenia z ťažkých vzoriek. [35] Problém nevyváženosti tried je prípad aj ACDC datasetu kde vyše 95% pixelov patrí pozadiu a ostatné triedy sú približne rovnomerne rozdelené. Fokálna strata $\mathcal{L}_{focal}$ pre viac-triednu segmentáciu môže byť definovaná ako [69]:

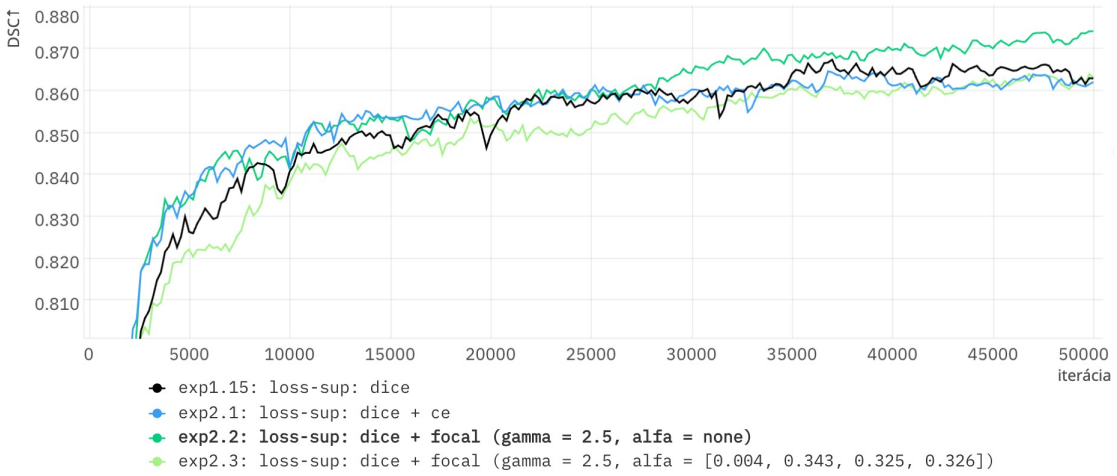
$$\mathcal{L}_{foc}(p, y) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} \alpha_c(1 - p_{i,c})^{\gamma} y_{i,c}\log(p_{i,c}) \tag{5.25}$$

Kde hyperparameter α_c nastavuje váhu (dôležitosť) triedy c a γ kontroluje stupeň zníženia váhy ľahko klasifikovateľných pixelov a zvýšenie dôrazu na nesprávne klasifikované (ťažké) pixely. $\gamma = 0$ je ekvivalentom štandardnej krížovej entropie.

Podľa autorov fokálnej straty [35] je všeobecné odporučané nastavenie $\gamma = 2$. V rámci experimentu 2.2 sme hľadali čo najoptimálnejšiu hodnotu γ pre náš dataset pričom sme zatiaľ parameter α nezadali. $\mathcal{L}_{sup}$ tohto experimentu je v tvare:

$$\mathcal{L}_{sup} = \mathcal{L}_{dice}(P_{TR}^{(4)}, Y) + \mathcal{L}_{dice}(P_{UP}^{(4)}, Y) + \mathcal{L}_{foc}(P_{TR}^{(4)}, Y) + \mathcal{L}_{foc}(P_{UP}^{(4)}, Y) \tag{5.26}$$

Experiment č.2.3 testuje $\mathcal{L}_{sup}$ v podobe dice koeficientu a fokálnej straty, pri nastavení oboch hyperparametrov: $\alpha = [0.004, 0.343, 0.325, 0.326]$ a $\gamma = 2.5$. Parameter γ je zvolený podľa najlepšieho behu predchádzajúceho experimentu. Aby parameter α správne nastavil príspevok rôznych tried do učenia, vypočítali sme α_c váhy ako inverznú frekvenciu pixelov každej triedy v rámci anotačných snímok trénovanej sady. Trieda pozadia má najnižšiu váhu $\alpha_1 = 0.004$.



Obr. 5.10: Graf priebehu najlepších trénovacích behov a ich vývoja validačného DSC↑ pre experimenty nastavenia zložky plného riadenia (exp. 2.1 až 2.3).

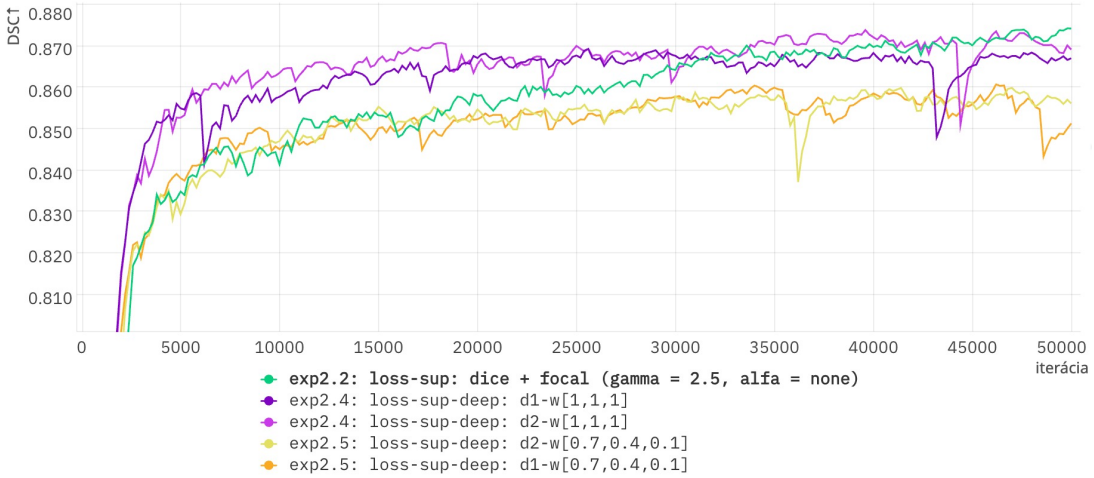
Priebeh trénovania najlepších behov opísaných experimentov 2.1 až 2.3 zobrazuje graf 5.10 v porovnaní s doteraz najlepším experimentom 1.15, ktorý doposiaľ využíval len stratu dice koeficientu v rámci $\mathcal{L}_{sup}$. Najlepší výkon dosiahlo nastavenie dice a fokálnej straty kde $\gamma = 2.5$ a α nie je nezadaná. S týmto nastavením (exp. 2.2) sme preto pokračovali ďalej.

5.7.2 Nastavenie plného riadenia na pomocných výstupoch

Experiment č. 2.4 rozširuje zložku $\mathcal{L}_{sup}$ na výstupy z pomocných segmentačných hláv na jednotlivých škálach jednej dekóder vetvy. Inšpiráciou je metóda LeFeD [72], kde v podobnej podobe je využívaná tzv. hlboká strata (ang. *deep supervised loss*), ktorá zavádza riadené učenie z anotácií aj do hlbokých vrstiev jednej dekóder vetvy ich architektúry. Ostatné dekóder vetvy sú učené štandardne len z hlavných výstupov vetiev. Myšlienkou za tým je podporiť minimalizáciu nezrovnalostí nielen zavedením rozdielnosti medzi dekódermi na úrovni architektúry ale aj odlišným použitím stratovej funkcie. Zložku $\mathcal{L}_{sup}$ sme na základe opísanej myšlienky rozšírili na výstupy segmentačných hláv dekóder vetvy UP:

$$\mathcal{L}_{sup} = \mathcal{L}_{dice}(P_{TR}^{(4)}, Y) + \mathcal{L}_{foc}(P_{TR}^{(4)}, Y) + \sum_{s=1}^{4} w_s [\mathcal{L}_{dice}(P_{UP}^{(s)}, Y) + \mathcal{L}_{foc}(P_{UP}^{(s)}, Y)] \quad (5.27)$$

Kde pomocou hyperparametra w váhujeme stratu na jednotlivých škálach s. Všetky váhy sme nastavili na $w_c = 1$. V rámci tohto experimentu 2.4 sme taktiež testovali aký má na to vplyv obmena dekóder vetiev, to znamená, že sme aplikovali hlbokú stratu na vetvu TR namiesto UP.



Obr. 5.11: Graf priebehu trénovacích behov a ich vývoja validačného DSC↑ pre experimenty nastavenia plného riadenia na pomocných výstupoch (exp. 2.4 a 2.5).

Experiment 2.5, podobne ako metóda LeFeD [72] , nastavuje pre $\mathcal{L}_{sup}$ v podobe hlbokej straty postupne nižšiu váhu $w = [1, 0.7, 0.4, 0.1]$ na jednotlivých škálach, aby boli pomocné predikcie hlbších vrstiev menej zapájané do tejto straty. Pre tento experiment bola v druhom behu taktiež otestovaná obmena vetiev. V oboch experimentoch aj napriek obmene zostal 1. dekóder (UP) ako vetva validácie.

Priebeh trénovania behov pre exp. 2.4 a 2.5 zobrazuje graf 5.11. Je možné pozorovať, že v exp. 2.5 zníženie váh spomalilo konvergenciu a celkovo obmena dekóderov nemala až taký vplyv na výsledný výkon. Pre porovnanie je zobrazený aj najlepší beh 2.2, kde $\mathcal{L}_{sup}$ sa riadi klasickým nastavením bez hlbokej straty. Jedine exp. 2.4 bol výkonnostne podobný, avšak ďalej do posledného experimentovania sme uprednostnili exp. 2.2, keďže má oveľa jednoduchší tvar pre $\mathcal{L}_{sup}$. Touto fázou sme ukončili experimentovanie s celkovou stratovou funkciou pre DBPNet.

5.8 Experiment metód zväčšovania rozlíšenia

V poslednom experimente č. 3 sme testovali odlišné algoritmy na zväčšovanie rozlíšenia príznakov v rámci dekóder vetiev. Ich vhodný výber je dôležitý, pretože ide v DBPNet o hlavný prvok tvorby odlišných perturbácií na úrovni architektúry dekóderov, čo je podstatnou súčasťou konzistentnej regularizácie. Z pôvodnej konfigurácie sme vyradili transponovanú konvolúciu, ktorá je v 1. dekóder vetve a testovali sme 3 kombinácie tzv. *upsampling* metód na základe odlišného algoritmu interpolácie: bilineárna, bikubická a algoritmus najbližšieho suseda (*nearest neighbour*). Pre každú kombináciu sme otestovali aj obmenu dekóderov, keďže vždy len 1. dekóder je použitý na finálnu inferenciu.

Nasledujúci graf 5.12 zobrazuje priebeh pôvodného nastavenia v predchádzajúcich experimentoch v porovnaní s najlepším behom z tohto experimentu, ktorý mal podobný o niečo nižší výsledný výkon ako pôvodná konfigurácia. Aj napriek tomu,

že transponovaná konvolúcia nie je v súčastnosti už veľmi používaná metóda kvôli šachovnicovým artefaktom [45], tak sa pre náš rámec ukázala byť najlepšou voľbou v pôvodnej kombinácii s bilineárnou interpoláciou.

Obr. 5.12: Graf priebehu trénovacieho experimentu 3 a jeho validačného DSC↑ pre najlepší beh z exp. 3: algoritmus najbližšieho suseda a bilineárna interpolácia.

6. Výsledky

6.1 Kvantitatívne výsledky

V nasledujúcej kapitole zhrnieme výsledky experimentov na zvolených metrikách. Výsledky sú rozdelené podla oblasti, s ktorou sme okolo navrhnutej DBPNet experimentovali: nastavenie čiastočne riadeného učenia a jeho zložiek (kap. 6.1.1), nastavenie plne riadeného učenia a jeho zložiek (kap. 6.1.2), metódy zväčšovania rozlíšenia v rámci dekóder vetiev (kap. 6.1.3). DBPNet s najlepším nastavením vyhodnocujeme v porovnaní s vybranými state-of-the-art riešeniami (kap. 6.1.4).

Výsledky dosiahnutých hodnôt metrík sumarizujeme pre každú skupinu (fázu) experimentov v samostatných tabuľkách. Zaznamenané metriky sú z validačnej fázy, v ktorej model dosiahol najvyššiu hodnotu DSC↑ z pomedzi všetkých iterácií daného behu. Nie sú preto uvádzané najlepšie dosiahnuté hodnoty HD_{95} ↓. Metriku DSC uvádzame v percentuálnej podobe (100% = dokonalý prekryv). Pre lepšiu interpretáciu, HD_{95} môže na začiatku tréningu namerať vzdialenosť až 60.

6.1.1 Experimenty čiastočného riadenia

Pre všetky nasledujúce experimenty bola zložka $\mathcal{L}_{sup}$ rovnaká podľa počiatočného návrhu.

Pre behy experimentov 1. a 2. skupiny je v tabuľkách 6.1 a 6.2 uvádzaná v jednotlivých stĺpcoch:

- Konfigurácia celkovej stratovej funkcie, resp. len čiastočného riadenia, zložka $\mathcal{L}_{sup}$ nie je uvádzaná pretože sa nemenila.
- Metóda generovania pseudo-anotácií (*ostrenie* / *T-softmax*) a prípadne, s ktorou zložkou stratovej funkcie bola použitá.
- Počet iterácií, po ktorých bol inkrementovaný parameter t.
- Použitie všetkých dát (anotovaných aj ne-anotovaných).

Exp:	**Nastavenie:**				**Metriky:**	
č.	Konfigurácia	Ostrenie	t	Všetky dáta	DSC%	HD_{95}
1.1	$\lambda \cdot \mathcal{L}_{con}$	✓	150	✓	83.48	14.41
1.2	$0.1 \cdot \mathcal{L}_{con}$	✓	150	✓	83.55	7.38
1.3	$\lambda \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux}$	✓	150	✓	84.29	9.42
1.4	$\lambda \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux}$	iba $\mathcal{L}_{\text{con}}^{main}$	150	✓	85.28	**4.28**
1.5	$\lambda \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux}$	iba $\mathcal{L}_{\text{con}}^{main}$	100	✓	**86.16**	6.71
1.6	$\lambda \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot 0.5 \cdot \mathcal{L}_{\text{con}}^{aux}$	iba $\mathcal{L}_{\text{con}}^{main}$	100	✓	85.47	4.95
1.7	$\lambda \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux}$	iba $\mathcal{L}_{\text{con}}^{main}$	100	$\mathcal{L}_{\text{con}}^{main}$✓; $\mathcal{L}_{\text{con}}^{aux}$ neanot.	85.38	8.45

Tabuľka 6.1: Výsledky experimentov 1. skupiny: Prototypy založené na vzdialenostnej metrike MSE a generovaní pseudo-anotácií v podobe *ostrenia*.

Exp:	**Nastavenie:**				**Metriky:**	
č.	Konfigurácia	T-softmax	t	Všetky dáta	DSC%	HD_{95}
1.8	$\lambda \cdot \mathcal{L}_{con}$	✓	150	✓	84.98	10.16
1.9	$0.1 \cdot \mathcal{L}_{con}$	✓	150	✓	85.79	5.04
1.10	$\lambda \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux}$	✓	150	✓	**86.48**	**1.93**
1.11	$\lambda \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux}$	✓	100	✓	86.26	3.93
1.12	$\lambda \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot 0.5 \cdot \mathcal{L}_{\text{con}}^{aux}$	✓	150	✓	85.88	4.20

Tabuľka 6.2: Výsledky experimentov 2. skupiny: Prototypy založené na vzdialenostnej metrike KL a generovaní pseudo-anotácií v podobe *T-softmax*.

Je možné konštatovať z exp. 1.3 a 1.10, že pre obe konfigurácie vzdialenostných metrík KL aj MSE bolo výhodné extrahovať regularizáciu pyramídových výstupov a hlavných výstupov do samostatných zložiek straty, vzhľadom na dodatočný

výkon a možnosť bližšieho nastavovania. Zároveň, pre učiaci rámec DPBNet je výhodnejšie regularizovať konzistenciu za pomoci KL divergencie a *T-softmax* (na všetkých škálach). Toto nastavenie dosiahlo vyšší prírastok výkonu už prvotnou konfiguráciou oproti metrike MSE a operácii *ostrenia* (viď porovnanie exp. 1.1 a 1.8) a pokračujúc aj v ďalších experimentoch.

Do čiastočne riadeného učenia a jeho zložiek sa ukázalo byť výhodnejšie zapájať všetky dáta (anotované aj ne-anotované), čo dokazuje horší výsledok exp. 1.7.

Exp:	**Nastavenie:**				**Metriky:**	
č.	Konfigurácia	$\mathcal{L}_{\text{con}}^{main}$	$\mathcal{L}_{\text{con}}^{aux}$	t	DSC%	HD_{95}
1.13	$0.1 \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux}$	*KL, T-softmax*	*KL, T-softmax*	150	**86.91**	**3.77**
1.14	$0.1 \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux}$	*KL, T-softmax*	MSE	150	86.54	3.81

Tabuľka 6.3: Výsledky experimentov 3. skupiny: Prototypy kombinujúce rôzne nastavenie váhovania a vzdialenostných metrík.

Alternatívou je regularizovať kombinovaným prístupom za použitia KL a *T-softmax* pre hlavné výstupy ($\mathcal{L}_{\text{con}}^{main}$) a MSE bez pseudo-anotácií pre pyramídové škály ($\mathcal{L}_{\text{con}}^{aux}$). Tento prístup mal v rovnakej konfigurácii podobné výsledky ako prístup využívajúci KL a *T-softmax* vo všetkých zložkách (viď. tabuľku 6.3).

Z hľadiska nastavenia váh bola výkonovo lepšia kombinácia fixnej váhy pre $\mathcal{L}_{\text{con}}^{main}$ a gaussoveho váhovania pre $\mathcal{L}_{\text{con}}^{aux}$ (ako v experimente č.1.13).

Exp:	**Nastavenie:**				**Metriky:**	
č.	Konfigurácia	$\mathcal{L}_{\text{con}}^{main}$	$\mathcal{L}_{\text{con}}^{aux}$	t	DSC%	HD_{95}
1.15	$0.1 \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux} + 0.1 \cdot \mathcal{L}_{um}$	*KL, T-softmax*	*KL, T-softmax*	150	**87.13**	**2.50**
1.16	$0.1 \cdot \mathcal{L}_{\text{con}}^{main} + \lambda \cdot \mathcal{L}_{\text{con}}^{aux} + \lambda \cdot \mathcal{L}_{um}$	*KL, T-softmax*	*KL, T-softmax*	150	86.86	2.97

Tabuľka 6.4: Výsledky experimentov 4. skupiny: Prototypy s pridanou zložkou minimalizácie entropie.

Pridanie zložky minimalizácie entropie s konštantnou váhou 0.1 dodatočne navýšilo výkon o len 0.2 DSC% ale rýchlosť konvergencie modelu sa zvýšila.

Celkovo je možnosť z výsledkov pozorovať, že prototypovaním s čiastočným riadením sa navýšil segmentačný výkon od prvotného návrhu (exp 1.1) až po exp 1.15 približne o 3.65 DSC% a 11.9 HD_{95} na validačnej sade.

6.1.2 Experimenty plného riadenia

Pre nasledujúce skupiny experimentov uvádzame v tabuľkách 6.5 a 6.6 konfiguráciu zložky plného riadenia $\mathcal{L}_{sup}$ vrátane stratových funkcií a hyperparametrov, ktoré boli použité a nastavované. Konfigurácia *loss-sup* označuje štandardné učenie na hlavných výstupoch vetiev a *loss-sup-deep* označuje rozšírenie tohto učenia v podobe opísanej hlbokej straty (kapitola 5.7.2). Stĺpec *dekóder* označuje vetvu, na ktorú bola aplikovaná časť $\mathcal{L}_{sup}$ v podobe hlbokej straty.

Exp:	**Nastavenie:**			**Metriky:**	
č.	Konfigurácia	γ	α	DSC%	HD_{95}
1.15	*loss-sup:* $\mathcal{L}_{dice}$	-	-	87.13	2.50
2.1	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{ce}$	-	-	86.97	2.06
2.2	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	1.0	-	86.91	3.80
2.2	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	2.0	-	87.12	3.31
2.2	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	2.5	-	**87.59**	**1.69**
2.2	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	3.0	-	86.90	2.46
2.3	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	2.5	[0.004, 0.343, 0.325, 0.326]	86.80	2.58

Tabuľka 6.5: Výsledky experimentov 5. skupiny: Prototypy nastavenia zložky plného riadenia.

Exp:	**Nastavenie:**					**Metriky:**	
č.	Konfigurácia	γ	α	w	dekóder	DSC%	HD_{95}
2.4	*loss-sup-deep:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	2.5	-	[1,1,1,1]	1.	87.18	**1.84**
2.4	*loss-sup-deep:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	2.5	-	[1,1,1,1]	2.	**87.58**	2.73
2.5	*loss-sup-deep:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	2.5	-	[1,0.7,0.4,0.1]	1.	86.37	5.64
2.5	*loss-sup-deep:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	2.5	-	[1,0.7,0.4,0.1]	2.	86.40	3.10

Tabuľka 6.6: Výsledky experimentov 6. skupiny: Prototypy nastavenia zložky plného riadenia v podobe hlbokej straty, dice koeficientu a fokálnej straty.

Do plne riadeného učenia sa ukázalo byť výhodné zapojiť fokálnu stratu s $\gamma = 2.5$ k už dovtedy používanej strate $\mathcal{L}_{dice}$ a tým získať aspoň 0.5% výkonu na DSC↑.

Behy experimentov konfigurácie *loss-sup-deep* boli výkonnejšie s váhovaním jednotlivých škál $w = [1, 1, 1, 1]$. Zároveň sa ukázalo, že na obmene dekóderov až tak nezáležalo. Beh experimentu 2.4 dosiahol na druhom dekodéri podobný výkon ako jednoduchšia *loss-sup* kde $\gamma = 2.5$. Preto sme uprednostnili ďalej používať štandardnú $\mathcal{L}_{sup}$ v nastavení s $\mathcal{L}_{dice}, \mathcal{L}_{focal}$ a $\gamma = 2.5$ (exp. 2.2).

6.1.3 Experiment metód zväčšovania rozlíšenia

Nasledujúca tabuľka zobrazuje kombinácie metód, ktoré sme testovali. Vo výsledku sme v DBPNet ponechali jednotlivo v dekóder vetvách ich pôvodné metódy: bilineárnu interpoláciu (UP) a transponovanú konvolúciu (TP).

Exp:	**Nastavenie:**			**Metriky:**	
č.	Konfigurácia	Metóda (1. dekóder)	Metóda (2. dekóder)	DSC%	HD_{95}
2.2	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	bilinear	transpose	**87.59**	**1.69**
3.1	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	nearest	bilinear	87.45	1.92
3.1	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	bilinear	nearest	86.76	2.19
3.2	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	bilinear	bicubic	86.83	2.52
3.2	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	bicubic	bilinear	86.87	2.38
3.3	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	bicubic	nearest	86.29	2.96
3.3	*loss-sup:* $\mathcal{L}_{dice}$, $\mathcal{L}_{focal}$	nearest	bicubic	86.77	1.96

Tabuľka 6.7: Výsledky posledného experimentu 3: kombinácie metód zväčšovania rozlíšenia v rámci dekóder vetiev.

6.1.4 Porovnanie s existujúcimi riešeniami

Navrhnutý prototyp DPBNet sme porovnali na testovacej sade ACDC [5] datasetu so súčasnými metódami stavu poznania čiastočne riadenej segmentácie medicínskych snímok. Ide o metódy, ktoré taktiež rôznym spôsobom pristupujú k konzistentnej regularizácii.

DBPNet sme nastavili podľa experimentu 2.2 kde využíva $\mathcal{L}_{sup}$ v konfigurácii s $\mathcal{L}_{dice}, \mathcal{L}_{foc}, \gamma = 2.5$ a konzistentnú regularizáciu na základe KL divergencie, *T-softmax* a minimalizácie entropie (kapitola 5.6.4.1). Pre demonštráciu schopnosti

čiastočne riadeného učenia sú metódy trénované pri 10% a 20% anotáciách v porovnaní so štandardným U-Net [49], ktorý je základom všetkých z metód. Výsledky U-Net sú po tréningu v plne riadenom režime za pomoci len 10% alebo všetkých (100%) anotácií. Zložitosť modelov je uvádzaná počas inferencie.

Metóda:	**Počet**	**Metriky:**				**Zložitosť:**
	anotácií:	DSC(%)↑	IoU(%)↑	HD_{95}↓	ASD↓	Para.(M)
U-Net	10%	77.34	66.20	9.18	2.45	1.81
U-Net	20%	85.15	75.48	6.20	2.12	1.81
U-Net	100%	91.65	84.93	1.89	0.56	1.81
UA-MT (2019) [71]	10%	81.58	70.48	12.35	3.62	1.81
SASSNet (2020) [32]	10%	84.14	74.09	5.03	1.40	1.81
DTC (2021) [38]	10%	82.71	72.14	11.31	2.99	1.81
URPC (2021) [37]	10%	81.77	70.85	5.04	1.41	1.83
MC-Net (2021) [66]	10%	86.34	76.82	7.08	2.08	2.58
MC-Net+ (2022) [65]	10%	87.10	78.06	6.68	2.00	1.81
DVCPS (2025) [70]	10%	**88.76**	**80.36**	5.03	1.43	–
DBPNet	10%	88.52	79.99	**3.41**	**1.01**	1.83
UA-MT (2019) [71]	20%	85.87	76.78	5.06	1.54	1.81
SASSNet (2020) [32]	20%	87.04	78.13	7.84	2.15	1.81
DTC (2021) [38]	20%	86.28	77.03	6.14	2.11	1.81
URPC (2021) [37]	20%	85.07	75.61	6.26	1.77	1.83
MC-Net (2021) [66]	20%	87.83	79.14	4.94	1.52	2.58
MC-Net+ (2022) [65]	20%	88.51	80.19	5.35	1.54	1.81
DBPNet	20%	**89.15**	**81.03**	**4.14**	**1.17**	1.83

Tabuľka 6.8: Porovnanie výsledkov navrhnutého DBPNet so SOTA metódami po trénovaní za pomoci 10% anotácií testovacej sady ACDC datasetu [5]

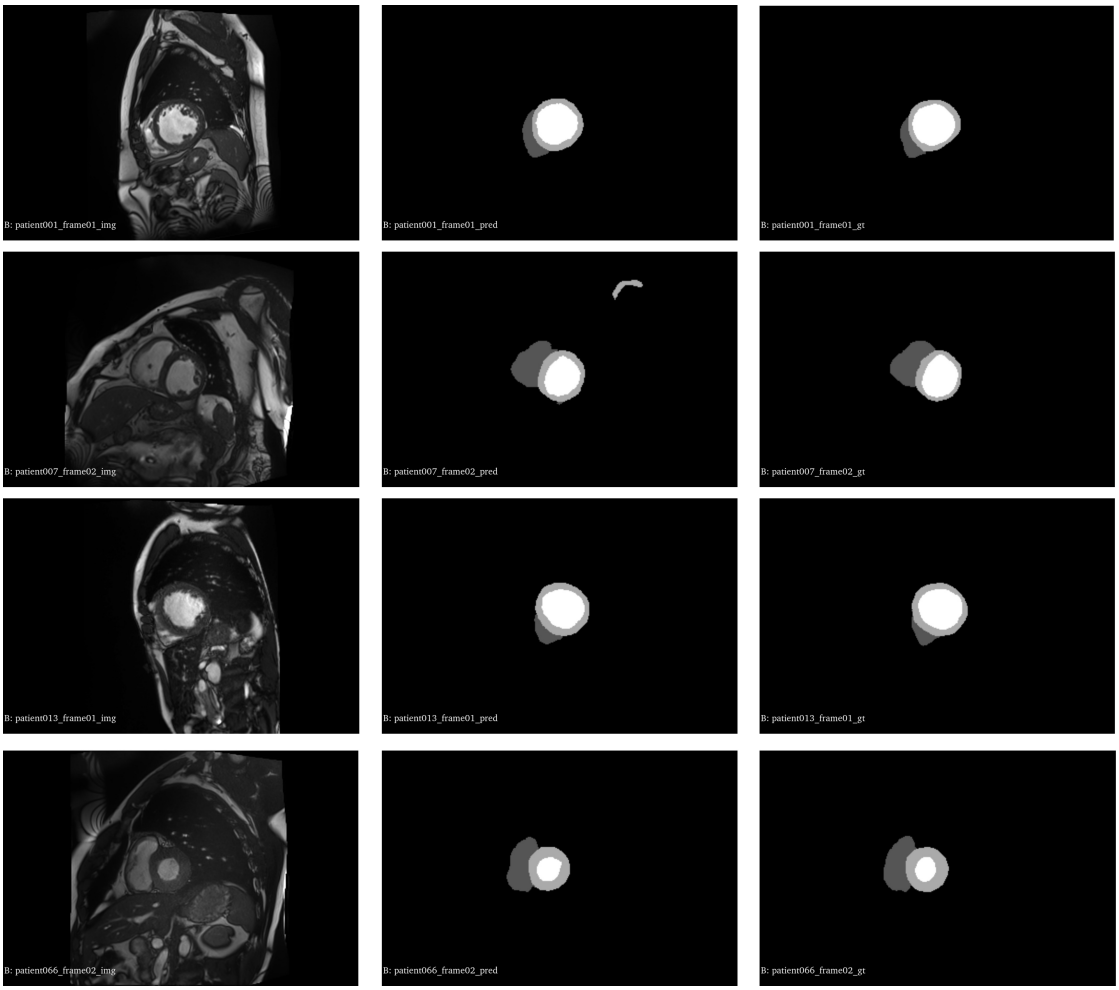
Výsledky metód boli prevzaté od MTNet+ [65] pričom DBPNet bol pre férové porovnanie trénovaný na rovnakej testovacej sade a pri rovnakých nastaveniach, vrátane generátorov náhodných čísel, okrem dĺžky trénovania. DBPNet konverguje až približne pri 50k iteráciách oproti ostatným metódam (30k). Výnimkou je metóda DVCPS [70] s podobnými nastaveniami tréningu, ktorých výsledok sme prevzali od ich autorov. Viac o nastaveniach sme opísali v kapitole 5.4.

DBPNet model dosiahol v porovnaní so súvisiacou prácou MTNet+ (s 3 dekódermi) lepší výkon pričom využíva 2 dekóder vetvy. Preto vyžaduje menší počet parametrov počas tréningu. Naopak má DBPNet komplexnejšiu stratovú funkciu,

ktorá môže vyžadovať dôkladnejšie nastavovanie a dlhší tréning. Aj napriek tomu, že sme prototypovali na základe metriky DSC ↑ tak DBPNet dosiahol najlepšie výsledky na boundary-based metrikách HD_{95} ↓ a ASD↓.

6.2 Kvalitatívne výsledky

Príklady výsledkov segmentácie testovacej sady ACDC datasetu navrhnutým a porovnávaným modelom DBPNet.



Obr. 6.1: Príklady segmentácie 2D rezov ACDC datasetu pomocou modelu DBP-Net trénovaného za pomoci 10% anotácií: 1. stĺpec vstupný rez, 2. stĺpec predikovaná maska modelom, 3. stĺpec anotačná maska.

7. Zhodnotenie

V našej diplomovej práci sme na základe analýzy hlbokého učenia a jej metód segmentácie medicínskych obrazových dát navrhli a implementovali vlastnú metódu segmentácie medicínskych snímok. Ide o hybridný prístup čiastočne riadenej segmentácie v podobe modelu DBPNet a jeho učiaceho rámca založenom na regularizácii konzistencie na úrovni architektúry, za účelom učenia sa z veľkého množstva ne-anotovaných dát a malého množstva anotácií.

Architektúru DBPNet sme postavili na základe 2 riešení: Modelu MCNet+[65], ktorý sme zjednodušili na 2 dekódery a rozšírili o pomocné segmentačné hlavy od URPC [40]. Pre DBPNet sme navrhli učiaci rámec, ktorý kombinuje existujúci prístup regularizácie konzistencie medzi hlavnými výstupmi dekóderov [65, 75] s vlastným prístupom regularizácie pomocných pyramídových výstupov naprieč dvoma dekódermi. Pre učiaci rámec sme navrhli prvotný prototyp stratovej funkcie využívajúci strednú kvadratickú chybu (MSE) a operáciu *ostrenia*. DBPNet a prvotný prototyp stratovej funkcie sme otestovali a porovnali so súvisiacimi riešeniami na ACDC datasete [5].

So stratovou funkciou sme ďalej experimentovali vo viacerých skupinách v podobe úprav a modifikácií. Skupiny experimentov sme zamerali najskôr okolo zložiek čiastočne riadeného učenia, ktoré využívajú anotované aj ne-anotované dáta. Výsledkom bola nová konfigurácia, ktorá regularizuje konzistenciu na základe KL

divergencie, operácie *T-softmax* a existujúcej zložky minimalizácie entropie [75]. Následne sme experimentovali so zložkou plne riadeného učenia, do ktorej sme vo výsledku zapojili fokálnu stratu [35] k strate dice koeficientom. Nakoniec sme testovali rôzne kombinácie metód zväčšovania rozlíšenia v dekóder vetvách na vhodné nastavenie regularizácie.

Prototyp s nastaveniami najlepších dosiahnutých výsledkov sme porovnali s existujúcimi riešeniami za pomoci verejného ACDC datasetu na vybraných metrikách. DBPNet prekonáva vo výsledných metrikách na testovacej sade pôvodné modely, z ktorých riešenie vychádzalo a má podobné výsledky ako súčasné riešenia pri použití 10% alebo 20% anotácií pri tréningu.

7.0.1 Možné vylepšenia

Učiaci rámec DBPNet by bolo možné namiesto U-Net základu otestovať na modernejších a súčasných architektonických blokoch hlbokého učenia [14, 7, 56] ako napríklad Mamba-UNet. [62]

Literatúra

[1] Charu C. Aggarwal. *Neural Networks and Deep Learning: A Textbook (Second Edition)*. 2nd. Springer, 2023. DOI: 10.1007/978-3-031-29642-0. URL: https://link.springer.com/book/10.1007/978-3-031-29642-0.

[2] Charu C Aggarwal et al. "Neural networks and deep learning". In: *Springer* 10.978 (2018), s. 3.

[3] Michela Antonelli et al. "The medical segmentation decathlon". In: *Nature communications* 13.1 (2022), s. 4128.

[4] Dor Bank, Noam Koenigstein a Raja Giryes. "Autoencoders". In: *Machine learning for data science handbook: data mining and knowledge discovery handbook* (2023), s. 353–374.

[5] Olivier Bernard et al. "Deep learning techniques for automatic MRI cardiac multi-structures segmentation and diagnosis: is the problem solved?" In: *IEEE transactions on medical imaging* 37.11 (2018), s. 2514–2525.

[6] Patrick Bilic et al. "The liver tumor segmentation benchmark (lits)". In: *Medical Image Analysis* 84 (2023), s. 102680.

[7] Hu Cao et al. "Swin-unet: Unet-like pure transformer for medical image segmentation". In: *European conference on computer vision*. Springer. 2022, s. 205–218.

[8] Liang-Chieh Chen et al. "Encoder-decoder with atrous separable convolution for semantic image segmentation". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, s. 801–818.

[9] Liang-Chieh Chen et al. "Rethinking atrous convolution for semantic image segmentation". In: *arXiv preprint arXiv:1706.05587* (2017).

[10] Xiaokang Chen et al. "Semi-supervised semantic segmentation with cross pseudo supervision". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, s. 2613–2622.

[11] Özgün Çiçek et al. "3D U-Net: learning dense volumetric segmentation from sparse annotation". In: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*. Springer. 2016, s. 424–432.

[12] Qian Da et al. "DigestPath: A benchmark dataset with challenge review for the pathological detection and segmentation of digestive-system". In: *Medical Image Analysis* 80 (2022), s. 102485.

[13] Bin Ding, Huimin Qian a Jun Zhou. "Activation functions and their characteristics in deep neural networks". In: *2018 Chinese control and decision conference (CCDC)*. IEEE. 2018, s. 1836–1841.

[14] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: `2010.11929 [cs.CV]`.

[15] Ian Goodfellow, Yoshua Bengio a Aaron Courville. *Deep Learning*. `http://www.deeplearningbook.org`. MIT Press, 2016.

[16] Yanming Guo et al. "Deep learning for visual understanding: A review". In: *Neurocomputing* 187 (2016), s. 27–48.

[17] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, s. 770–778.

[18] Nicholas Heller et al. “The kits19 challenge data: 300 kidney tumor cases with clinical context, ct semantic segmentations, and surgical outcomes”. In: *arXiv preprint arXiv:1904.00445* (2019).

[19] Dan Hendrycks a Kevin Gimpel. “Gaussian error linear units (gelus)”. In: *arXiv preprint arXiv:1606.08415* (2016).

[20] Nabil Ibtehaz a M Sohel Rahman. “MultiResUNet: Rethinking the U-Net architecture for multimodal biomedical image segmentation”. In: *Neural networks* 121 (2020), s. 74–87.

[21] Sergey Ioffe a Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *International conference on machine learning*. pmlr. 2015, s. 448–456.

[22] *ISLES Challenge - Ischemic Stroke Lesion Segmentation*. `https://www.isles-challenge.org/`. Accessed on December 5, 2023.

[23] Rushi Jiao et al. “Learning with limited annotations: a survey on deep semi-supervised learning for medical image segmentation”. In: *Computers in Biology and Medicine* 169 (2024), s. 107840.

[24] Qiangguo Jin et al. “Semi-supervised histological image segmentation via hierarchical consistency enforcement”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2022, s. 3–13.

[25] Khuyen Le. *An overview of VGG16 and NiN models*. `https://lekhuyen.medium.com/an-overview-of-vgg16-and-nin-models-96e4bf398484`, Last accessed on 2024-04-22. 2021.

[26] Diederik P Kingma a Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[27] Dani Kiyasseh et al. “Segmentation of left atrial MR images via self-supervised semi-supervised meta-learning”. In: *Medical Image Computing and Computer*

Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part II 24. Springer. 2021, s. 13–24.

[28] Neeraj Kumar et al. "A multi-organ nucleus segmentation challenge". In: *IEEE transactions on medical imaging* 39.5 (2019), s. 1380–1391.

[29] Yann LeCun, Yoshua Bengio a Geoffrey Hinton. "Deep learning". In: *nature* 521.7553 (2015), s. 436–444.

[30] Chaoqun Li, Liejun Wang a Shuli Cheng. "Enhanced transformer encoder and hybrid cascaded upsampler for medical image segmentation". In: *Expert Systems with Applications* 238 (2024), s. 121965.

[31] Pengzhi Li, Yan Pei a Jianqiang Li. "A comprehensive survey on design and application of autoencoder in deep learning". In: *Applied Soft Computing* 138 (2023), s. 110176.

[32] Shuailin Li, Chuyu Zhang a Xuming He. "Shape-aware semi-supervised 3D semantic segmentation for medical images". In: *Medical Image Computing and Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, October 4–8, 2020, Proceedings, Part I 23*. Springer. 2020, s. 552–561.

[33] Xiaomeng Li et al. "Transformation-consistent self-ensembling model for semisupervised medical image segmentation". In: *IEEE transactions on neural networks and learning systems* 32.2 (2020), s. 523–534.

[34] Zewen Li et al. "A survey of convolutional neural networks: analysis, applications, and prospects". In: *IEEE transactions on neural networks and learning systems* 33.12 (2021), s. 6999–7019.

[35] Tsung-Yi Lin et al. "Focal loss for dense object detection". In: *Proceedings of the IEEE international conference on computer vision*. 2017, s. 2980–2988.

[36] Zhuang Liu et al. "A convnet for the 2020s". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, s. 11976–11986.

[37] Xiangde Luo et al. "Efficient semi-supervised gross target volume of nasopharyngeal carcinoma segmentation via uncertainty rectified pyramid consistency". In: *Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part II 24*. Springer. 2021, s. 318–329.

[38] Xiangde Luo et al. "Semi-supervised medical image segmentation through dual-task consistency". In: *Proceedings of the AAAI conference on artificial intelligence*. Zv. 35. 10. 2021, s. 8801–8809.

[39] Xiangde Luo et al. "Semi-supervised medical image segmentation via cross teaching between cnn and transformer". In: *International conference on medical imaging with deep learning*. PMLR. 2022, s. 820–833.

[40] Xiangde Luo et al. "Semi-supervised medical image segmentation via uncertainty rectified pyramid consistency". In: *Medical Image Analysis* 80 (2022), s. 102517.

[41] Bjoern H Menze et al. "The multimodal brain tumor image segmentation benchmark (BRATS)". In: *IEEE transactions on medical imaging* 34.10 (2014), s. 1993–2024.

[42] Bjoern H Menze et al. *The Multimodal Brain Tumor Segmentation Benchmark (BRATS)*. `https://www.med.upenn.edu/sbia/brats2018/data.html`. 2018.

[43] Fausto Milletari, Nassir Navab a Seyed-Ahmad Ahmadi. "V-net: Fully convolutional neural networks for volumetric medical image segmentation". In: *2016 fourth international conference on 3D vision (3DV)*. Ieee. 2016, s. 565–571.

[44] Michael A Nielsen. *Neural networks and deep learning*. Zv. 25. Determination press San Francisco, CA, USA, 2015.

[45] Augustus Odena, Vincent Dumoulin a Chris Olah. "Deconvolution and checkerboard artifacts". In: *Distill* 1.10 (2016), e3.

[46] Jizong Peng et al. "Deep co-training for semi-supervised image segmentation". In: *Pattern Recognition* 107 (2020), s. 107269.

[47] Imran Qureshi et al. "Medical image segmentation using deep semantic-based methods: A review of techniques, applications and emerging trends". In: *Information Fusion* 90 (2023), s. 316–352.

[48] Prajit Ramachandran, Barret Zoph a Quoc V Le. "Searching for activation functions". In: *arXiv preprint arXiv:1710.05941* (2017).

[49] Olaf Ronneberger, Philipp Fischer a Thomas Brox. "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*. Springer. 2015, s. 234–241.

[50] Frank Rosenblatt. "The perceptron: a probabilistic model for information storage and organization in the brain." In: *Psychological review* 65.6 (1958), s. 386.

[51] Holger R Roth et al. "Deeporgan: Multi-level deep convolutional networks for automated pancreas segmentation". In: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part I 18*. Springer. 2015, s. 556–564.

[52] David E Rumelhart, Geoffrey E Hinton a Ronald J Williams. "Learning representations by back-propagating errors". In: *nature* 323.6088 (1986), s. 533–536.

[53] Maohao Shen et al. “Labeling cost sensitive batch active learning for brain tumor segmentation”. In: *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. IEEE. 2021, s. 1269–1273.

[54] Ivan Nunes Silva et al. “Artificial neural networks: a practical course”. In: *Springer* (2016).

[55] Karen Simonyan a Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[56] Mingxing Tan a Quoc Le. “Efficientnet: Rethinking model scaling for convolutional neural networks”. In: *International conference on machine learning*. PMLR. 2019, s. 6105–6114.

[57] Antti Tarvainen a Harri Valpola. “Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results”. In: *Advances in neural information processing systems* 30 (2017).

[58] *The Cancer Imaging Archive*. `https://www.cancerimagingarchive.net/`. Accessed on December 5, 2023.

[59] Athanasios Voulodimos et al. “Deep learning for computer vision: A brief review”. In: *Computational intelligence and neuroscience* 2018 (2018).

[60] Qi Wang et al. “A comprehensive survey of loss functions in machine learning”. In: *Annals of Data Science* (2020), s. 1–26.

[61] Risheng Wang et al. “Medical image segmentation using deep learning: A survey”. In: *IET Image Processing* 16.5 (2022), s. 1243–1267.

[62] Ziyang Wang et al. “Mamba-unet: Unet-like pure visual mamba for medical image segmentation”. In: *arXiv preprint arXiv:2402.05079* (2024).

[63] M Arif Wani et al. *Advances in deep learning*. Springer, 2020.

[64] Jianxin Wu. “Introduction to convolutional neural networks”. In: *National Key Lab for Novel Software Technology. Nanjing University. China* 5.23 (2017), s. 495.

[65] Yicheng Wu et al. “Mutual consistency learning for semi-supervised medical image segmentation”. In: *Medical Image Analysis* 81 (2022), s. 102530.

[66] Yicheng Wu et al. “Semi-supervised left atrium segmentation with mutual consistency training”. In: *Medical image computing and computer assisted intervention–MICCAI 2021: 24th international conference, Strasbourg, France, September 27–October 1, 2021, proceedings, part II 24*. Springer. 2021, s. 297–306.

[67] Yuxin Wu a Kaiming He. “Group normalization”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, s. 3–19.

[68] Yutong Yan et al. “Deep active learning for dual-view mammogram analysis”. In: *Machine Learning in Medical Imaging: 12th International Workshop, MLMI 2021, Held in Conjunction with MICCAI 2021, Strasbourg, France, September 27, 2021, Proceedings 12*. Springer. 2021, s. 180–189.

[69] Michael Yeung et al. “Unified focal loss: Generalising dice and cross entropy-based losses to handle class imbalanced medical image segmentation”. In: *Computerized Medical Imaging and Graphics* 95 (2022), s. 102026.

[70] Baoqi Yu et al. “Semi-supervised medical image segmentation with diverse views via cross-pseudo supervision”. In: *Signal, Image and Video Processing* 19.1 (2025), s. 1–15.

[71] Lequan Yu et al. “Uncertainty-aware self-ensembling model for semi-supervised 3D left atrium segmentation”. In: *Medical image computing and computer assisted intervention–MICCAI 2019: 22nd international conference, Shenzhen, China, October 13–17, 2019, proceedings, part II 22*. Springer. 2019, s. 605–613.

[72] Qingjie Zeng et al. “Discrepancy matters: Learning from inconsistent decoder features for consistent semi-supervised medical image segmentation”. In: *arXiv preprint arXiv:2309.14819* (2023).

[73] Aston Zhang et al. "Dive into deep learning". In: *arXiv preprint arXiv:2106.11342* (2021).

[74] Han Zheng et al. "Semi-supervised segmentation of liver using adversarial learning with deep atlas prior". In: *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part VI 22*. Springer. 2019, s. 148–156.

[75] Lanfeng Zhong et al. *Semi-supervised Pathological Image Segmentation via Cross Distillation of Multiple Attentions*. 2023. arXiv: 2305.18830 [cs.CV].

[76] Zongwei Zhou et al. "Unet++: Redesigning skip connections to exploit multiscale features in image segmentation". In: *IEEE transactions on medical imaging* 39.6 (2019), s. 1856–1867.