Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra kybernetiky

Bakalářská práce

Automatická segmentace satelitních snímků

Plzeň 2023 Pavel Balda

Místo této strany bude
zadání práce.

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 21. května 2023

Pavel Balda

Abstract

The task of automatic segmentation of satellite images finds application in many areas of modern research. The first part of thesis presents possible approaches to the problem and available data. The majority of the work then focuses on approaches to semantic segmentation of satellite images on the LoveDA benchmark dataset. Within the scope of this work, selected methods with manual feature extraction are presented, along with their results on the chosen dataset. Subsequently, thesis describes a gradual development of deep-learning methods, with evaluation in the LoveDA Semantic Segmentation competition. The main outputs of the thesis consist of a summary of the problem, statistics of the methods used, and a functional segmentation model of a neural network.

Abstrakt

Úloha automatické segmentace satelitních snímků nachází využití v mnoha oblastech moderního výzkumu. V první části této práce jsou prezentovány možné přístupy k problematice a dostupná data. Většina práce se poté věnuje přístupům k sémantické segmentaci satelitních snímků na benchmarkovém datasetu LoveDA, kdy jsou v rámci práce prezentovány nejprve vybrané metody s ruční extrakcí příznaků a jejich výsledky na vybraném datasetu a následně postupný vývoj metod hlubokého učení umělých neuronových sítí s vyhodnocením v soutěži LoveDA Semantic Segmentation. Hlavní výstupy práce tvoří shrnutí problematiky, statistiky použitých metod a funkční segmentační model neuronové sítě.

Obsah

1 Úvod

S vývojem technologií, které nám umožňují lépe zkoumat naše životní prostředí se všemi jeho jevy a procesy, se také zlepšují technologie zachycování zemského povrchu. Se zvyšováním kvality získávaných leteckých a satelitních snímků dochází stále častěji k jejich využití v četných odvětvích lidské činnosti. Množství dostupných dat stále roste a satelitní obrazová data některých oblastí jsou analyzována na denní bázi. Metody automatické segmentace satelitních snímků umožňují provádět široké spektrum analýz satelitních snímků bez nutnosti jejich manuálního zpracování. Může tak docházet ke zpracovávání velkého množství satelitních dat a tím i ke zlepšování poznání mnoha demografických a přírodních jevů.

Existuje mnoho přístupů k automatické segmentaci. V této práci bude shrnuta problematika a motivace automatické segmentace satelitních snímků a možnost využití dostupných datasetů. Na vybraném datasetu se bude práce věnovat sémantické segmentaci pomocí klasických (LBP, HOG, RGB) a moderních (hluboké učení neuronových sítí) metod umělé inteligence a jejich porovnání.

Cílem této práce je seznámit se s problematikou automatické segmentace satelitních snímků a shrnout dostupné datasety pro hluboké učení. Dále je účelem této práce je na vybraném datasetu porovnat segmentační metody s ručně extrahovanými příznaky s moderní metodou umělých neuronových sítí. Výstupem práce by mělo být přehledné shrnutí problematiky, vyhodnocení segmentačních metod a modelů neuronových sítí, zhodnocení jejich silných a slabých stránek a jejich celkový přínos problematice.

1.1 Segmentace

Segmentace je v počítačovém vidění úloha, při níž se snažíme rozdělit prostor obrázku (snímku) na relevantní objekty nebo sémantické třídy. V kontextu zpra-

cování digitalizovaného obrazu se jedná o úlohu klasifikace, kdy je předmětem klasifikace každý pixel (případně skupina pixelů) v obrázku.

Sémantická segmentace je úloha, při níž rozdělujeme prostor obrázku do tříd podle sémantické příslušnosti. Na příkladu satelitních snímků tedy klasifikujeme pixely každého domu do třídy *Budova* a pixely každé silnice do třídy *Silnice*.

Segmentace instancí si klade za cíl rozlišit od sebe jednotlivé objekty v obraze na základě jejich vlastností. Na příkladu satelitních snímků tedy klasifikujeme všechny pixely do tříd *Objekt1*, *Objekt2*, ..., *objektN*.

Panoptická segmentace je poté přístup kombinující úlohu segmentace sémantické a segmentace instancí. Úloha panoptické segmetnace znamená rozlišit nejen třídu (jako u sémantické segmentace), ale také jednotlivé objekty v rámci této třídy. V našem příkladu bychom tedy pixely každého domu, nebo silnice chtěli přiřadit do objektů např. *Budova1*, *Budova2*, *Silnice1* atd. podle toho, k jaké budově, nebo k jaké silnici jednotlivé pixely náleží. Ze své podstaty se jedná o úlohu na řešení složitější.

Pojmem automatická segmentace se poté rozumí segmentace prováděná umělým algoritmem, který je schopen vstupnímu obrázku vytvořit masku popisující příslušnost k jednotlivým třídám (resp. objektům) jednotlivých pixelů.

V této bakalářské práci se zaměřím na algoritmy sémantické segmentace celého prostoru snímků do více tříd.

1.2 Satelitní snímky

Satelitní snímek je fotografie zemského povrchu z umělé družice. Satelitní snímky jsou často schopny zachytit cenné informace o zemském povrchu, povětrnostních vlivech a demografických faktorech z makroskopického hlediska. Průběžným sledováním satelitních snímků stejné oblasti lze také pozorovat ekologické, kulturní i jiné trendy a lze monitorovat a předpovídat řadu dalších makroskopických dějů.

Satelitní snímky mohou mít různé rozlišení. Typické rozlišení se pohybuje od

desítek metrů na pixel (např. satelity Landsat-1, Landsat-2, ASTER) do desítek centimetrů na pixel (většinou komerční využití - satelity WorldView-3, GeoEye-1, Pleiades-HR) [1].

Technika pořizování a technologie zpracování satelitních snímků stále pokračuje, následkem čehož jsme svědky stále většího rozlišení satelitních snímků a obecně kvalitnějších satelitních dat. Jedním z nejvýznamějších projektů v této oblasti je dozajista Google Earth [2], který na základě pořízených snímků z více zdrojů sestavil velmi detailní 3D model zemského povrchu určitých oblastí.

1.2.1 Využití satelitních snímků

Satelitní snímky lze použít k řešení velkého množství problémů a mohou sloužit jako zdroj cenných podpůrných informací v mnoha rozdílných odvětvích. Lze je využít pro předpověď počasí, kdy se využívá jejich schopnost zachytit mračna a vývoj a směr jejich pohybu. Dalším významným využitím je oblast zemědělství. Pomocí satelitních snímků jde posuzovat kvalitu a složení půdy a v dlouhodobém měřítku dopady eroze a odhady budoucích výnosů. Nemalé uplatnění mají satelitní snímky pro monitoring přírodních katastrof, např. zjišťování následků záplav, postup tání ledovců, nebo vývoj rozsáhlých požárů, nebo ropných skvrn. Dále se dá tato technologie použít k vojenským účelům pro špionáž a průzkum krajiny, kdy je možné zjistit např. prostupnost terénu, ale i rozložení nepřátelských sil. Noční snímky se pak dají použít k detekci významných energetických, civilních i vojenských uzlů. Dobrým příkladem je také možnost využití satelitních snímků v regionálním územním plánování, kdy jsou satelitní snímky často součástí vizualizací a studií. Nejen s postupující kvalitou pak satelitní snímky získávají uplatnění ve velmi širokém spektru oblastí, jako jsou například archeologie, kartografie, lesnictví, vzdělání a v mnoha dalších.

Některé satelity umožňují pořizování satelitních snímků v jiných spektrech než je spektrum viditelného světla (např. snímání infračervených a ultrafialových frekvencí). To obecně značně rozšiřuje možnosti využití satelitních snímků i vy-

užití metod jejich sémantické segmentace.

1.2.2 Využití sémantické segmentace

Sémantická segmentace satelitních snímků nachází využití v mnoha z výše zmíněných aplikacích. Za zopakování a upřesnění stojí sledování a analýza změn a stavů v krajině (např. odlesňování, rozšiřování pouští, polní eroze, udržení vody v krajině), sledování vývoje a stavu biodiverzity oblastí (např. složení lesů, projevy přírodní zpětné vazby při vymizení živočišného druhu atd.), monitorování přírodních katastrof (povodně, požáry, sopečné erupce, dopady zemětřesení, tání ledovců), sledování dopravních sítí a jejich stavu (mosty, silnice, polní cesty, železnice), zemědělství (analýza produkce plodin, katastr nemovitostí, plánování zásahů a udržitelný rozvoj) a městská infrastruktura (plánování vhodných míst k výstavbě infrastruktury).

1.3 Vývojové technologie a přístupný kód

Algoritmy v této práci vyvíjím v programovacím jazyku Python [3]. Dvěma stěžejními knihovnami jsou Pytorch [4] a Scikit-learn [5]. Jako IDE používám nejnovější verzi PyCharm Community Edition. Pro trénování neuronových sítí používám osobní počítač. Při trénování jsem využil paralelní výpočetní platformu CUDA [6] vyvinutou společností NVIDIA a umožňující znatelné zrychlení trénovacího procesu umělých neuronových sítí.

Zdrojový kód implementace vytvořených metod v jazyku Python je dostupný v repozitáři na platformě GitHub pod odkazem `https://github.com/pavbal/ASoSI`.

2 Data

2.1 Datasety

Existuje několik datasetů pro sémantickou segmentaci satelitních snímků. Každý z těchto datasetů zpravidla obsahuje několik dvojic, které jsou tvořeny satelitním snímkem a jeho maskou (údaje o tom, k jaké třídě jednotlivé pixely náleží). Většinou je snímek reprezentován barevným RGB obrázkem a maska šedotónovým obrázkem s hodnotami odpovídajícími indexům tříd.

Tyto datasety zpravidla vznikají pro experimenty s hlubokým učením neuronových sítí a tudíž obvykle obsahují mnoho dvojic snímek-maska (počet záleží na rozlišení snímků, tedy na počtu jeho pixelů). Pro tyto úlohy je zpravidla nutné mít k dispozici hodně těchto dvojic. Tyto masky jsou na vytváření velmi časově nákladné (narozdíl od úloh klasifikace, kde k obrázku stačí přiřadit omezené množství popisků předem definovaných kategorií) a proto je tvorba segmentačních datasetů relativně náročná a nákladná činnost.

V následující podkapitole rozeberu dostupné datasety pro sémantickou segmentaci satelitních snímků, z nichž si následně jeden vyberu pro účely implementace metod segmentace.

2.1.1 Datasety pro sémantickou segmentaci satelitních snímků

LandCoverNet

LandCoverNet [7] je benchmark dataset postavený na satelitních snímcích z družic Landsat-8, Sentinel-1 a Sentinel-2. Jeden pixel snímků připadá přibližně na 10 až 15 délkových metrů zemského povrchu. Dataset je rozdělen dle kontinentů (Afrika, Jižní Amerika, Severní Amerika, Asie, Austrálie, Evropa). Díky rozlišení snímků zachycuje velký povrch v malém rozlišení.

Dataset dělí povrch na 7 tříd, které navíc dělí do dvou kategorií - *holé (voda, umělý, přírodní, permanentně sněžný/ledový)* a *porostlé (lesnatý, nelesnatý)*. LandCoverNet Europe tvoří cca 9,5 % celého datasetu, který je v plné velikosti velmi obsáhlý. V kombinaci s jeho poměrně hrubým rozlišením tedy vyplývá jeho velké pokrytí (velká nasnímaná plocha).

GID

GID (Gaofen Image Dataset) [8] je dataset skýtající 150 snímků povrchu Číny rozdělených do 30 000 trénovacích patchů. Snímky byly pořízeny satelitem Gaofen-2 a jeden pixel odpovídá cca 4 délkovým metrům zemského povrchu. Dataset umožňuje volbu mezi 5 a 15 segmentačními třídami.

PASTIS

PASTIS (Panoptic Agricultural Satellite TIme Series) [9] je od předchozích datasetů odlišný. Tento benchmark dataset se zabývá panoptickou a sémantickou segmentací zemědělských plodin, které rozděluje do 18 tříd. Dataset je postaven i pro úlohu panoptické segmentace a tudíž má vedena jednotlivá pole jako objekty (těch je v datasetu více než 120 tisíc). Dataset se skládá z 2433 sérií snímků představujících pozorování stejné oblasti napříč roční dobou. Každá z těchto sérií se skládá z 38 až 61 snímků pořízených satelitem Sentinel-2 (pixel odpovídá 10 m) v různých fázích růstu plodin a s rozlišením 128×128 pixelů.

INRIA Aerial Image Labeling

Dataset INRIA Aerial Image Labeling [10] je výjimkou ze všech zde uvedených datasetů v tom, že je složen pouze ze dvou tříd - *budova* a *ne-budova*. Jedná se tedy o dataset pro binární sémantickou segmentaci. Dataset je složen z 360 snímků s rozlišením 5000×5000 pixelů, přičemž jeden pixel odpovídá přibližně 30 cm zemského povrchu. Trénovací i testovací množina dohromady pokrývají 810 km^2.

LoveDA

LoveDA (Remote Sensing Land-Cover Dataset for Domain Adaptive Semantic Segmentation) [11] je benchmark dataset tvořen téměř 6000 snímky s rozlišením 1024×1024 pixelů rozdělených mezi 7 sémantických tříd, kdy jeden pixel připadá na 0,3 metru zemského povrchu. Dataset je určen, kromě úlohy sémanitické segmentace, také na úlohu adaptace domény bez učitele (unsupervised domain adaptation), která spočívá v aplikaci algoritmu na jiné prostředí, než na kterém byl natrénován, a zkoumání a optimalizaci jeho přizpůsobení.

Dataset LoveDA jsem si pro jeho neobvyklou koncepci vybral pro účely této práce. Jedním z důvodů byla jeho vyhovující velikost. Dále mě zaujala náročnost, která je z podstaty datasetu kladena na segmentační algoritmus, kdy oblasti na snímkách v trénovací, validační a testovací množině pocházejí z různých geografických oblastí Číny.

Charakteristikám datasetu LoveDA se věnuji níže v kapitole 2.2.

Ostatní příbuzné datasety

Existuje a je k dispozici mnoho dalších datasetů podobných těm uvedeným výše. Patří mezi ně např. SpaceNet 2 [12], RWanda Built-up Region Segmentation [13], MiniFrance [14], Five-Billion-Pixels [15] a LandCover.ai [16].

2.2 Dataset LoveDA

Dataset LoveDA (Land-cover Domain Adaptive semantic segmentation) [17][11] je víceúčelový dataset pro trénování metod počítačového vidění v oblasti sémantické segmentace snímků zemského povrchu za účelem dálkového průzkumu Země.

2.2.1 Charakteristiky datasetu

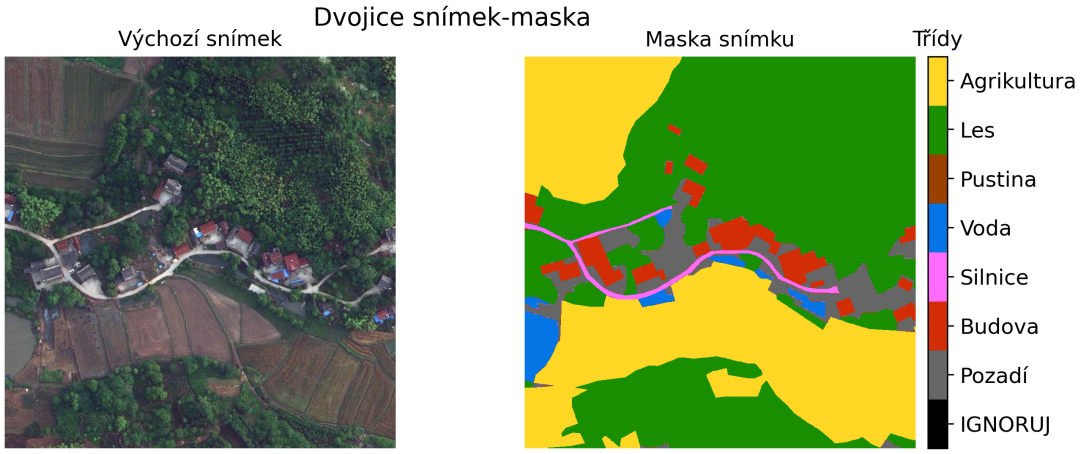
Dataset využívá technologii HSR (high spatial resolution) snímků. Jeden pixel odpovídá přibližně 0,3 metru zemského povrchu. Celkově snímky pokrývají povrch

o rozloze 536,15 km^2.

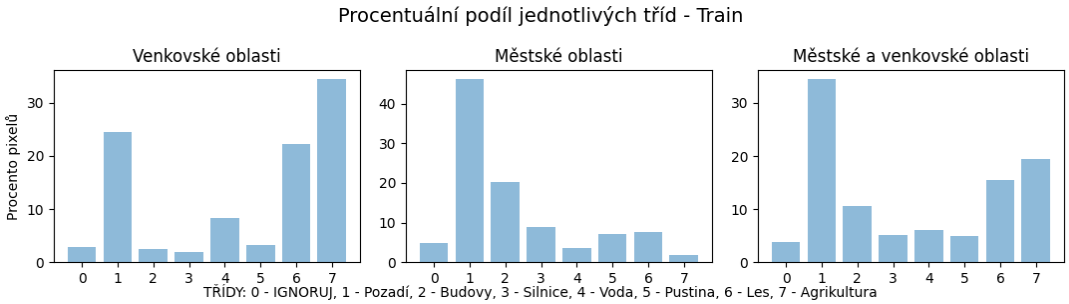
Dataset sestává z 5987 HSR RGB snímků a jim odpovídajících šedotónových masek. Šedotónové masky k testovací množině datasetu nejsou k dispozici přímo ke stažení kvůli zachování objektivity a výpovědní hodnotě prezentovaných výsledků v soutěži s datasetem spojené. Snímky i masky jsou čtvercové s rozlišením 1024×1024 pixelů. Masky různými hodnotami pixelů rozdělují prostor snímků (obrázků) do 8 tříd, přičemž 7 tříd je sémanticky a informačně věcných; jedná se o třídy *pozadí* (background), *budova* (building), *silnice* (road), *voda* (water), *pustina* (barren), *les* (forest) a *agrikultura* (agricultural). Pomyslná třída *IGNORUJ* (IGNORE) je v datsetu obsažena z důvodu toho, že nějaké snímky vzniklé rozdělením snímků větších nezaplní celou plochu 1024×1024 pixelů.

Dvojice snímek-maska jsou v rámci dostupného datasetu rozděleny do dvou množin - trénovací (Train) (2522 dvojic) a validační (Val) (1669 dvojic). Testovací množina (Test) potom obsahuje 1796 snímků, přičemž jejich masky nejsou k dispozici veřejně (vysvětleno výše). Kromě tohoto standardního rozdělení je však snímek (případně i snímek s maskou) v rámci každé této množiny klasifikován do jedné ze dvou tříd Rural (venkovská oblast) a Urban (městská oblast). Toto dělení je napříč konkurenčními datasety unikátní. Výhodou této dodatečné klasifikace je možnost použít krom metod ryzí sémantické segmentace i metody tzv. UDA (Unsupervised Domain Adaptation). Výskyt snímků z venkovského i městského prostředí způsobuje velkou proměnlivost v distribuci a tvaru objektů jednotlivých tříd. Vizualizaci jedné dvojice snímek-maska můžete vidět na obrázku 2.1.

Snímky datasetu LoveDA zachycují 18 geografických oblastí ze tří územních celků čínských měst Nanjing, Changzhou a Wuhan. Struktury v rámci jedné třídy se tudíž nacházejí v různých geografických oblastech a v rozdílných krajinách, což má za následek velkou vnitrotřídní rozmanitost datasetu. Každá z 18 oblastí je přitom zastoupena právě v jedné ze tří základních množin (Train, Test a Val), tudíž se segmentační algoritmus v testovací a validační množině potká nejen s danými lokalitami, ale i krajinnými rázy a oblastními specifiky, poprvé, což klade



Obrázek 2.1: Vykreslení dvojice snímek-maska v rámci trénovací množiny datasetu LoveDA, upraveno pro přehlednou vizualizaci



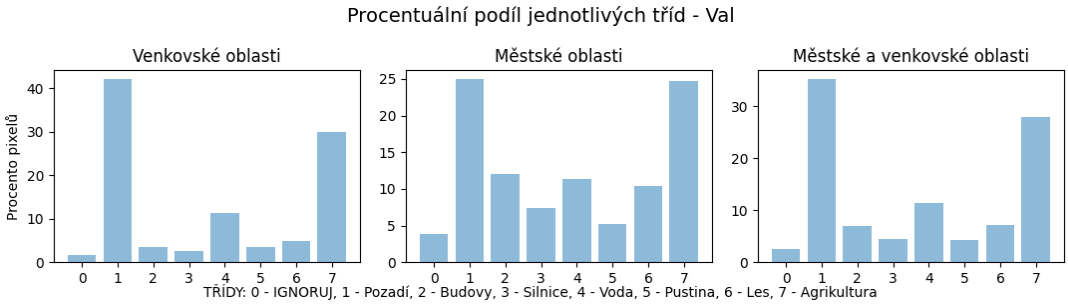
Obrázek 2.2: Procentuální rozložení tříd v maskách trénovací množiny datasetu LoveDA

na generalizační schopnosti algoritmu vysoké nároky a tím zvyšuje náročnost datasetu.

Ze statistiky trénovací množiny na obrázku 2.2 vyčteme, že výrazný podíl snímků zabírá třída pozadí. Na snímcích obou oblastí zabírá tato rozmanitá třída téměř 35 % celkové plochy. Větší celkové zastoupení mají i třídy budova, les a agrikultura. Ostatní třídy se poté pohybují na hranici kolem 5 %.

Velké procentuální rozdíly mezi venkovskými a městskými oblastmi pozorujeme u dvojice tříd *les* a *agrikultura*, které mají přirozeně výrazně vyšší zastoupení ve venkovských oblastech než v těch městských. Opačný jev pak pozorujeme u třídy *budova*, která má výrazně vyšší zastoupení v oblastech městských.

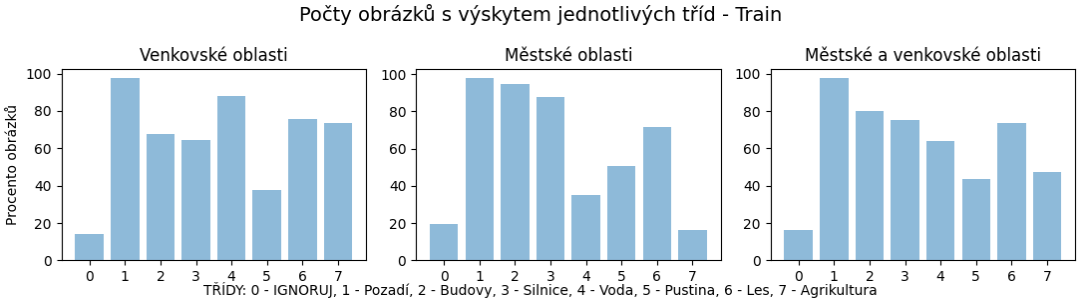
Procentuální statistiky z validační množiny (viz obrázek 2.3) se oproti sta-



Obrázek 2.3: Procentuální rozložení tříd v maskách validační množiny datasetu LoveDA

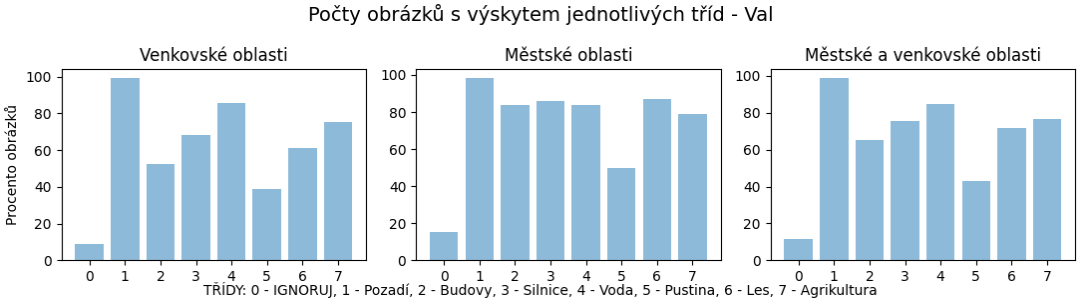
tistikám trénovací množiny v některých aspektech relativně liší. To je pravděpodobně způsobeno fyzicko-geografickými a kulturními odlištnostmi výše zmíněných lokalit napříč Čínou. Největší rozdíl je v zastoupení lesů, kdy je jejich podíl ve venkovských i městských oblastech výrazně nižší než v případě oblastí z trénovací množiny. Celkový poměr však zůstává podobný (méně lesů a více agrikultury).

Dalším velkým rozdílem je velké procento třídy *agrikultura* v čistě městských oblastech. Podíl třídy *pozadí* u městských oblastí je menší, ale podíl u oblastí venkovských je výrazně vyšší. Obě tyto změny jsou přibližně stejně velké.



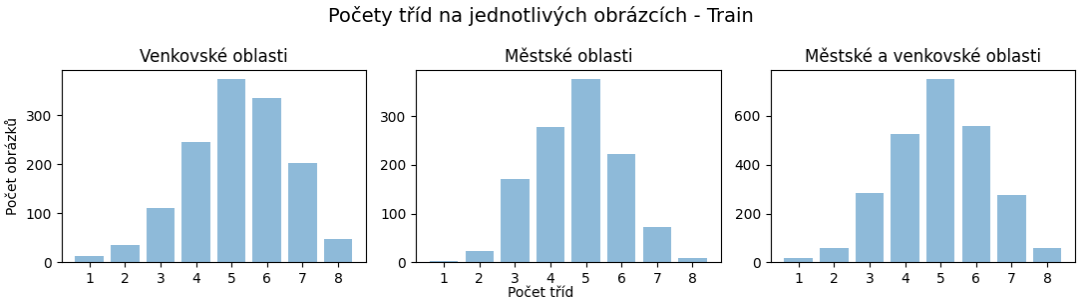
Obrázek 2.4: Sloupcový graf vyjadřující četnost zastoupení jednotlivých tříd na maskách snímků v trénovací množině

Na obrázcích 2.4 a 2.5 vidíme několik rozdílů mezi množinami co se týče výskytu tříd na jednotlivých obrázcích v testovací a validační množině. V trénovací množině se na menší části snímků vyskytuje třída *agrikultura*. To je způsobeno malým výskytem této třídy v městských oblastech. Dále můžeme z obrázků vypozorovat, že třída *pustina* se vyskytuje na poměrně méně snímcích než ostatní



Obrázek 2.5: Sloupcový graf vyjadřující četnost zastoupení jednotlivých tříd na maskách snímků ve validační množině datasetu LoveDA

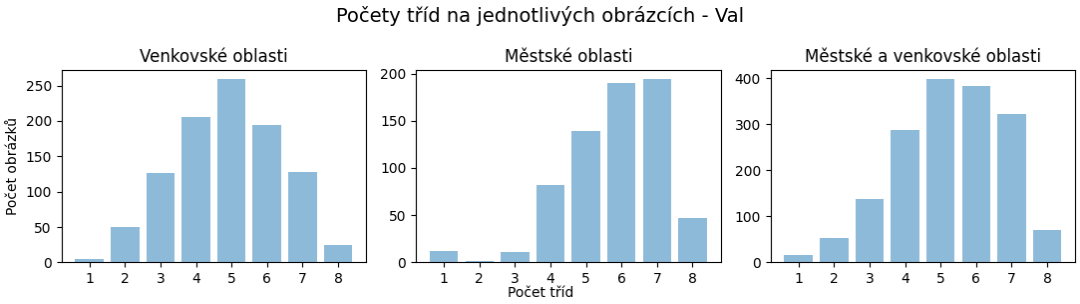
třídy u obou množin.



Obrázek 2.6: Sloupcový graf vyjadřující třídní rozmanitost na základě seskupení počtu masek s určitým počtem tříd v trénovací množině datasetu LoveDA

Obrázky 2.6 a 2.7 znázorňují, na kolika obrázcích se vyskytuje určitý počet tříd. Po vydělení počtem snímků v jednotlivých množinách by se nám z těchto statistik stala diskrétní pravděpodobnostní rozdělení popisující počet tříd na snímku (masce) dvou množin datasetu i datasetu jako celku.

Ze statistik 2.2, 2.3, 2.4, 2.5 vidíme, že se množiny relativně liší v procentuálním zastoupení i výskytu určitých tříd na snímcích. Snímky validační množiny jsou též poměrně bohatší o obsažené třídy. Tyto skutečnosti kladou vyšší nároky na segmentační algoritmus, který je nucen popisovat snímky v kontextu, se kterým se během trénování nesetkal.



Obrázek 2.7: Sloupcový graf vyjadřující třídní rozmanitost na základě seskupení počtu masek s určitým počtem tříd ve validační množině datasetu LoveDA

2.2.2 Soutěž LoveDA Semantic Segmentation Challenge

Soutěž LoveDA Semantic Segmentation je projekt, který si klade za cíl podporovat výzkum dálkového mapování a krajinný průzkum na HSR snímcích [18][11]. Soutěž je k dispozici na webové open-source platformě CodaLab [19], která umožňuje vytvářet a účastnit se soutěží týkajících se práce s daty a porovnávat výsledky s ostatními účastníky.

Jak jsem vysvětlil výše (2.2.1), masky testovací množiny datasetu nejsou k dispozici ke stažení přímo. V případě, že chceme vyhodnotit náš segmentační algoritmus a porovnat jej s ostatními účastníky soutěže, je možné nahrát archiv s predikovanými maskami pro snímky testovací množiny Test do webového prostředí CodaLab, kde následně proběhne pro každé predikované masky jejich konzistentní vyhodnocení. Soutěže se v v době psaní této práce zúčastnilo přibližně 190 týmů a své výsledky z nich zveřejnilo 93 z nich.

2.2.3 Vyhodnocování algoritmu

Pro vyhodnocení segmentačního algoritmu je obecně možné využít řadu metrik a kritérií. Nejintuitivnějším kritériem posouzení kvality predikce je přesnost (accuracy), ta se vypočítá následovně:

$$\text{accuracy} = \frac{\sum_{i=1}^{N} \text{correct}_i}{\sum_{i=1}^{N} \text{total}_i} \tag{2.1}$$

kde $correct_i$ je počet správně klasifikovaných pixelů do i-té třídy, $total_i$ je celkový počet pixelů i-té třídy a N je celkový počet tříd.

Přesnost predikce nám v případě segmentační úlohy jednoduše říká, jaký podíl pixelů byl klasifikován správně. Tento přístup však může být za určitých podmínek pro segmentační úlohu nevhodný.

Jedním z případů, kdy není použití procentuální přesnosti ideální, je případ, kdy je zastoupení tříd v datasetu výrazně nerovnoměrné (tj. když se četnost pixelů mezi jednotlivými třídami v datasetu výrazně liší). To je také případem datasetu LoveDA. Jak vyplývá ze statistik procentuálního podílu jednotlivých tříd v trénovací a validační množině (obrázky 2.2 a 2.3), které nám informaci o zastoupení tříd přímo předkládají, jsou v datasetu vcelku velké rozdíly v jejich zastoupení. Abychom mohli brát procentuální přesnost jako směrodatnou, muselo by se zastoupení každé třídy (s předpokladem o nezapočítání irelevantní třídy IGNORUJ) pohybovat kolem hodnoty 14,3 %.

Protože je v datasetu zastoupení tříd značně nerovnoměrné, zvýhodňovalo by procentuální kritérium třídy s velkým zastoupením. Kdyby např. algoritmus predikoval všechny relevantní pixely jako třídu *pozadí*, bylo by pravděpodobné, že by dosáhl více jak 30% procentuální přesnosti, což by se jako samostatné číslo nemuselo zdát tak špatné. Tento výsledek ale v praxi není použitelný a jediná jeho hodnota spočívá v informaci o třídě s velkou předpokládanou apriorní pravděpodobností.

Pro validační účely našich algoritmů je mnohem vhodnějším kritériem tzv. intersection-over-union (průnik dělený sjednocením, dále IoU). Pro použití v úloze sémantické segmentace nám kritérium IoU udává pro každou predikovanou třídu podíl průniku četnosti pixelů této třídy v masce s její predikcí a jejího sjednocení. Výpočet metriky tedy lze popsat množinovým vztahem:

$$IoU_n = \frac{M_n \cap P_n}{M_n \cup P_n} \tag{2.2}$$

kde n je index třídy, M_n je množina pixelů odpovídajících třídě n a P_n jsou

množiny pixelů klasifikovaných algoritmem do třídy n.

Hodnota kritéria pro každou třídu se tedy pohybuje mezi hodnotami 0 (nulová shoda, tj. nulový průnik masky a její predikce) a 1 (shoda úplná). Tato hodnota nenese informaci o zastoupení třídy a tudíž ji nemůže zvýhodnit, ani znevýhodnit. Celkově (ať už v rámci jednoho snímku, nebo celého datasetu) se kritérium prezentuje jedním číslem jako aritmetický průměr IoU pro každou třídu (tzv. mean Intersection-over-Union, dále mIoU). Kritérium mIoU je tedy vypočítáno následovně:

$$mIoU = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FP_i + FN_i} = \frac{1}{N}\sum_{i=1}^{N} IoU_i \tag{2.3}$$

kde v našem případě N je počet tříd, TP je počet správně klasifikovaných pixelů do i-té třídy, FP je počet chybně klasifikovaných pixelů do i-té třídy a FN je počet pixelů náležících i-té třídě, ale klasifikovaných do třídy jiné.

Kritérium mIoU je pro své vhodné vlastnosti používáno jako hlavní metrika úspěšnosti v soutěži LoveDA Semantic Segmentation i v oficiálních dokumentech datasetu [11][18].

Další metrikou použitelnou pro účely segmentace (v základní podobě pro dichotomickou) a vhodnou pro rozdělení tříd našeho datasetu je F1 score. Ta vyjadřuje vyváženost mezi precizností (precision) a úplností (recall):

$$\text{precision} = \frac{TP}{TP + FP} \tag{2.4}$$

$$\text{recall} = \frac{TP}{TP + FN} \tag{2.5}$$

kde (stejně jako u mIoU výše) je TP počet správně klasifikovaných pixelů do konkrétní třídy, FP počet chybně klasifikovaných pixelů do konkrétní třídy a FN počet pixelů náležících konkrétní třídě, ale klasifikovaných špatně.

Precizností tedy v našem případě chápeme poměr počtu pixelů správně klasifikovaných jedné třídě a počtu všech pixelů klasifikovaných do této třídy. Úplnost pak chápeme jako senzitivitu binární klasifikace příslušnosti k jedné třídě (v našem případě tedy jako podíl počtu pixelů, které jsou správně klasifikovány příslušné třídě a celkového počtu pixelů, které k této třídě mají náležet). Tyto dva poměry jsou poté použity pro výpočet F1 score podle následujícího vztahu:

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{2.6}$$

Tato metrika je v naší trojici jakýmsi protipólem k procentuální přesnosti, neboť špatná klasifikace vzácnějších tříd má často za následek velký úbytek na celkovém skóre.

3 Segmentační metody

Důležitým krokem ve většině úloh strojového učení je extrakce příznaků.

Jedním přístupem je příznaky extrahovat ručně za pomoci znalosti dané problematiky (handcrafted-feature extraction). Toho může být dosaženo aplikací rozlišných statistických metod, transformací dat, filtrací, nebo jiných příznaků přímo uzpůsobených charakteru úlohy.

Druhým přístupem je extrakce příznaků pomocí strojového učení. V tomto případě využíváme algoritmus, jehož cílem je naučit se extrahovat relevantní příznaky z dat. Tento přístup je používán v end-to-end přístupech (v hlubokém učení).

V případě ruční extrakce příznaků v segmentační úloze je potřeba využít k samotné klasifikaci vhodný klasifikátor, který bude na základě těchto příznaků přiřazovat obrazům jejich třídy. Jedním z nejpoužívanějších klasifikátorů je SVM (support vector machine) [20]. Tento klasifikátor rozděluje iterativním procesem prostor příznaků pomocí nadrovin, jimž upravuje parametry tak, aby dosáhl správné klasifikace (resp. nejnižší hodnoty ztrátové funkce). Dalším používaným klasifikátorem je GNB (Gaussian Naive Bayes, varinta Bayesova klasifikátoru [21]). Ten klasifikuje obrazy na základě nejvyšší pravděpodobnosti a za předpokladu jejich normálního rozdělení.

3.1 Vybrané přístupy k segmentaci

3.1.1 LBP

Metoda LBP (local binary pattern) [22] je klasickou metodou umělé inteligence, která nevyužívá statistické zpracování, ani filtrační přístup. Jedná se o metodu ruční extrakce příznaků, která operuje v řádu pixelů a jejich okolí. Tradičně byla používána pro úlohy klasifikace, detekce objektů a rozpoznávání obličejů.

Metoda zpracovává šedotónový obrázek pixel po pixelu a hodnotí jejich okolí. Nejprve zvolený počet okolních pixelů vyprahuje podle centrálního (0, když je jas pixelu menší než centrální a 1 jinak), následně seřadí získané binární hodnoty do víceciferného binárního čísla (počet jeho cifer bude roven počtu uvažovaných okolních pixelů), to následně bez změny pořadí seskupí tak, aby bylo co možná nejnižší.

Takto získané číslo (po převedení do desítkové soustavy a případném vynormování), které je definované pro každý pixel, nám slouží jako hodnota pro daný pixel v novém šedotónovém obrázku, který nám vznikne iterativní aplikací metody na všechny pixely v původním obrázku. Nově získaný šedotónový obrázek (ilustrovaný na obrázku 3.1) je pak přímým výstupem většiny implementací metody LBP.

Obrázek 3.1: Příklad aplikace metody (transformace) LBP na obrázek z knihovny scikit-image.

3.1.2 HOG

Metoda HOG (Histograms of oriented gradients) [23] je metodou klasické umělé inteligence s ruční extrakcí příznaků používanou především pro detekci obrazu. Je založena na myšlence, že se tvar a vzhled zobrazených objektů dá poměrně věrně charakterizovat lokálními gradienty.

Metoda rozčlení šedotónový obraz na čtvercové buňky o fixním počtu pixelů. Hlavním parametrem metody je krom velikosti buňky také počet směrů udávající jemnost a zároveň datový objem výsledné reprezentace. Metoda vypočte v rámci jedné buňky gradient pro všechny tyto směry (rovnoměrně rozložené v rovině). Každé této buňce přiřadí metoda histogram zachycující právě hodnoty těchto gradientů v zadaném počtu směrů. Příklad aplikace HOG transformace na obrázek je vizualizován na obrázku 3.2, kde je použita velikost buňky 16×16 pixelů.

Metodou vlastně vytváříme reprezentaci každé buňky vektorem (hodnot gradientů v různých směrech). To nejčastěji chápeme jako extrakci vektoru užitečných příznaků uchopitelné velikosti, na základě které můžeme provádět klasifikaci.

Obrázek 3.2: Příklad aplikace metody HOG na obrázek z knihovny scikit-image.

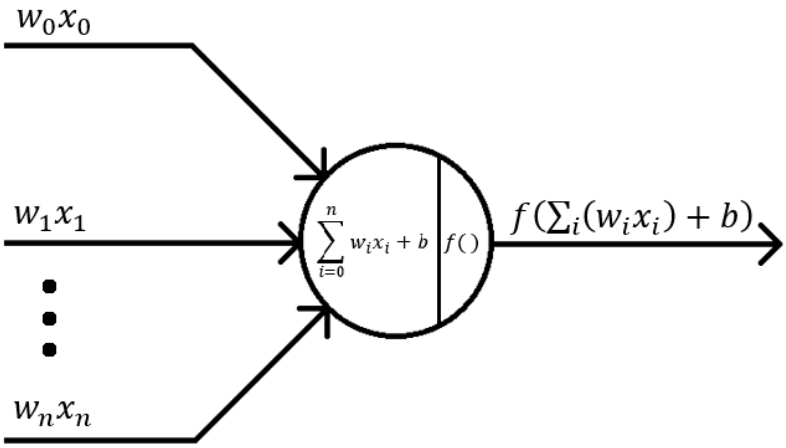
3.2 Umělá neuronová síť

Umělé neuronové sítě jsou široce používaným nástrojem pro řešení mnoha problémů nejen v počítačovém vidění. Řadí se do metod extrakce příznaků pomocí strojového učení a uplatňují se moderních přístupech v oborech, jako je překlad jazyka, porozumění textu, rozpoznávání řeči a mnoho dalších. Tyto sítě jsou inspirovány biologickými sítěmi neuronů, které se nacházejí v lidském mozku. Podobně jako lidský mozek, neuronové sítě jsou složeny z mnoha propojených jednotek - neuronů - které zpracovávají informace a přenáší je mezi sebou pomocí signálů.

Umělé neuronové sítě se skládají z neuronů uspořádaných do vrstev, kde jsou aktivovány v diskrétním čase. Aktivační frekvence (četnost aktivací za jednotku času) hrající klíčovou roli v biologických neuronech je zde tedy nahrazena výstupní hodnotou aktivační funkce počítanou v diskrétních okamžicích v po sobě jdoucích vrstvách neuronů.

Každý neuron v umělé neuronové síti (viz obrázek 3.3) je výpočetní jednotkou, která ze vstupů (tj. z výstupů neuronů předešlé vrstvy), svých vah, prahu a hodnoty aktivační funkcí vypočte výstup, který předá neuronům v další vrstvě.

Základním prvkem klasických neuronových sítí je plně propojená (fully-connected) vrstva neuronů. Výstup každého neuronu v této vrstvě jde na vstup všech neuronů ve vrstvě následující. Stejně tak tyto neurony na vstupech obdrží obecně výstupy všech neuronů ve vrstvě předchozí. Každá z těchto hran je parametrizována svou vahou, což vede k prudkému nárůstu parametrů s rozměry vrstvy a s počtem vrstev. To zvyšuje nejen riziko přetrénování sítě, ale také způsobuje významný nárůst výpočetní náročnosti s rozměry a s jejich počtem. Z těchto důvodů se v posledních letech budují architektury, které umožňují snížit celkový počet parametrů sítě za zachování (a dokonce zlepšení) jejich výsledků. Typickým a nejpoužívanějším typem architektury navržené především pro zpracování obrázků je konvoluční neuronová síť.



Obrázek 3.3: Schéma jednotky neuronové sítě - neuronu. proměnné x_i představují výstupy aktivačních funkcí neuronů z předešlé vrstvy a w_i jsou optimalizované váhy jednotlivých těchto hran, pro každý vstup neuronu je tedy vyhrazen jeden parametr w_i. Parametr b poté představuje prahovou hodnotu, tedy optimalizovanou nelineární složku, neuronu. Funkce $f()$ je aktivační funkcí neuronu.

Algoritmus zpětné propagace

Aby byla umělá neuronová síť užitečná, je třeba důmyslným způsobem měnit její parametry (váhy a prahy neuronů) na základě jejích výsledků. Toho je dosahováno algoritmem zpětné propagace (backpropagation), který umožňuje vypčtením gradientu zlepšit odhad parametrů. Činí tak na základě požadovaného výsledku (v našem případě jednotlivých masek). Algoritmus zpětné propagace tedy umožňuje samotný proces učení neuronové sítě při předkládání trénovacích dvojic (snímek - maska).

Samotný algoritmus tedy představuje výpočet gradientu. Za pomoci aktuálních vah a prahů vypočítá hodnoty výstupů neuronů. Z těchto výstupů a z požadovaných vstupů následně vypočítá ztrátovou funkci (viz dále 3.2.1). Gradient ztrátové funkce se poté vypočítá zpětně pomocí řetězového pravidla (chain-rule) a propaguje se do předcházejících vrstev sítě. Tento proces je opakován, dokud se nedosáhne požadovaného stupně naučení sítě.

Konvoluční neuronová síť

Konvoluční neuronové sítě (CNN) vznikly jako analogie k napodobení kognitivní zrakové funkce živočichů, která dokáže vnímat prvky reálného světa jako objekty.

Podstatou a myšlenkou konvolučních neuronových sítí je konvoluční vrstva. Tato vrstva, narozdíl od klasické plně propojené vrstvy, má na vstupech přístup pouze k určité lokální podoblasti neuronů vrstvy předchozí. Konvoluční vrstva disponuje hloubkou, ve které neurony pod sebou vidí na stejnou oblast vrstvy předchozí, ale nikoliv na sebe navzájem. A protože každá úroveň neuronů v hloubce sdílí jedny optimalizované váhy a jednu prahovou hodnotu, tak každý neuron v této jedné úrovni hloubky provádí stejnou operaci nad určitou oblastí předchozí vrstvy [24]. Klasickým příkladem aplikace je použití kompaktního lokálního okolí (okna) v neuronech (resp. pixelech) předchozí vrstvy (např. Moorovo okolí).

Hloubkovou úrovní sdílené parametry tedy můžeme brát jako konstantní matici vah. Výstup této hloubkové úrovně potom představuje konvoluci této matice s vstupní vrstvou (typicky předzpracovaným obrázkem). Konvoluční vrstvu tedy můžeme brát jako soubor paralelně uspořádaných lokálních filtrů.

Další důležitou vrstvou CNN je agregační (pooling) vrstva. Jedná se o vrstvu, která slouží ke snížení rozměru dat z vrstvy předchozí a zvýšení robustnosti vůči drobným posunům a změnám v datech. Nejčastěji je používán tzv. max-pooling, čímž se z každé oblasti dat vybere nejvyšší hodnota a tato hodnota se použije jako reprezentace této oblasti v následující vrstvě sítě. Tento proces tedy shrnuje informace z většího množství neuronů předchozí vrstvy do jediné hodnoty. Nejčastější použití pooling vrstvy je použití maxima z oblasti 2×2 pro reprezentaci v další vrstvě, čímž dochází k redukci parametrů následující vrstvy na čtvrtinu.

CNN pro segmentaci

V oblasti sémantické segmentace je populární použití tzv. enkodér-dekodér sítí, které jsou navrženy tak, aby mohly efektivně zpracovat informace v obrázku v různých měřítcích. Výstupem těchto sítí je pravděpodobnostní mapa pro každou

třídu v každém pixelu. Obecná architektura typu enkodér-dekodér je využívána v širším spektru úloh, jako je, kromě sémantické segmentace, např. úloha generování obrázků, detekce anomálií, nebo překlad psaného jazyka.

Mezi populární konvoluční architektury a modely patří např. U-Net [25], PSP-Net (Pyramid Scene Parsing Network) [26] a DeepLab [27].

DeepLabV3

DeepLabV3 [28] (a DeepLabV3+) je jedním z nejpoužívanějších přístupů k sémantické segmentaci pomocí umělých neuronových sítí. Jedná se o hlubokou konvoluční neuronovou síť typu enkodér-dekodér. Nejvýraznější charakteristikou její architektury je použití tzv. Atrous konvoluce [29]. Tento druh konvoluce spočívá v rozšíření konvolučního okna (jádra) určitých vrstev sítě bez nutnosti zvyšování počtu parametrů konvoluční vrstvy. Toho je dosaženo vložením mezer (dilatací) mezi buňky okna (typicky např. moorova okolí). Tím jsou vstupní data neuronu konvoluční vrstvy rozprostřeny dál od sebe. Volením různé dilatace a rozměru původního okna je tedy v jednotlivých úrovních konvoluční vrstvy dosahováno efektu různého rozšíření zorného pole sítě (specificky filtrů v konvolučních vrstvách) a to (jak jsem psal výše) bez nutnosti zvyšování počtu parametrů. Důsledkem správného použití Atrous konvoluce je schopnost sítě vnímat širší kontext obrazu a segmetnovat i velmi jemné struktury. Příkladem efektivního využití Atrous konvolučních vrstev je tzv. ASPP (Atrous Spatial Pyramid Pooling) blok (viz příloha A.1). ASPP blok je soubor několika sériových Atrous vrstev s různou velikostí dilatace. Tento modul umožňuje zachycení obrazu ve více prostorových měřítkách a tím zlepšení segmentačních vlastností.

DeepLabv3 a jeho vylepšená verze DeepLabV3+ jsou díky svým vlastnostem využívány jako základ architektur neuronových sítí v mnoha aplikacích, jako je například rozpoznávání obličejů [30], segmentace lékařských snímků [31], detekce překážek v autonomních vozidlech, nebo rozpoznávání objektů a textur na satelitních snímcích. Právě jeho vhodnost k využití pro práci se satelitními snímky

je hlavní motivací k využití modelu vycházejícího z architektury DeepLabV3 pro účely této práce.

DeepLabV3 s implementací architektury ResNet50

ResNet50 [32] je architektura neuronové sítě o 50 vrstvách (48 konvolučních vrstev). Jejím základním specifikem jsou reziduální bloky (viz obrázek v příloze A.2). Tyto bloky umožňují větší hloubku sítě, aniž by se zhoršila její schopnost učení. V každém reziduálním bloku je vstupní signál převeden na výstup přes několik konvolučních vrstev a dále přidán k původnímu vstupu, aby se výsledný výstup lépe přizpůsobil optimalizaci neuronové sítě.

DeepLabV3 ResNet50 je varianta konvoluční sítě DeepLabV3 využívající jako základní architekturu síť ResNet50. Ta je doplněna o Atrous konvoluční vrstvy a ASPP bloky a zachovává si princip enkodér-dekodér. Její struktura je zobrazena v příloze A.3.

3.2.1 Důležité komponenty neuronových sítí a trénovacího procesu

Pro efektivní implementaci neuronové sítě je nutné porozumět fungování nejen jejímu, ale také porozumět procesu trénovací smyčky a sloučit její komponenty do funkčního celku. Různé kombinace následujících komponent a jejich hyperparametrů se budou různě rychle trénovat a jejich výsledek bude dosahovat různých kvalit. Pro vytvoření co možná nejúčinějšího segmentačního algoritmu je tedy žádoucí porozumět těmto komponentám a vztahům mezi nimi.

Trénovací a validační proces

Trénovací proces je proces, kdy po předkládání dat a jejich propagaci neuronovou sítí, jsou na základě vypočtené ztrátové funkce aktualizovány parametry (váhy a prahy) jednotlivých neuronů. Klíčovou úlohu zde plní algoritmus zpětné propagace (viz 3.2). Trénovací proces zpravidla probíhá po tzv. epochách (tj. úsecích,

kdy je neuronové síti předložena celá trénovací množina). Často se poté přistupuje k dávkování vstupních dat, kdy jsou parametry sítě aktualizovány až po průchodu několika vstupních dat (obrázků). Velikost dávky je jedním z hyperparametrů neuronové sítě.

Samotný proces trénování je tvořen cyklem, v rámci něhož dojde vždy k předložení jednoho normalizovaného obrázku (data), nebo jejich dávky, na vstup neuronové sítě, odkud dojde k jeho dopředné propagaci vrstvami sítě až na její výstup, který představuje predikci sítě. Na základě odlišností predikce a očekávaného výstupu (masky) je poté vypočítána ztráta pomocí kritéria ztrátové funkce (viz 3.2.1). Ztráta je pak následně ve formě gradientů algoritmem zpětné propagace propagována od výstupu do celé sítě. Na základě těchto gradientů jsou poté pomocí optimizeru (viz 3.2.1) aktualizovány parametry sítě (váhy a prahy jednotlivých neuronů).

Validační proces je poté proces, kdy jsou neuronové síti na vstup předkládána data z validační množiny, na nichž dochází k vyhodnocení chybovosti neuronové sítě pomocí zvolených metrik (viz 2.2.3). Na základě tohoto procesu je poté určována kvalita neuronové sítě a úspěšnost trénovacího procesu.

Ztrátová funkce

Ztrátová funkce slouží jako metrika, která kvantifikuje míru odlišnosti požadovaného a predikovaného výstupu sítě. Ztrátových funkcí používaných v úlohách segmentace existuje několik. Nejvíce využívaná ztrátová funkce využívá křížové entropie (tzv. Cross Entropy Loss), ta je vypočítána následovně:

$$CELoss = -\frac{1}{K}\sum_{i=1}^{K}\sum_{n=1}^{N} y_{in}\log(p_{in}) \tag{3.1}$$

kde K je celkový počet pixelů, N je počet segmentačních tříd, y_{in} je binární indikátor rozhodující, zda patří pixel i do třídy n a p_{in} je sítí predikovaná pravděpodobnost (výstup sítě), že pixel i spadá do třídy n.

Funkce tedy vypočítává ztrátu pro každý vstupní vzorek jako negativní logaritmus pravděpodobnosti, kterou model přiřazuje správné třídě. Tím zvyšuje

ztrátu v případech, kdy se predikované pravděpodobnosti výrazně liší od skutečných hodnot. Z důvodu její vhodnosti využiji ztrátovou funkci na základě křížové entropie v samotné implementaci trénovacího cyklu.

Optimizer

Optimizer je algoritmus, který se používá k optimalizaci parametrů sítě během procesu učení. Ten funguje na základě algoritmu gradientního sestupu (gradient descent) s využitím derivací ztrátové funkce vzhledem k parametrům sítě (tj. s využitím jejich změny). Derivace ztrátové funkce poté udávají směr (z definice gradientu), kterým se mají parametry upravit tak, aby byla ztrátová funkce v dalším kroku co nejmenší.

Existuje několik druhů optimizerů. Nejzákladnějším z nich je SGD (Stochastic Gradient Descent) [33]. Pro každý vzorek na vstupu je vypočten gradient ztrátové funkce vzhledem k parametrům sítě, následně jsou tyto parametry aktualizovány tak, aby se posunuly v opačném směru gradientu a tím snížily hodnotu ztrátové funkce. Velikost změny aktualizace je poté vyjádřena hyperparametrem konstantou učení (learning rate).

Dalším používaným optimizerem je Adam [34]. Adam si narozdíl od SGD volí velikost své konstanty učení sám. Díky této adaptabilitě zvládá lépe různorodé gradienty a může pomoci dosáhnout rychlejší konvergence trénovacího cyklu.

Plánovač

Plánovač (scheduler) je nástroj, který je využíván k dynamické změně velikosti konstanty učení (learning rate) optimizeru při běhu trénovacího procesu, typicky po dosažení určitého počtu epoch, nebo při poklesu ztrátové funkce.

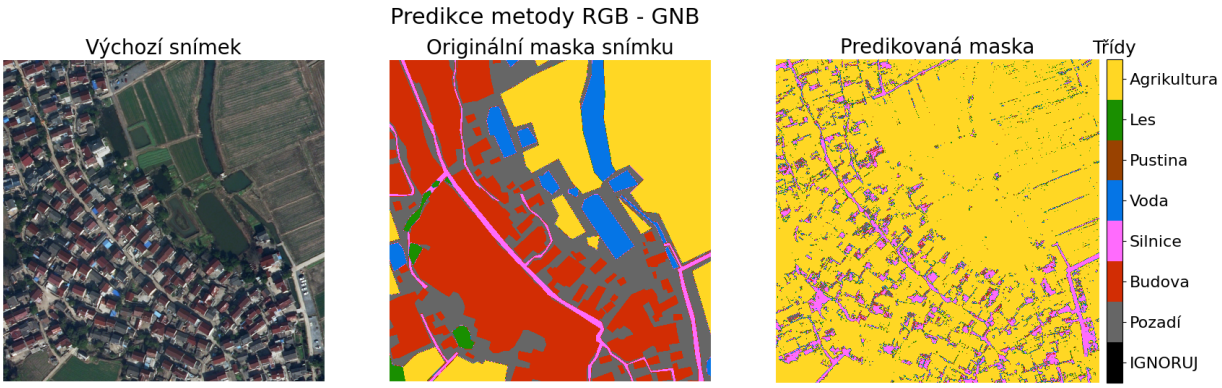
4 Experimenty

4.1 Klasické metody

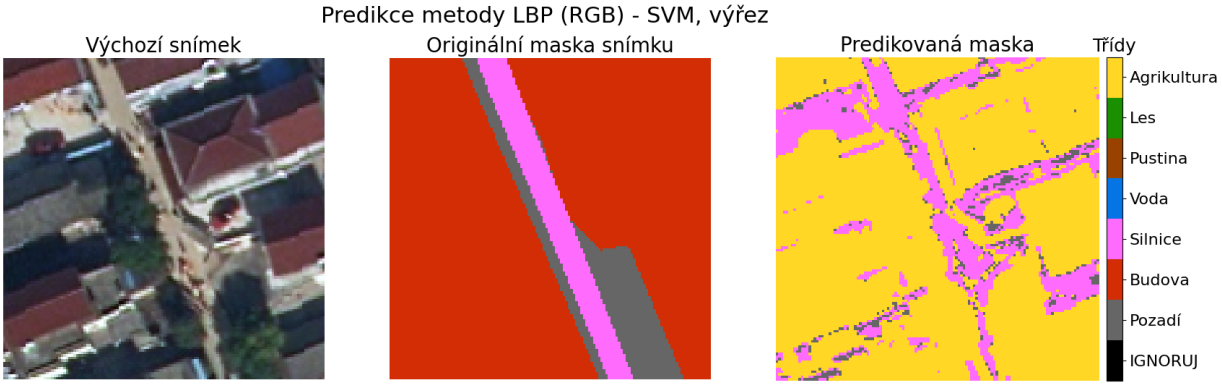
4.1.1 Klasifikace dle RGB složek

Abych mohl následující metody a jejich použitelnost s něčím porovnat, natrénoval jsem klasifikátor GNB čistě na vycentrovaných (odečtením hodnoty 128 od každého pixelu každé barevné vrstvy) RGB snímcích. Metoda v některých ohledech vykazovala lepší výsledky (viz tabulka 4.1) než metoda LBP. Jedná se přitom pouze o vícedimenzionální prahování jasových hodnot RGB složek jednotlivých pixelů v původním obrázku. Příklad predikce je vidět na obrázcích 4.1 a 4.2.

K nevýhodám této metody patří velikost její reprezentace, kdy je každý pixel prezentován třemi hodnotami a klasifikován sám za sebe bez kontextu okolních pixelů. Výsledky metody RGB natrénované na prvních 16 snímcích trénovací množiny (Train) jsou vidět v porovnání s metodou LBP v tabulce 4.1.



Obrázek 4.1: Výsledek použití GNB klasifikátoru natrénovaného na podmnožině 16 snímků trénovací množiny datasetu LoveDA na původní RGB snímek.



Obrázek 4.2: Detail zobrazeného výsledku z obrázku 4.1

4.1.2 LBP

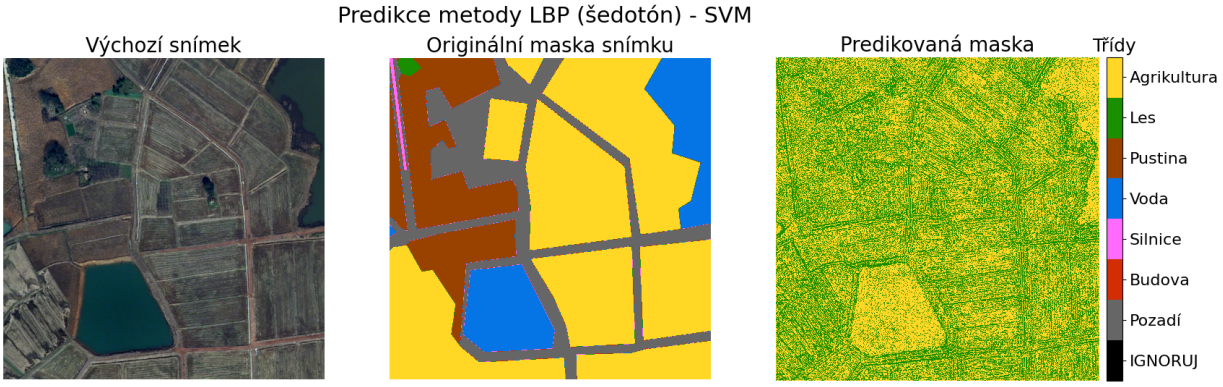
Použití LBP pro segmentaci je neobvyklé a proto důkladně popíši svůj postup.

Nejprve jsem aplikoval standardní LBP transformaci uvažující jednoduché Moorovo okolí (osmiokolí) na snímek převedený do šedotónového zobrazení. Při takto použité metodě tudíž není uvažováno rozložení a vliv samotných barev. Pro každý pixel jsem tak aplikací transformace získal jedno číslo, které jsem použil jako příznak pro natrénování klasifikátorů GNB a SVM. Vzhledem k jednoduchosti metody nepředpokládám dosažení valných výsledků, avšak bude možné si lépe představit přínos přístupu LBP úloze sémantické segmentace.

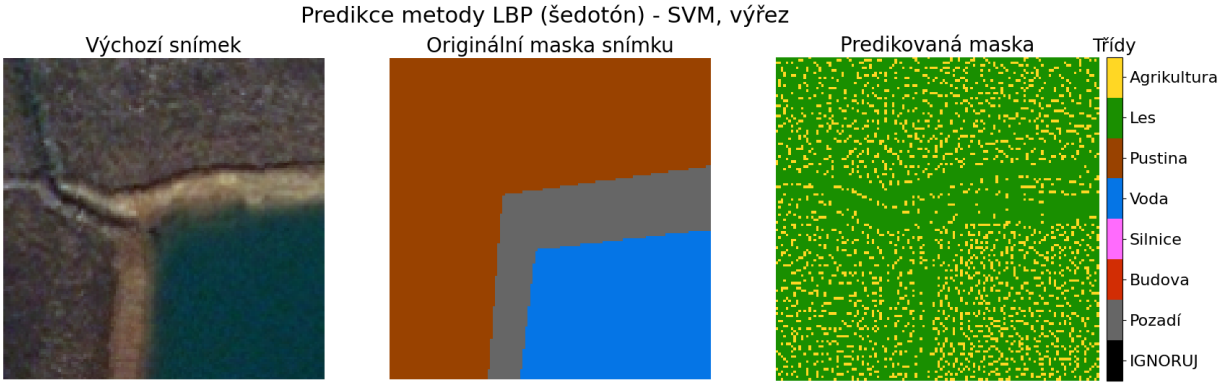
Jelikož je zpracování obrázků pixel po pixelu výpočetně náročné a způsobilost metody je nejasná, rozhodl jsem se natrénovat klasifikátory (GNB a SVM) nejprve na podmnožině trénovací množiny (16 snímků). Protože tak jejich natrénování trvalo několik minut, měl jsem možnost spustit metodu s více vstupními parametry. Jako testovací data jsem použil další obrázek z trénovací množiny. Bylo to z důvodu, že snímky v rámci jedné množiny datasetu mají menší vnitrotřídní variabilitu a tedy by mělo být pro klasifikátor lehčí predikovat správnou třídu. Tento přístup jsem zvolil, neboť jsem potřeboval zjistit, zda se použitá metoda pro řešení úlohy hodí a zda má její řešení nějaký přínos. Vizualizace výsledku metody (SVM) můžete vidět na obrázku 4.3, zvětšenou variantu potom na obrázku 4.4. Realizace metody LBP - GNB je v praxi nepoužitelná, neboť

predikuje pro celý obrázek pouze jednu třídu.

Spíše taktových výsledků jako na obrázcích 4.3 a 4.4 je dosahováno při relativně malém počtu iterací klasifikátoru SVM. V případě velkého množství iterací se informace začne ztrácet a klasifikátor dosahuje horších kriteriálních výsledků (viz tabulka 4.1).



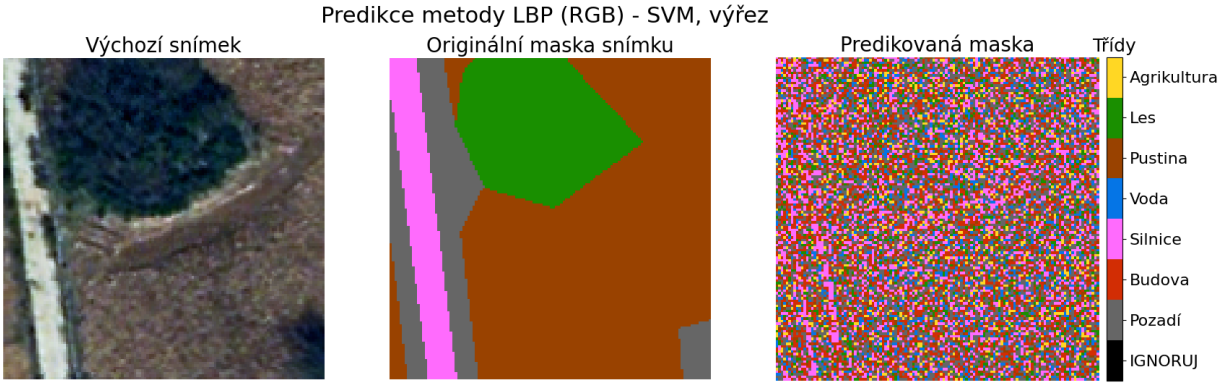
Obrázek 4.3: Vizualizace snímku 23 po použití metody LBP (klasifikátor SVM dle jednoho příznaku)



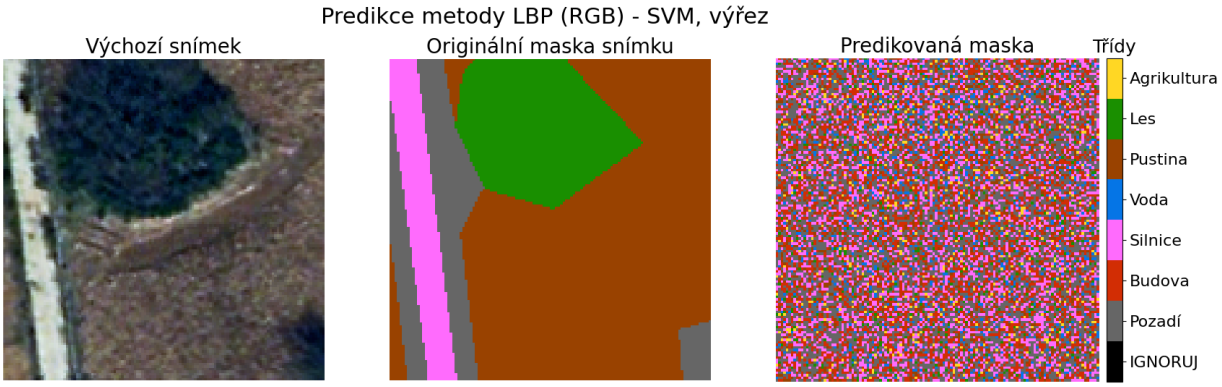
Obrázek 4.4: Vizualizace zvětšené části snímku 23 po použití metody LBP (klasifikátor SVM dle jednoho příznaku).

Po neúspěchu této metody jsem se rozhodl vyzkoušet vytvořit příznakový vektor na základě LBP obrazů z RGB složek původního obrázku, kdy jsem si vytvořil tři šedotónové obrázky reprezentující hodnoty jasu jednotlivých základních barev na snímcích. Tím jsem chtěl rozšířit prostor příznaků a umožnit tak klasifiká-

toru lépe rozhodovat mezi hodnotami pixelů. Výsledek byl též neuspokojivý (viz zvětšené obrázky klasifikátoru SVM 4.5 a 4.6).

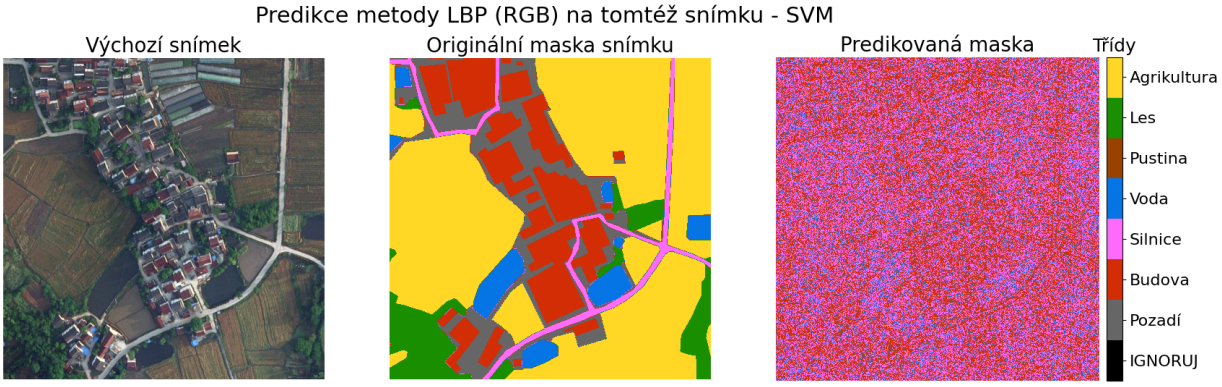


Obrázek 4.5: Vizualizace zvětšené části snímku 23 po použití metody LBP (klasifikátor SVM se třemi příznaky vycházejících ze složek RGB po 4 iteracích).

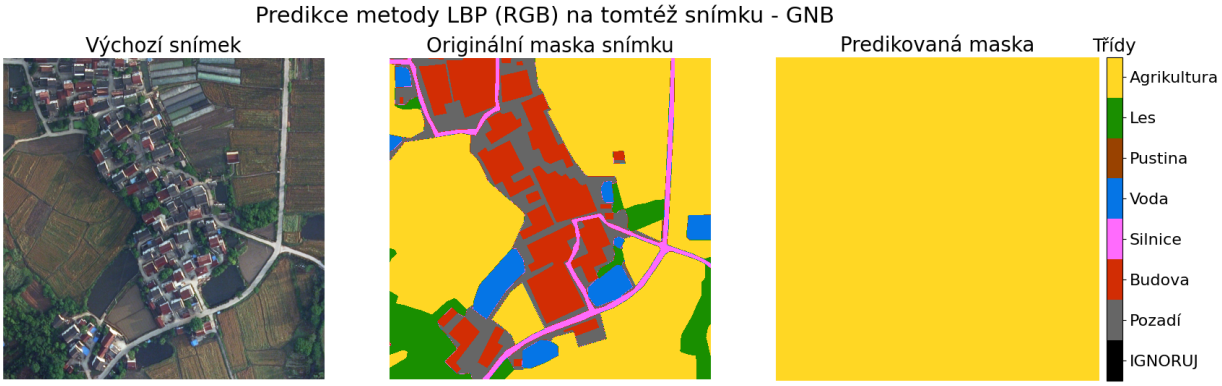


Obrázek 4.6: Vizualizace zvětšené části snímku 23 po použití metody LBP (klasifikátor SVM se třemi příznaky vycházejících ze složek RGB po 20 iteracích).

Abych tedy ukázal jasnou nefunkčnost tohoto přístupu, natrénoval jsem klasifikátor na stejném obrázku, na kterém jsem ho poté otestoval (viz obrázky 4.7, 4.8). Při 100 iteracích SVM klasifikátoru dosáhl přesnosti pouze 18,9 %, což je mírně lepší výsledek, než kdybychom třídy určovali náhodně (to by přesnost byla cca 14,3 %). klasifikátor GNB dosáhl mnohem lepšího výsledku co se přesnosti týče, avšak v ostatních kritériích zaostával.



Obrázek 4.7: Vizualizace použití SVM klasifikátoru (5 iterací) natrénovaného na 1 snímku trénovací množiny datasetu LoveDA na predikci tříd stejného snímku.



Obrázek 4.8: Vizualizace použití GNB klasifikátoru natrénovaného na 1 snímku trénovací množiny datasetu LoveDA na predikci tříd stejného snímku. Vidíme, že klasifikátor klasifikoval celý snímek jednou třídou, přičemž získal relativně dobrou přesnost (accuracy) Tento model je však v praxi nepoužitelný

Výsledky LBP

Jak lze odpozorovat z vizualizací a tabulky 4.1, metoda LBP se nezdá příliš vhodná na tuto segmentační úlohu. Jelí výsledky jsou kvůli jejím omezeným možnostem reprezentace velmi slabé. Ovšem v porovnání s metodou RGB (tedy podobou té nejjednodušší metody segmentace, viz 4.1.1) má část metod LBP k dispozici stejný počet příznaků a přesto dopadla na stejných podmnožinách hůř ve všech sledovaných kritériích než metoda RGB.

V tabulce (4.1) si můžeme všimnout totožných výsledků metod v prvních dvou

sloupcích. To je způsobeno tím, že klasifikátor GNB v obou případech klasifikoval všechny pixely do třídy *Agrikultura* (viz obrázek 4.8), čímž zredukoval úlohu segmentace na úlohu klasifikace do nejpočetnější třídy.

Výsledek segmentace právě jednoho snímku totožného s trénovacím (jako trénovací i testovací množinou zvolen obrázek *Train/6*) dopadla v neprospěch použitelnosti metody LBP. Selhání metody v takto zjednodušeném případě, kdy validujeme na celé trénovací množině, dokazuje, že příznaky metody nejsou schopny zachytit charakteristiky jednotlivých tříd. To přisuzuji její omezené síle a hloubce reprezentace příznaků a kontextu.

Tato metoda je sama o sobě tedy pro tuto problematiku velmi nevhodná a nebudu se jí tudíž dále zabývat. Po provedených testech jsem tak usoudil, že kvůli omezením metody nemá příliš smysl ji trénovat na celém datasetu, neboť by to jeho větší vnitrotřídní rozmanitostí kladlo na metodu ještě větší nároky a výsledek by byl pravděpodobně podobně a více neuspokojivý. Použití nějakého vyhlazovacího filtru by pravděpodobně odhad zlepšila, avšak nepředpokládám, že by šlo o výraznější změnu

4.1.3 HOG

Metoda ze své podstaty není příliš uvažována jako vhodná k segmentaci, neboť nezpracovává obrázek v řádu pixelů, ale v rámci buněk (tedy souborů více pixelů), je tedy nutné přistoupit ke kompromisu. Ten v mém případě spočíval v tom, že jsem výslednou segmentaci prováděl jako klasifikaci větších buněk, nikoliv pixelů. Pro tento účel jsem musel upravit vstupní masky, které jsem taktéž rozdělil na buňky, přičemž každé buňce jsem přiřadil modus ze zastoupených tříd v buňce (tedy třídu v rámci této buňky nejvíce zastoupenou).

Výsledná segmentace implementace je tedy hrubší a ztrácíme detaily.

Kompromis tohoto přístupu mi ovšem umožnil natrénovat klasifikátor (GNB, SVM) na celé trénovací množině v přijatelném čase. Velikosti buněk jsem zvolil 16×16 pixelů (velikost 32 pixelů by byla velmi necitlivá k vnímání silnic a jiných

LBP	**LBP (RGB)**				
Klasifikátor (iterací)	GNB	SVM (4)	GNB	SVM (5)	SVM (100)
Trénovací množina	Train/0-15	Train/0-15	Train/6	Train/6	Train/6
Testovací množina	Train/16-31	Train/16-31	Train/6	Train/6	Train/6
přesnost (accuracy)	0,338	0,131	0,273	0,229	0,189
F1_score	0,072	0,107	0,061	0,135	0,122
mIoU	0,048	0,058	0,039	0,077	0,068

LBP	**LBP (GS)**				**RGB**
Klasifikátor (iterací)	GNB	SVM (10)	GNB	SVM (10)	GNB
Trénovací množina	Train/0-15	Train/0-15	Train/6	Train/6	Train/0-15
Testovací množina	Train/16-31	Train/16-31	Train/6	Train/6	Train/16-31
přesnost (accuracy)	0,338	0,304	0,535	0,277	0,340
F1_score	0,072	0,100	0,100	0,106	0,106
mIoU	0,048	0,062	0,076	0,065	0,068

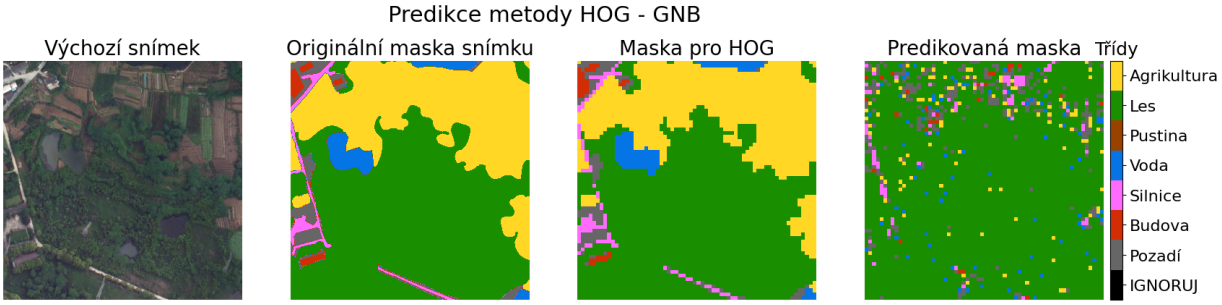
Tabulka 4.1: Tabulka výsledků metody LBP získaných natrénováním na šedotónovém obrázku LBP (GS), nebo na jeho třech barevných kanálech LBP (RGB) s použitím klasifikátorů SVM a GNB na různých trénovacích podmnožinách trénovací množiny (Train) s otestováním (validací) na různých podmnožinách také této množiny. Poslední sloupec znázorňuje počínání metody RGB pro srovnání (viz 4.1.1). Rozsah čísel za lomítkem značí indexy (názvy) příslušných snímků v datasetu.

tenkých linií, neboť 32 pixelů je v kontextu měřítka přibližně 10 m), použil jsem standardní počet směrů (tedy po dvaceti stupních), kdy jsem nechal nastavení bez znaménka (tj. uvažují se hodnoty v rozsahu 0 až 180 stupňů). Takto použité nastavení se obecně osvědčilo v úlohách detekce, na něž byla metoda HOG používána.

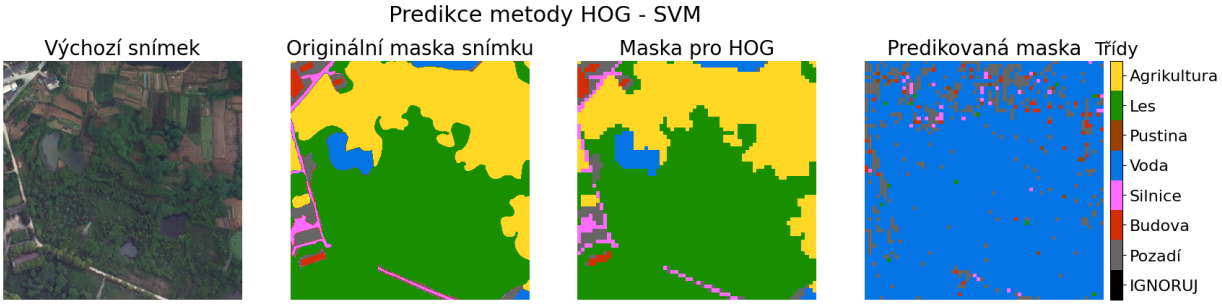
Z důvodu abstrakce příznaků v poměru $(16 \times 16 \times 3) : 9$ (počet přízanků LBP ku počtu příznakům HOG) je co do paměti tato reprezentace úspornější než metoda LBP. Konkrétně v případě velikosti buňky 16×16 má tato metoda přibližně 85 krát paměťově úspornější reprezentaci než metoda LBP (RGB) a cca 28 krát paměťově úspornější reprezentaci než metoda LBP (GS).

U provedených experimentů jsem tedy vycházel především z reprezentace snímku, kdy je každá oblast pixelů buňky (16×16) reprezentována 9 hodnotami (dle hodnot gradientů). Tyto hodnoty v mé implementaci poslouží jako příznaky pro klasifikátory SVM a GNB.

Na obrázcích 4.9 a 4.10 jsou vizualizace, že metoda dokáže zachytit obecné



Obrázek 4.9: Vizualizace použití HOG - klasifikátoru GNB natrénovaného na celé trénovací množině (Train) na obrázek z validační množiny (Val). Maska pro HOG je maska upravená (zmenšená) funkcí modus.



Obrázek 4.10: Vizualizace použití HOG - klasifikátoru SVM se 100 iteracemi natrénovaného na celé trénovací množině (Train) na obrázek z validační množiny (Val). Maska pro HOG je maska upravená (zmenšená) funkcí modus.

Výsledky HOG

Z výsledků metody zaznamenaných v tabulkách 4.2 a 4.3 je patrné, že metoda zvládá úlohu lehce lépe než metoda LBP. To vede k závěru, že metoda poskytuje lepší reprezentaci dat než metoda LBP. Je také (viz výše) mnohem paměťově efektivnější a predikce netrvá příliš dlouho.

Z obrázků 4.9, 4.10 a obrázků v přílohách A.4, A.5 je možné odvodit silné a slabé stránky Bayesova přístupu ke klasifikaci a přístupu klasifikátoru SVM. Zatímco Bayesův přístup (GNB) určuje s větší přesností hojně zastoupené třídy, klasifikátor SVM lépe rozeznává rozdíly mezi třídami (pouze rozdíly, neboť často klasifikuje do špatné třídy). V případě výsledků obou klasifikátorů je možnost implementovat nějaký třídně závislý filtr (vyhlazovací, zvětšovací a zvýhodňovací určité třídy) na základě informací ze zjištěných výsledků. To by pravděpodobně (při správné filtraci) dokázalo výsledky zlepšit.

Na efektivnější implementaci takového filtru jsou však predikce stále příliš nekonzistentní. Dochází ke špatnému rozpoznávání i hojně zastoupených tříd, jako je např. třída *voda*, která nemá takovou vnitrotřídní a vzorovou rozmanitost.

Metoda jako taková dokáže relativně dobře (vzhledem k ostatním třídám a vzhledem k její velikosti) rozpoznávat třídu *silnice* i přesto, že je tato třída z důvodu svého tenkého rozložení velmi znevýhodněna kompresí HOG. Tento jev přisuzuji povaze metody HOG zaměřovat se na gradienty a tedy schopnost efektivně detekovat hrany právě takové, jakými jsou jasové rozdíly mezi silnicí a okolním terénem.

Protože jsou trénování a predikce metody HOG relativně (k LBP) rychlé procesy, měl jsem možnost trénovat a validovat jednotlivé algoritmy na celých množinách datasetu k tomu určených (Train, Val) (viz tabulka 4.2) a Test (viz tabulka 4.3).

Klasifikátor SVM si číselně vedl v důležitých metrikách poměrně hůře než klasifikátor GNB. Za zmínku také stojí rozdíl ve výsledcích u klasifikátorů SVM při rozdílných iteracích na stejných trénovacích a validačních množinách. Zatímco u malé trénovací množiny vede řádové zvýšení iterací ke zlepšení, u velké trénovací množiny vede zvýšení iterací ke zhoršení predikce. Tento jev by se dal přisoudit efektu přetrénování a obtížné separabilitě jednotlivých tříd.

Dalším zajímavým jevem je poté výsledek experimentu zaznamenaný v tabulce 4.2 ve druhém sloupci zprava, kdy je otestována predikce algoritmu na-

trénovaném pomocí klasifikátoru GNB na malé a relativně stejnorodé trénovací množině. Takto natrénovaný model dosál lepších výsledků než ostatní implementace GNB a SVM. Klasifikátor tedy odvádí lepší práci v momentě, kdy je natrénován na méně datech. Můžeme se domnívat, že má tak nejspíše možnost věrněji zachytit alespoň některé třídy a s pomocí jejich lepší klasifikace navýšit hodnotu výstupního kritéria mIoU.

Pro ověření vhodnosti zvolené velikosti buňky jsem provedl experiment s velikostí buňky 8×8 (v tabulce 4.3 první sloupec zprava). Výsledek klasifikátoru GNB byl v tomto případě horší, avšak takřka srovnatelný jako při stejných podmínkách u velikosti buňky 16×16. Tato implementace je však čtyřikrát náročnější na paměť a proces trénování trvá déle. Za těchto okolností jsem se rozhodl účinnost metody HOG pro buňku 8×8 pixelů dále neprozkoumávat.

HOG	**HOG** (16×16)						
Klasifikátor (iterací)	GNB	SVM (10)	GNB	SVM (10)	SVM(100)	GNB	SVM (10)
Trénovací množina	Train/0-15	Train/0-15	Train/0-15	Train/0-15	Train/0-15	Train	Train
Testovací množina	Train/16-31	Train/16-31	Val	Val	Val	Val	Val
přesnost (accuracy)	0,285	0.246	0,272	0,23	0,315	0,188	0,111
F1 score	0,116	0,086	0,117	0,085	0,105	0.104	0,047
mIoU	0,071	0,054	0,072	0,053	0,067	0,061	0,028

Tabulka 4.2: Výsledky metody HOG získané natrénováním klasifikátorů SVM a GNB na různých podmnožinách trénovací množiny (Train) a s otestováním (validací) na různých podmnožinách trénovací množiny (Train) a množině validační (Val). Rozsah čísel za lomítkem značí indexy (názvy) příslušných snímků v datasetu.

4.2 Umělá neuronová síť

K implementaci modelů neuronové sítě použiji výše uvedené technologie (viz 1.3). Pro významné zrychlení trénovacího procesu využiji výpočetní platformy Cuda, díky které mohu trénovat neuronové sítě v řádu jednotek až desítek hodin s pomocí grafické karty na mém osobním počítači.

Neuronová síť DeepLabV3 - Resnet50 (viz 3.2) je určena k segmetnačním

HOG - CL	**HOG** (16×16)				**HOG** (8×8)
Klasifikátor (iterací)	GNB	SVM(1)	SVM(10)	GNB	GNB
Trénovací množina	Train	Train	Train	Train/0-15	Train
Testovací množina	Test (CL)	Test (CL)	Test (CL)	Test (CL)	Test (CL)
mIoU	0,069	0,071	0,061	0,078	0,069

Tabulka 4.3: Výsledky metody HOG získané natrénováním klasifikátorů SVM a GNB na různých trénovacích podmnožinách množiny Train datasetu LoveDA a s otestováním (validací) na množině Test prostřednictvím rozhraní CodaLab v rámci soutěže LoveDA Semantic Segmentation. Rozsah čísel za lomítkem značí indexy (názvy) příslušných snímků v datasetu.

úlohám. Síť je navržena pro jiný počet tříd. To lze změnit pro účel našeho datasetu předefinováním poslední vrstvy sítě tak, aby vracela výstup odpovídající 8 segmentačním třídám.

Zvolené hyperparametry a předzpracování dat

Počet tříd jsem zvolil u všech implementací 8, přestože segmentačně platných jich je pouze 7. To jsem učinil z důvodu, že neuronová síť se naučí jednoduše rozpoznávat pixely třídy *IGNORUJ*, neboť mají hodnotu 0. Zároveň se nám tak tato třída neplete do klasifikace tříd ostatních a testovací skript na platformě Codalab pixely této třídy do vyhodnocení nijak neuvažuje.

Protože mají snímky datasetu velké rozlišení (1024×1024), rozhodl jsem se každý z těchto snímků rozdělit na čtyři menší snímky o velikosti 512×512 a ty použít jako základ vstupních dat neuronové sítě. Stejně jsem učinil s příslušnými maskami k těmto obrázkům

V rámci načítacího modulu jsem určil velikost dávky (batch size) jako 4. Tato hodnota je dost velká na to, aby došlo ke znatelnému urychlení trénovacího procesu, a protože budeme snímky během trénovacího procesu předkládat v náhodném pořadí (volitelný parametr sítě), docílíme tím zároveň větší variability vstupních dat, po nichž bude docházet k aktualizaci vah a prahů sítě, neboť místo jednotlivých obrázků původní velikosti půjde na vstup náhodná čtveřice rozděle-

ných obrázků napříč všemi snímky trénovací množiny. Tato velikost dávky také dobře koresponduje s počtem menších snímků ve větších, což umožňuje následné jednodušší sestavování čtveřic rozdělených snímků zpět do snímků původních pro účely vyhodnocení výsledků predikce na testovací množině na platformě CodaLab.

Na vstupní obrázky jsem poté uplatnil předepsané transformace (rozepsáno níže), které jsou doporučeny využívat při užívání určité architektury (resp. modelu) a to z důvodu, že je síť pro tyto parametry navržena a optimalizována. Další významný důvod použití předepsaných transformací je případ použití předtrénovaných vah sítě. Jedná se o parametry sítě, které jsou předtrénovány na jiném datasetu pro stejný typ úlohy. Tyto parametry (váhy) poté využijeme v pozdějších modelech jako inicializační váhy pro náš trénovací cyklus. Výhodou předtrénované sítě je její schopnost už v začátku spolehlivě zachycovat některé příznaky, což vede k lepším výsledkům a ke výraznému snížení trénovacího času.

Klíčovou transformací je zde normalizace. Při ní dojde k úpravě střední hodnoty a směrodatné odchylky RGB kanálů obrázku a tedy i k jejich přizpůsobení normálnímu rozdělení pravděpodobnosti. Pro optimální chování modelu je předepsána normalizace červené složky pixelů do normálního rozdělení se střední hodnotou 0,485 a směrodatnou odchylkou 0,229. Pro zelenou složku je poté střední hodnota 0,456 a směrodatná odchylka 0,224 a pro modrou složku platí střední hodnota 0,406 a směrodatná odchylka 0,225.

Jako ztrátovou funkci použijeme v našich modelech křížovou entropii (Cross Entropy Loss, viz 3.2.1 a rovnice 3.1)

Optimalizace trénovacího cyklu

Krom implementace výše zmíněných komponent je důležité mít o průběhu trénovacího procesu přehled, aby bylo možné ho efektivně ladit a zlepšovat tak jeho výsledky. Samotná data o ztrátové funkci v trénovací smyčce jsou však omezeným ukazatelem kvality sítě. Abychom dokázali zastavit trénování předtím, než

začne docházet k přetrénování sítě, je užitečné po každé epoše zařadit za trénovací smyčku smyčku validační. Ve validační smyčce dochází k vyhodnocování kvality schopnosti neuronové sítě zobecňovat. Tento proces proběhne po dané epoše tak, že vypočítáme ztrátovou funkci a další případně relevantní metriky (v našem případě přesnost, F1 score a mIoU) u predikcí sítě na validační množině (bez aktualizace parametrů sítě). Ve chvíli, kdy se začne síť v predikcích zhoršovat (hodnota sledovaných metrik na validační množině začne klesat), můžeme uložit váhy modelu s nejlepším výsledkem (zobecněním dle použitých metrik) a vyhnout se tak použití modelu přetrénovaného. Zároveň můžeme použít data o získaných metrikách pro vhodnější nastavení hyperparametrů trénovacího cyklu (použitý optimizer, plánovač apod.)

4.2.1 Trénování modelů

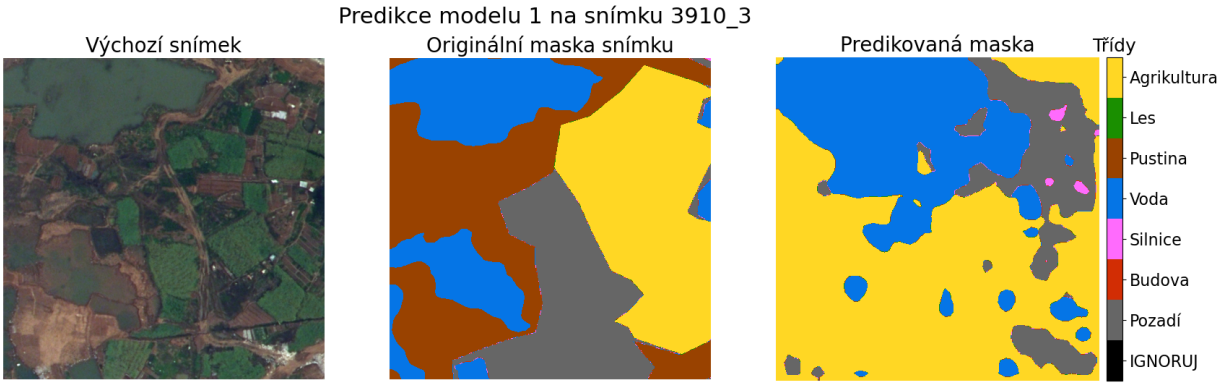
Všechny modely neuronové sítě jsou validovány na celé validační množině (Val). Jejich výsledky jsou uvedeny v tabulce 4.4.

Modely trénované bez validační smyčky

První funkční implementací byl *model 1*. Vznikl natrénováním modelu DeepLabV3-Resnet50 s náhodně inicializovanými počátečními vahami na prvních 16 snímcích trénovací množiny (Train/0-15, stejně jako část implementací metod HOG, LBP a RGB). Tento model se trénoval 10 epoch bez zpětné vazby o kritériích validační smyčky a využíval defaultní implementaci optimizeru Adam bez plánovače. Rozhodl jsem se pro tento optimizer, neboť jsem se domníval, že svou adaptibilitou vyhovuje podstatě úlohy. Dle očekávání nedosáhl *model 1* příliš dobrých výsledků (viz tabulka 4.4), přesto však předčil všechny implementace metod LBP, HOG a RGB co se kritérií mIoU a F1 score týče. Protože je trénovací podmnožina v tomto případě velmi omezená, vyskytují se v predikcích sítě specifické chyby. Tato množina neobsahuje široké silnice a tedy je nedokáže správně klasifikovat, když se vyskytují na obrázcích validační množiny (viz příloha A.6). Zároveň nemá

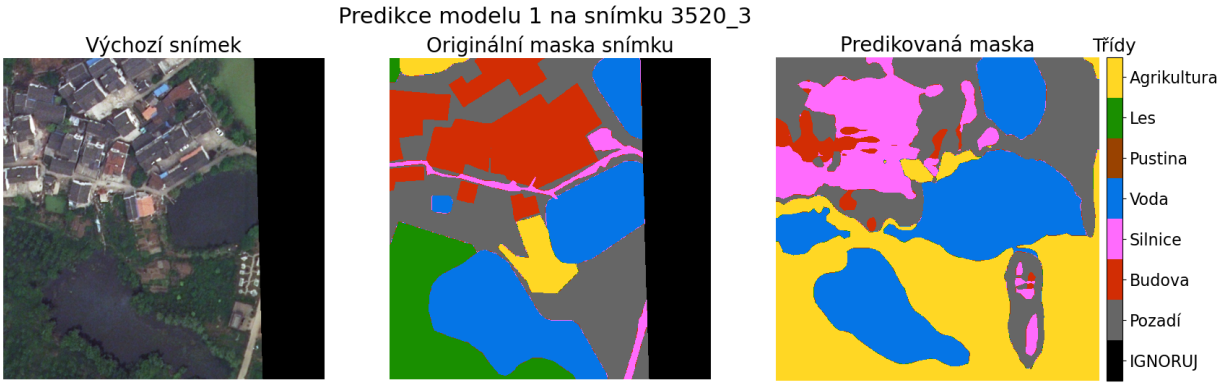
tato omezená podmnožina přítomnu třídu *pustina*, což má za následek úplnou neschopnost sítě tuto třídu predikovat (viz nulové kritérium IoU u třídy *pustina* v tabulce 4.4), což je vidět např. na obrázku 4.11. Dalším důsledkem omezenosti trénovací podmnožiny je, že jsou všechny její snímky úplné, takže neobsahují jediný pixel třídy *IGNORUJ*, což má za následek efekt viditelný na obrázku 4.12 a v příloze A.9, kdy algoritmus přisuzuje pixely třídy *IGNORUJ* třídám, u nichž vnímá nejvyšší pravděpodobnost, čímž vznikají překvapivě smysluplné fantomové predikce neviděného prostoru.

Model 1 jde brát jako simulaci nedostatku trénovacích dat. Podmnožina jeho trénovacích dat je dále příliš konzistentní vzhledem k rozmanitosti ostatních snímků datasetu, a proto model nemá možnost naučit se o mnoho lepší zobecňovací schopnosti. Jeho výstupy mají pak sice lepší výsledky než klasické metody, avšak stále nejsou použitelné a spolehlivé. V soutěži LoveDA Semantic Segmentation challenge dosáhl model skóre mIoU 0,147.



Obrázek 4.11: Predikce modelu 1 na části snímku 3910 s rozlišením 512 × 512 pixelů.

Model 2 přímo vychází z *modelu 1* (stejný optimizer - Adam, počet epoch a s absencí informace validační smyčky, náhodná inicializace vah). Jediným rozdílem je, že byl *model 2* natrénován na celé trénovací množině. V celkových kritériích na validační množině si vede pochopitelně mnohem lépe než *model 1*. Zlepšení zaznamenal v segmentaci všech tříd (viz tabulka 4.4). Dokáže nyní segmentovat



Obrázek 4.12: Predikce modelu 1 na části snímku 3520 s rozlišením 512 × 512 pixelů.

třídu *pustina*, stále však s velmi neuspokojivým výsledkem. Největší přírůstek kritéria IoU metoda zaznamenala u třídy *silnice*, což, domnívám se, úzce souvisí se schopností mnohem lépe klasifikovat široké silnice (viz příloha A.6) a se zlepšením schopnosti rozeznávat budovy a silnice. Model také již umí poznat třídu *IGNORUJ* (viz příloha A.9). V trénovacích datech modelu byly tentokrát i městské oblasti s bohatou zástavbou, model je tedy spolehlivější co se týče segmentace budov a zastavěných oblastí, jak jde vidět na obrázku 4.13.

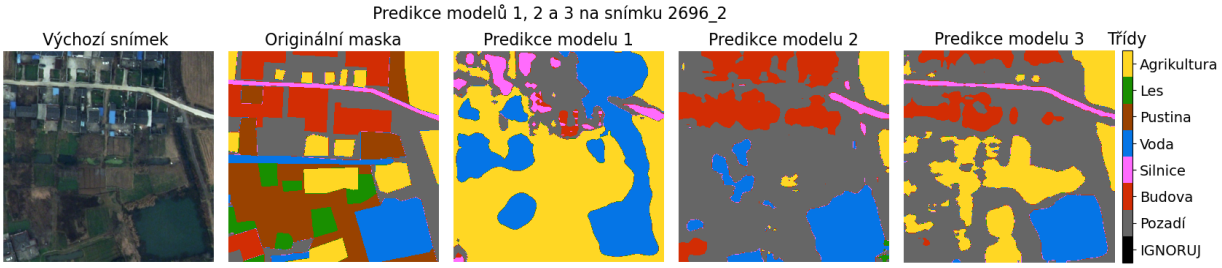
Model 2 je ve všech pozorovaných ohledech lepším modelem než *model 1*. Stále však není dostatečně optimalizován a kvůli absenci validační smyčky nemůžeme zabránit případnému nedotrénování nebo přetrénování. Výstup modelu je však již použitelný a dle metrik v tabulce 4.4 vykazuje větší spolehlivost než model předchozí. V soutěži LoveDA Semantic Segmentation obdržel model skóre mIoU 0,385.

Model 3 je nastavením *modelu 2*, kdy trénovací cyklus běžel o 20 epoch déle. Ve všem ostatním je identický. Ukázalo se, že ačkoliv se průměrná hodnota ztrátové funkce na validační množině téměř zdvojnásobila, všechna ostatní relevantní kritéria zaznamenala lepší výsledek než u předchozího modelu (viz tabulka 4.4). Tento jev přisuzuji větší nerozhodnosti modelu - efektu, kdy ve výsledných tensorech (následně transformovaných do predikované masky) obecně vyšly méně rozhodující pravděpodobnosti pro správnou třídu (menší rozdíl mezi pravděpo-

dobností přiřazení do správné třídy a pravděpodobností k ní opačnou), ovšem po vybrání maximálních pravděpodobností bylo dosaženo větší úspěšnosti. *Model 3* výrazněji vyrovnává některé neduhy *modelu 2*, jako je například zlepšení kvality segmentace třídy *agrikultura* (dle IoU v tabulce 4.4) a třídy *budova*. Zejména u budov je vidět podstatné zlepšení v zachycování jejich pozice i tvaru oproti předchozím modelům v příloze A.12.

Model 3 je již použitelný model, ač má stále problémy s generalizací a segmentací některých tříd. Model dokáže věrněji zachycovat rozhraní jednotlivých tříd, přestože má ve svých predikcích stále znatelné rezervy. V soutěži LoveDA Semantic Segmentation dosáhl tento model skóre mIoU 0,444.

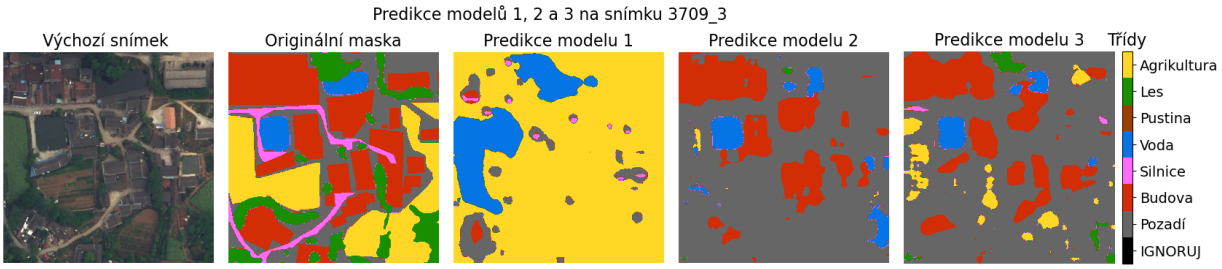
Ve výsledcích *modelů 1, 2* a *3* je vidět znatelný pokrok. Kriteriální úspěšnosti mIoU *modelu 1* podobné klasickým metodám byly napříč *modely 2* a *3* cca trojnásobně zlepšeny a z místy nahodilých predikcí se staly vcelku použitelné aproximace. Modely však byly natrénovány slepě vzhledem k efektům přetrénování a nedotrénování. Dalším krokem je tedy optimalizace trénovacího cyklu pomocí monitorování průběžného vývoje kritérií na validační smyčce.



Obrázek 4.13: Predikce modelů 1, 2 a 3 na části snímku 2696 s rozlišením 512 × 512 pixelů.

Modely trénované s pomocí validační smyčky

Následující trojice modelů se od trojice minulé liší v tom, že v rámci trénovacího cyklu má již implementovanou validační smyčku. Mimo hodnoty ztrátové funkce trénovací smyčky (která má tendenci stále klesat) tak máme přístup k důleži-

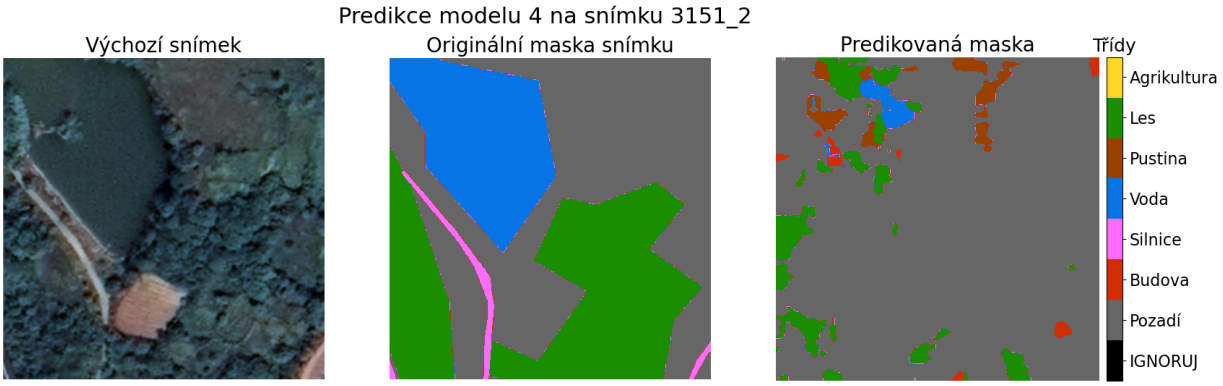


Obrázek 4.14: Predikce modelů 1, 2 a 3 na části snímku 3709 s rozlišením 512 × 512 pixelů.

tým metrikám smyčky validační. Konkrétně se jedná o průměrnou ztrátu epochy, průměr kritéria F1 score pro všechny třídy a kritérium IoU pro každou třídu a jejich průměr mIoU. Ač mají všechny tyto metriky vypovídající hodnotu o kvalitě segmentace, v našem případě budeme dávat přednost kritériu mIoU, které je hodnotícím kritériem v soutěži LoveDA Semantic Segmentation. Sledování těchto kritérií během trénování neurovoné sítě nám umožní uložit model, který dosáhne maximální hodnoty kritéria mIoU a tím vybrat nejefektivnější model z průběhu celého procesu trénování. V přídadě, kdy pak začne s probíhajícími epochami klesat hodnota metrik IoU (mIoU), F1 score a přesnosti (accuracy), dochází již v trénovací smyčce k přetrénování modelu, kdy model dále ztrácí schopnost generalizace tím, že se příliš přispůsobuje trénovacím datům.

Model 4 je tedy prvním modelem, který vznikl na základě nejlepšího výsledku kritéria mIoU v průběhu trénování. Startovní váhy jsou inicializovány náhodně. Stále zde používám optimizer Adam, a protože mohl průběh vývoje kritéria vypadat různě, rozhodl jsem se model spustit na 60 epoch, což je výrazně více epoch než u předchozích modelů. Nejlepšího výsledku dosáhl model v 19. epoše (viz tabulka 4.4), přesto však dosáhl horších výsledků než model předchozí. To přisuzuji náhodnému faktorů v podobě předkládání snímků a náhodné inicializaci počátečních vah, které mohly způsobit, že ztrátová funkce v tomto případě konvergovala k lokálnímu minimu s vyšší hodnotou. Model má v kritériích IoU pro většinu tříd podobnou úspěšnost. Výjimku tvoří třídy *pustina* a *les*. Příklad chybné predikce

velké plochy lesa je znázorněn na obrázku 4.15. V soutěži LoveDA Semantic Segmentation dosáhl skóre mIoU 0,428. To je nižší hodnota než u *modelu 3*, což je po předešlé analýze očekávaný výsledek.



Obrázek 4.15: Predikce modelu 4 na části snímku 3151 s rozlišením 512 × 512 pixelů.

Model 5 byl vybrán ze sekvence modelů trénovaných 30 epoch, neboť 60 epoch minulého modelu bylo zbytečně mnoho. Protože předchozí model dosáhl nejlepšího výsledku v 19. epoše, rozhodl jsem se uměle u *modelu 5* desetinásobně zmenšit po 19. epoše velikost konstanty učení optimizeru Adam. Záměrem bylo, aby byl model v okolí tohoto minima zvládnul dokonvergovat blíže nalezenému minimu. Model však dosáhl svého celkového nejlepšího výsledku mIoU již v 16. epoše. Než se tedy uplatnilo umělé zmenšení velikosti konstanty učení plánovačem, další tři epochy již docházelo k přetrénování a následná změna velikosti konstanty učení již nedokázala dosáhnout lepších výsledků. Oproti předchozímu modelu *model 5* lépe rozpoznává (vzhledem k IoU) třídy *pozadí, voda, les* a *budova* (viz tabulka 4.4). Obzvláště v segmentaci budov *model 5* zaznamenává velký pokrok (jak je vidět např. na obrázku 4.18), kdy dokáže predikovat relativně přesné shluky budov na rozdíl od *modelu 4*, který v okolí budov na tomto obrázku vytváří neurčitý počet shluků s nejednoznačným ohraničením. *Model 5* je sice lepším modelem než *model 4*, stále však v hlavním kritériu mIoU zaostává za *modelem 3*. Tento jev si lze potvrdit i z výsledných predikcí modelů, např. porovnáním obrázků 4.13 a 4.16. V soutěži LoveDA Semantic Segmentation však dosáhl *model 5* skóre mIoU

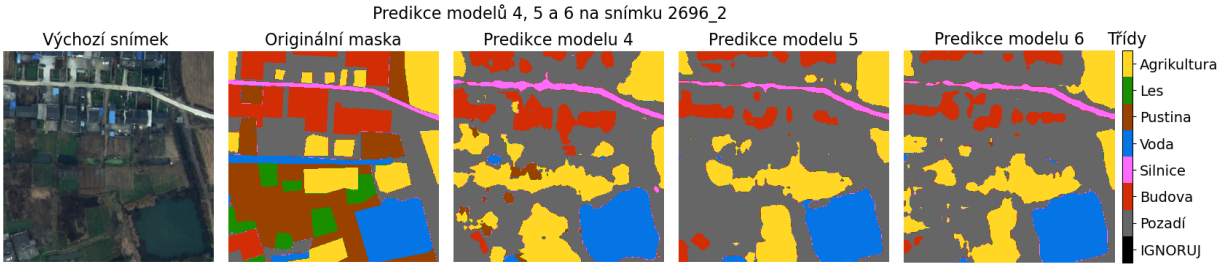
nejlepšího ze všech předchozích a to 0,446. Takové nepředvídatelné rozdíly jsou pravděpodobně způsobeny podstatou datasetu, kdy snímky v testovací množině pochází z jiné geografické oblasti, a mohou tak způsobit méně předvídatelné chování segmentačního algoritmu (viz 2.2.1).

Model 6 vychází z *modelu 5*. Používá stejný optimizer a taktéž byl trénován na 30 epoch při náhodných startovních vahách. Abych potlačil nevýhodu předchozí nepredikovatelnosti modelu, zavedl jsem zde podmíněný plánovač, který ve chvíli, kdy hodnota kritéria mIoU validační smyčky přesáhla určitou stanovenou hodnotu (hodnotu blízkou hodnotě maximální minulého modelu), snížil velikost konstanty učení na desetinu původní hodnoty. Tento model dosáhl svého maxima v 24. epoše a vykazuje lepší výsledky než všechny modely předchozí (viz tabulka 4.4). Dle kritérií IoU *model 6* nad modely předešlými dominuje v segmentaci tříd *pozadí, budova* a *silnice*. Na *model 5* poté lehce ztrácí v segmentaci třídy *les* a *voda*, na *model 3* pak ztrácí ve zbylých třídách *pustina* a *agrikultura*. Na obrázku 4.18 lze vidět zlepšení segmentační schopnosti budov, kdy od výsledku předchozího modelu vymizely některé chybně predikované shluky této třídy.

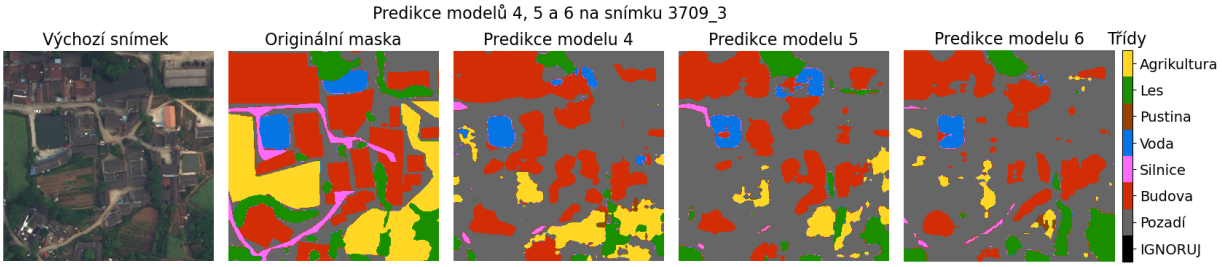
Model 6 vychází, co se týče metrik validační smyčky, nejlépe ze všech v této práci dosud představených modelů. Na testovacím datasetu v soutěži LoveDA Semantic Segmentation však dosáhl skóre mIoU pouze 0,443. To ho řadí v úspěšnosti až za *model 5* (který získal skóre 0,446) i za model *model 3* (který dosáhl na mIoU 0,444). Podobný jev se vyskytoval již u *modelu 5* a je pravděpodobné, že jeho původ tkví v odlišné textuře krajiny jiné geografické oblasti, jejíž snímky testovací množina obsahuje, nebo v náhodné inicializaci vah.

Predikční schopnost členitého prostředí s velkým množstvím tříd a hranicemi mezi nimi *modelů 4, 5, 6* lze pozorovat na obrázku 4.17. Jde vidět, že modely mají ve svých predikcích rezervy, avšak v porovnání s předchozí trojicí modelů (viz obrázek 4.14), je zlepšení *modelů 4, 5, 6* patrné. Příklad obrázku 4.16 nám

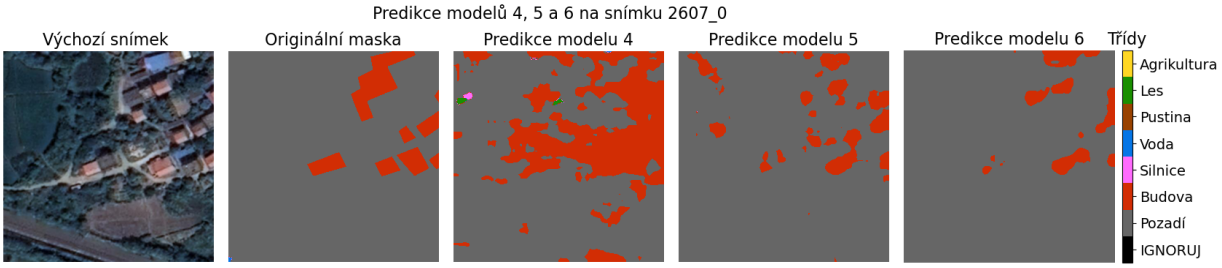
poté ukazuje, že predikce modelů uvažují o třídě *pozadí* ve správném kontextu - totiž jako o skutečném pozadí na němž se vyskytují shluky jiných tříd (typicky tříd *budova* a *silnice*). Pravděpodobně kvůli velké vnitrotřídní variabilitě této třídy dochází k tomu, že je v predikcích nadhodnocována na úkor především tříd *budova* a *pozadí*. Právě rozlišení mezi třídami *budova* a *pozadí* dělá neuronové síti problémy (tento jev je viditelný např. na obrázku 4.16), následkem čehož má třída *pustina* ve všech modelech zdaleka nejnižší IoU.



Obrázek 4.16: Predikce modelů 4, 5 a 6 na části snímku 2696 s rozlišením 512 × 512 pixelů.



Obrázek 4.17: Predikce modelů 4, 5 a 6 na části snímku 3709 s rozlišením 512 × 512 pixelů.



Obrázek 4.18: Predikce modelů 4, 5 a 6 na části snímku 2607 s rozlišením 512 × 512 pixelů.

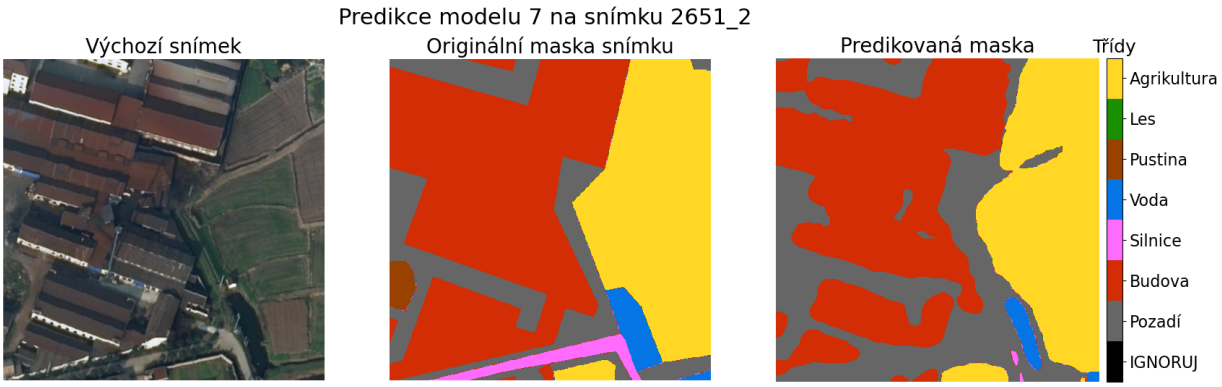
Modely s optimizerem SGD

Pro všechny následující modely jsem se rozhodl místo dosavadně používaného optimizeru Adam použít optimizer SGD. Učinil jsem tak z důvodu, že výsledky modelů stále nebyly schopné příliš konkurovat výsledkům v soutěži. Usoudil jsem, že by mohlo výsledky vylepšit použití jiného optimizeru. U optimizeru SGD je nutné zvolit jako parametr konstantu učení a případně momentum (parametr udávající vliv minulé změny vah na tu současnou).

Nejprve jsem vyzkoušel nový optimizer na *modelu 7*. Použil jsem stejný koncept plánovače jako u *modelu 6*, kdy se po určité prahové hodnotě mIoU (konkrétně dosaženo v 15. epoše) desetkrát zmenšila velikost konstanty učení (z původních 0,01 na 0,001). Jako momentum optimizeru jsem zvolil 0,9, což znamená, že se většina minulých aktualizací podílí na aktualizaci současné. Startovní váhy byly nastaveny náhodně. Model dosáhl nejvyššího kritéria mIoU dvě epochy po této změně. Jeho reprezentativní verzí je tedy ten natrénován na 17 epochách. Již při trénování bylo patrné, že jsou výsledky na validační množině znatelně lepší než v případě trénování předešlých modelů. Průměr ztrátové funkce je poté poměrně nižší než u předchozí trojice modelů. Ve všech metrikách na validačním datsetu *model 7* překonal všechny modely přechozí (viz tabulka 4.4). Dokonce je překonal i v IoU kritériu pro každou třídu validační množiny. Největšího zlepšení segmentace dle IoU se dostalo třídě *voda* (jak je vidět např. na porovnání obrázků 4.17 a 4.22). Na obrázku 4.19 poté lze vidět velmi kvalitní segmentaci tohoto modelu, kde je v podstatě jedinými většími chybami vynechání shluku třídy *pustina* a větší části třídy *silnice*. S třídou *pozadí* však mají problém zatím modely všechny.

Model 7 již produkuje použitelné a kvalitní výstupy. Ve všech ohledech překonal modely předešlé a v soutěži LoveDA Semantic Segmentation získal na testovací množině skóre mIoU 0,492. To je oproti výsledkům předchozích modelů znatelně lepší výkon, který již stačil na přesun ze spodních pozic tabulky účast-

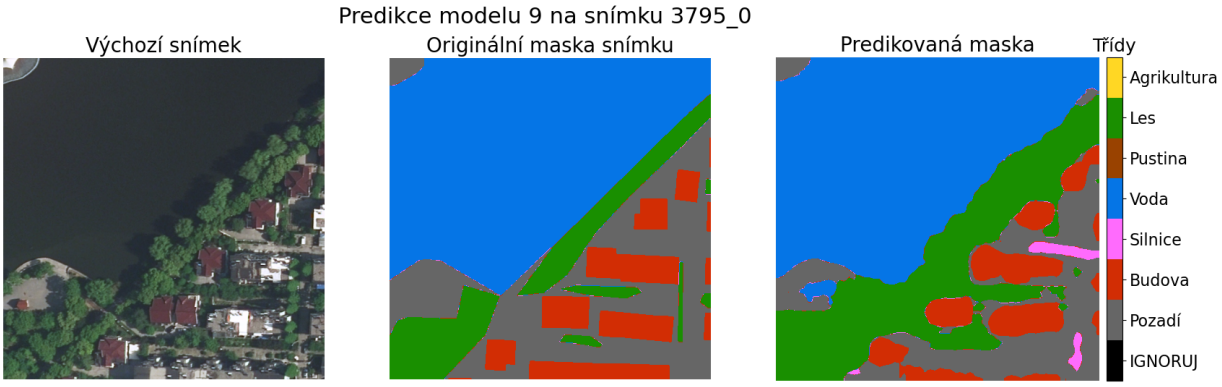
níků o něco výše.



Obrázek 4.19: Predikce modelu 7 na části snímku 2651 s rozlišením 512×512 pixelů.

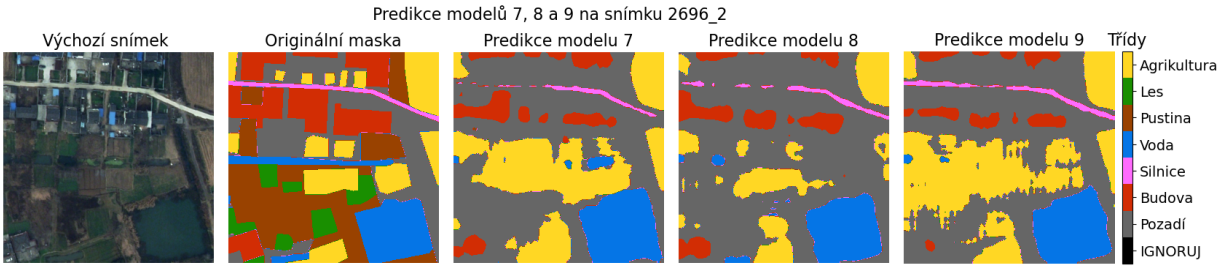
Model 8 vychází taktéž z optimizeru SGD s výchozí hodnotou 0,01, čímž se podobá předchozímu modelu. Účelem tohoto modelu bylo zdokonalit model předchozí a zlepšit jeho výsledek na náhodně inicializovaných vahách. Využil jsem zde poprvé možnost plánovače snížit v průběhu velikost konstanty učení vícekrát. Při trénování jsem tedy v každé z epoch 2, 4, 6, 8 a 16 dvakrát zmenšil velikost konstanty učení. Tato volba měla zlepšit a zpřesnit konvergenci modelu do minima. Nejlepšího kriteriálního ohodnocení mIoU dosáhla trénovací smyčka v epoše 10. Tento model překonal všechna kritéria validační množiny (mIoU, přesnost, F1 score) modelu předchozího v řádu setin (viz tabulka 4.4). Z tohoto hlediska se tedy jedná o dosud nejúspěšnější model. Z jednotlivých tříd zaznamenal *model 8* největší zlepšení v segmentaci třídy *voda*. V soutěži LoveDA Semantic Segmentation však získal *model 8* skóre mIoU 0,489, což je naopak v řádu setin horší výsledek než v případě předchozího modelu. Ač je tedy *model 8* na predikci testovací množiny horší než *model 7*, je obtížné hodnotit, který z modelů je objektivně lepším segmentačním algoritmem. Každý z nich lehce vyniká v různých oblastech a jejich výsledky jsou si kvalitativně velmi blízké (viz obrázky 4.22, 4.21) a přílohy A.8, A.13.

Model 9 obohatil koncept předchozího modelu o předtrénované váhy na datasetu COCO [35]. Protože má dataset COCO jiný počet tříd něž náš segmentační problém, bylo nutné načíst model odpovídající datasetu COCO a následně upravit jeho poslední vrstvu, aby vracela predikce pro 8 tříd. Takto předdefinovaný model má výhodu v tom, že již jeho vrstvy obsahují funkční extraktory příznaků, v následku čehož zpravidla dochází k rychlejší konvergenci sítě a k lepším výsledkům datasetu. Z toho důvodu jsem dvojnásobně zmenšil velikost počáteční konstanty učení optimizeru SGD (tedy na hodnotu 0,005). Plánovači jsem poté naordinoval vždy dvojnásobně zmenšení velikosti konstanty učení v epochách 3, 4 a 15. Domněnka ohledně rychlejší konvergence se potvrdila, když dosáhl model nejvyššího kritéria mIoU v pouhé 6. epoše z celkových 20 epoch trénovacího cyklu. *Model 9* dosáhl nejnižší hodnoty ztrátové funkce a nejlepší hodnoty F1 score na validační množině ze všech dosud představených modelů (viz tabulka 4.4). V kritériu mIoU o jednu setinu zaostává za předchozím modelem, což ale (jak je popsáno u modelů předchozích) nemusí mít vliv na výsledné kritérium na množině testovací. Model dosáhl většího pokroku (a nejvyššího skóre IoU ze všech předchozích modelů) v segmentaci tříd *budova, agrikultura* a *silnice*. Velké zlepšení modelu v segmentaci širokých silnic je vidět v příloze A.13. Ukázka relativně kvalitní segmentace budov je uvedena na obrázku 4.20. Model naopak výrazněji zaostává v segmentaci třídy *les* za *modelem 7* a v segmentaci tříd *pustina* a *voda* za *modelem 8*. V predikcích *modelu 9* a dvou modelů předchozích jsou na první pohled rozdíly. Na obrázku 4.21 je vidět jeho lepší segmentace třídy *silnice* a velmi odlišné chování k segmentaci třídy *agrikultura*, kdy jsou hrany a konzistence shluků narušovány třídou *pozadí*. V příloze A.11 lze poté vidět, že model jako jediný odhalil přítomnost tříd *les* a *agrikultura*, přestože jejich predikce nejsou nikterak přesné. Na stejném obrázku je poté vidět horší segmentace třídy *voda*, kdy jsou její shluky menší než v originální masce i v predikcích předchozích dvou modelů. V soutěži LoveDA Semantic Segmentation dosáhl model skóre mIoU 0,481. To je o více než setinu horší než *model 7*, který tedy nadále zůstává modelem nejlépe umístěným.



Obrázek 4.20: Predikce modelu 9 na části snímku 3795 s rozlišením 512 × 512 pixelů.

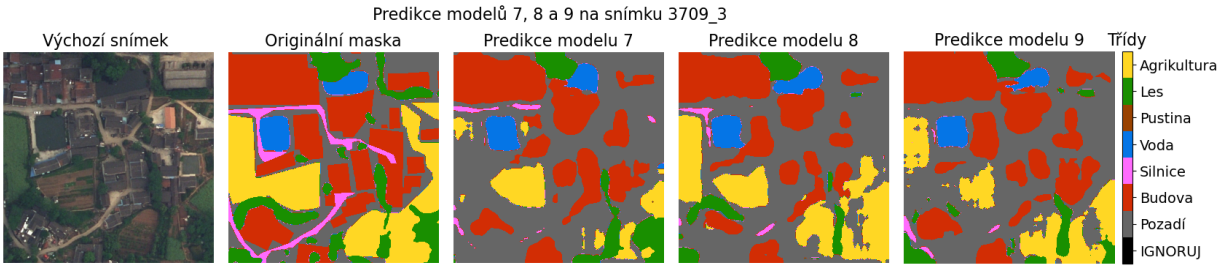
Výsledky *modelů 7, 8, 9* jsou si vzhledem k ostatním dvěma trojicím vzájemně nejpodobnější. Výběr nejlepšího modelu z těchto tří je poté otázkou preferencí na kvalitu segmentace jednotlivých tříd. Některé rozdíly mezi jednotlivými modely jsou patrné z obrázků 4.21, 4.22 a z příloh A.8, A.11, A.13. V rámci přílohové části je možné přehledně porovnat výstupy *modelů 1* až *9* na dvojici stejných obrázků. Výstupy *modelů 7, 8, 9* jsou potom již použitelné a v soutěži LoveDA Semantic segmentation za nejlepším výsledkem mIoU zaostávají o přibližně 6 setin. Umísťují se tak v poslední třetině výsledkové tabulky soutěže.



Obrázek 4.21: Predikce modelů 7, 8 a 9 na části snímku 2696 s rozlišením 512 × 512 pixelů.

Model 10

Zkušenosti získané z předchozích tří trojic modelů mi posloužily pro tvorbu *modelu 10*. Tento model také vycházel z předtrénovaných vah na datasetu COCO



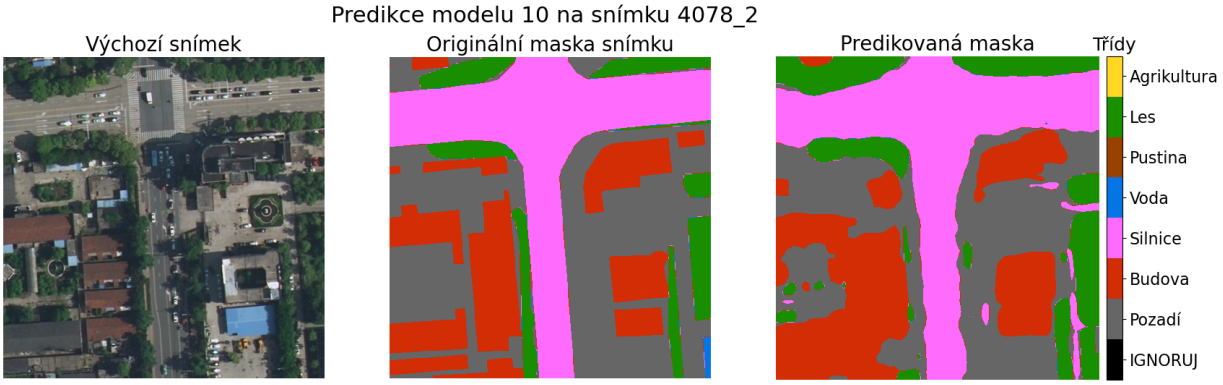
Obrázek 4.22: Predikce modelů 7, 8 a 9 na části snímku 3709 s rozlišením 512 × 512 pixelů.

(jako u *modelu 9*). Plánovač vycházel z původní hodnoty konstanty učení 0,005 a byl nastaven na její dvojnásobné zmenšení v epochách 3, 4 a 15 (stejně jako u předchozího modelu). Zásadním rozdílem tedy bylo zkoumání nejen modelu s nejvyšším mIoU na validační množině, nýbrž i ostatní přijatelné verze modelu (model se tentokrát ukládal po každé epoše a měl jsem tedy k dispozici 20 modelů v různých fázích procesu trénování).

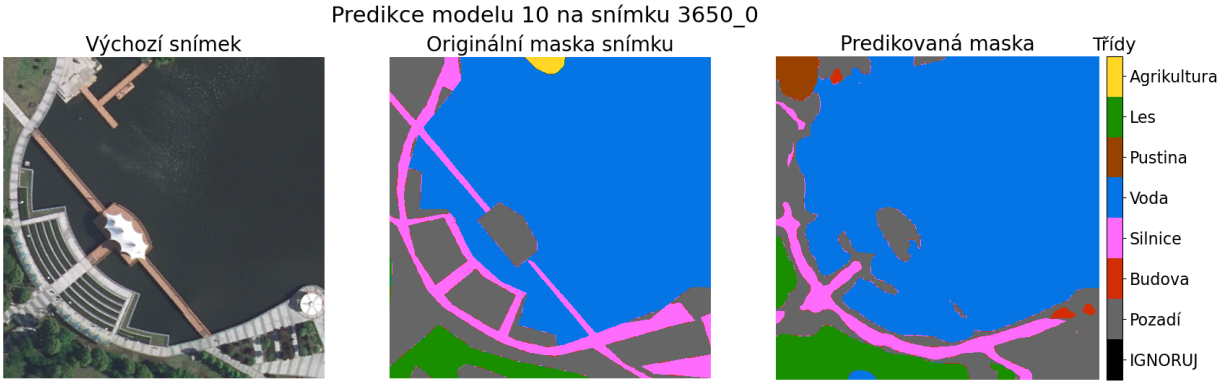
Na grafu 4.26 vidíme záznam metrik IoU pro různé třídy a jejich průměru mIoU v jednotlivých epochách modelu. Přestože trénovací cyklus dosáhl nejlepšího kritéria mIoU po 12. epoše, po otestování na testovacím datasetu v rámci soutěže dopadl mnohem lépe model uložený po proběhnutí 6. epochy. Je možné si všimnout, že do 6. epochy byl trend mIoU rostoucí. Potom začalo kritérium mIoU oscilovat a v 12. epoše (pravděpodobně vlivem náhlého zlepšení segmentace třídy *agrikultura*) kritérium mIoU překonalo svou hodnotu z 6. epochy. Je tedy možné, že se model náhodou přiblížil nějaké charakteristice validační množiny.

Jak je vidět na tabulce 4.4, *model 10* dosáhl značně nižší hodnoty ztrátové funkce než modely předchozí. Zároveň má model ze všech nejvyšší F1 score a jako první model přesáhl na validační množině hodnotu mIoU 0,5 a to se zlepšením o dvě setiny oproti *modelu 9*. Za *modelem 8* zaostává v segmentaci třídy *voda* a *silnice* (vzájemné porovnání modelů na obrázku 4.25). Ve všech ostatních třídách vykazuje nejlepší výsledky ve smyslu IoU ze všech v této práci uvedených modelů. Protože *model 10* překonal *model 9* ve všech kritériích, je možné ho odůvodněně označit jako jeho vylepšení (což dává smysl i z hlediska implementace, která je

krom drobných vylepšení metod monitorování cyklu, shodná). Příklad predikcí *modelu 10* je vidět na obrázcích 4.23 a 4.24. V přílohách A.14, A.15 a A.16 je pak k vidění porovnání modelů na jiných snímcích. Společně s *modelem 3* pak obrázky v přílohách (a s obrázkem 4.25) znázorňují průřez vývojem modelů v této práci.

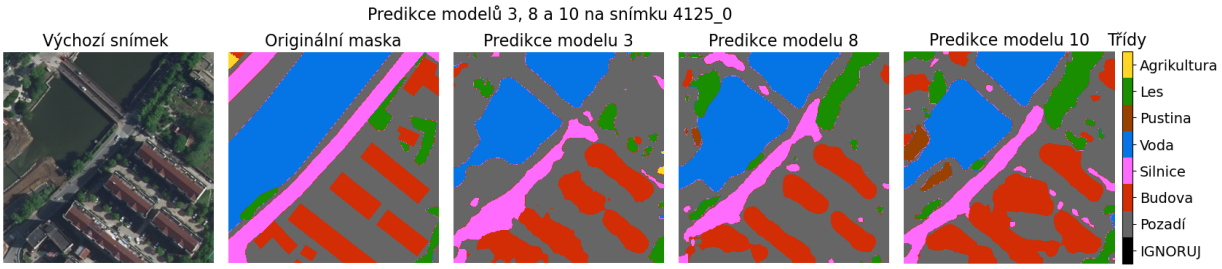


Obrázek 4.23: Predikce modelu 10 na části snímku 4078 s rozlišením 512 × 512 pixelů.



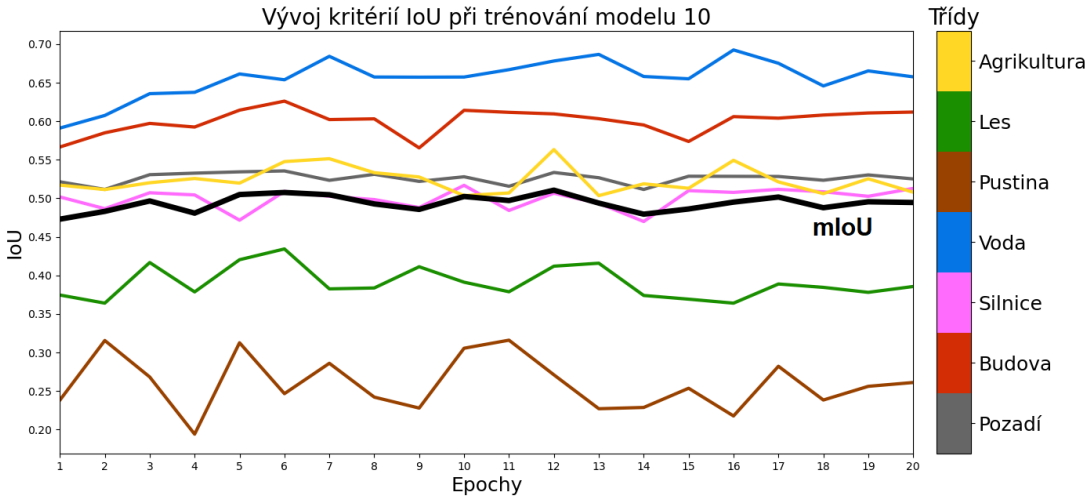
Obrázek 4.24: Predikce modelu 10 na části snímku 3650 s rozlišením 512 × 512 pixelů.

Průběh ztrátové funkce napříč epochami trénování je znázorněn v grafu 4.27. Jak jsem vysvětloval u modelů výše, nejlepší model z hlediska mIoU na validační množině nemusí mít nejnižší hodnotu ztrátové funkce. To je případ i *modelu 10*, kdy ztrátová funkce dosáhla nejnižší hodnoty po třetí epoše. Model z třetí epochy však nedosahuje takové kvality (z hlediska mIoU) jako model z epochy šesté.

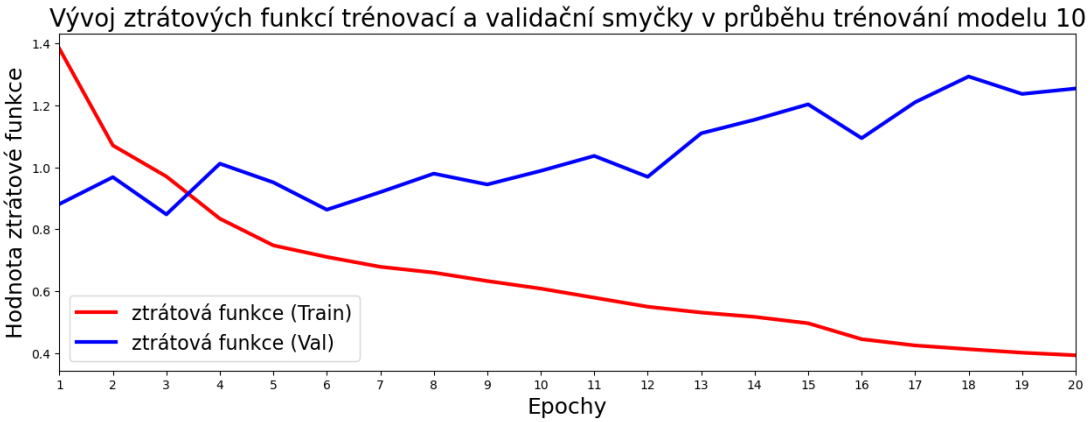


Obrázek 4.25: Predikce modelů 3, 8 a 10 na části snímku 4125 s rozlišením 512×512 pixelů.

Protože se může hodnota ztrátové funkce validační smyčky skokově měnit (dojde mezitím k hodně aktualizacím vah), je lepším vodítkem pozorovat výkyvy v trendu. Před epochou 6 hodnota ztrátové funkce dvakrát za sebou poklesla. To znamená, že se predikce trénovací smyčky dvě epochy po sobě více zlepšovaly než zhoršovaly. Je méně pravděpodobné, že je tento jev náhodný (v porovnání s jednorázovými poklesy v průběhu cyklu) a lze se tedy domnívat, že bude mít tento model větší kvality než modely po jednorázovém poklesu. *Model 10* dosáhl v soutěži LoveDA Semantic Segmentation skóre mIoU na testovací množině 0,503. To je nárůst o více jak setinu oproti druhému v pořadí - *modelu 7*. Je to tedy nejlepší výsledek ze všech v této práci implementovaných metod a jeden z hlavních výstupů této práce.



Obrázek 4.26: Záznam vývoje kritérií IoU a jejich průměru mIoU napříč trénovanými epochami trénovacího cyklu modelu 10.



Obrázek 4.27: Záznam vývoje ztrátových funkcí trénovací a validační smyčky napříč epochami trénování modelu 10.

4.2.2 Vyhodnocení umělých neuronových sítí

Tabulka 4.4 obsahuje číselná vyhodnocení *modelů 1* až *10*.

Model 10 dosáhl v soutěži LoveDA Semantic Segmentation skóre miou 0,503. Tento výsledek se v tabulce zveřejněných výsledků umístil na 75. pozici z celkových 93 zveřejněných. Model zaostává za nejlepšími modely v soutěži v kritériu mIoU o přibližně 5 setin.

model neuronové sítě	**model 1**	**model 2**	**model 3**	**model 4**	**model 5**	**model 6**	**model 7**	**model 8**	**model 9**	**model 10**
optimizer	Adam	Adam	Adam	Adam	Adam	Adam	SGD	SGD	SGD	SGD
plánovač	ne	ne	ne	ne	ne	ano	ano	ano	ano	ano
epoch trénovacího cyklu	*10	10	30	60	30	30	30	20	20	20
epoch modelu (dle mIoU)	-	-	-	19	16	24	17	10	5	6 (12)
ztrátová funkce (Train)	*0,951	0,704	2,090	**0,149**	4,113	0,327	0,183	0,272	0,231	0,393
mIoU (Train)	*0,305	0,558	0,672	**0,900**	0,705	0,769	0,860	0,802	0,828	0,728
ztrátová funkce (Val)	2,013	1,593	3,045	2,948	4,460	1,813	1,548	1,323	1,224	**0,863**
přesnost (accuracy) (Val)	0,320	0,548	0,609	0,605	0,595	0,634	0,675	0,679	0,683	**0,699**
F1_score (Val)	0,141	0,265	0,315	0,306	0,315	0,32	0,355	0,356	0,357	**0,364**
mIoU (Val)	0,130	0,345	0,419	0,397	0,413	0,431	0,482	0,487	0,486	**0,507**
IoU (Val) - pozadí	0,195	0,435	0,497	0,469	0,473	0,502	0,524	0,526	0,524	**0,535**
IoU (Val) - budova	0,120	0,379	0,546	0,475	0,529	0,562	0,587	0,587	0,599	**0,626**
IoU (Val) - silnice	0,072	0,419	0,442	0,449	0,403	0,47	0,491	0,489	0,501	**0,509**
IoU (Val) - voda	0,187	0,416	0,427	0,464	0,509	0,469	0,604	**0,668**	0,639	0,654
IoU (Val) - pustina	0	0,119	0,230	0,186	0,194	0,225	0,239	**0,264**	0,218	0,246
IoU (Val) - les	0,021	0,317	0,359	0,310	0,383	0,367	0,431	0,388	0,395	**0,434**
IoU (Val) - agrikultura	0,319	0,327	0,433	0,422	0,400	0,424	0,496	0,485	0,523	**0,548**
mIoU (Test)	0.147	0.385	0,444	0,428	0,446	0,443	0,492	0,489	0,481	**0,503**

* model se týká zmenšeného datasetu Train/0-15

Tabulka 4.4: Výsledky natrénovaných modelů neuronové sítě na trénovací (Train), validační (Val) a testovací (Test) množině. Tučně jsou označeny minima ztrátových funkcí a maxima ostatních kritérií. Epoch modelu udává, po kolikáté epoše z celého trénovacího cyklu byl model uložen a déle netrénován.

5 Závěr

V této práci jsem objasnil oblasti využití a technologii satelitních snímků společně s problematikou jejich automatické segmentace. Tuto problematiku jsem přiblížil pomocí definic základních pojmů a následnou charakteristikou několika datasetů, které se této problematice věnují a podporují její výzkum. Datasety se liší ve své velikosti, rozlišení snímků, počtu tříd a každý má svá unikátní specifika (např. segmentace plodin dle různých fází jejich růstu, nebo čistě segmentace budov).

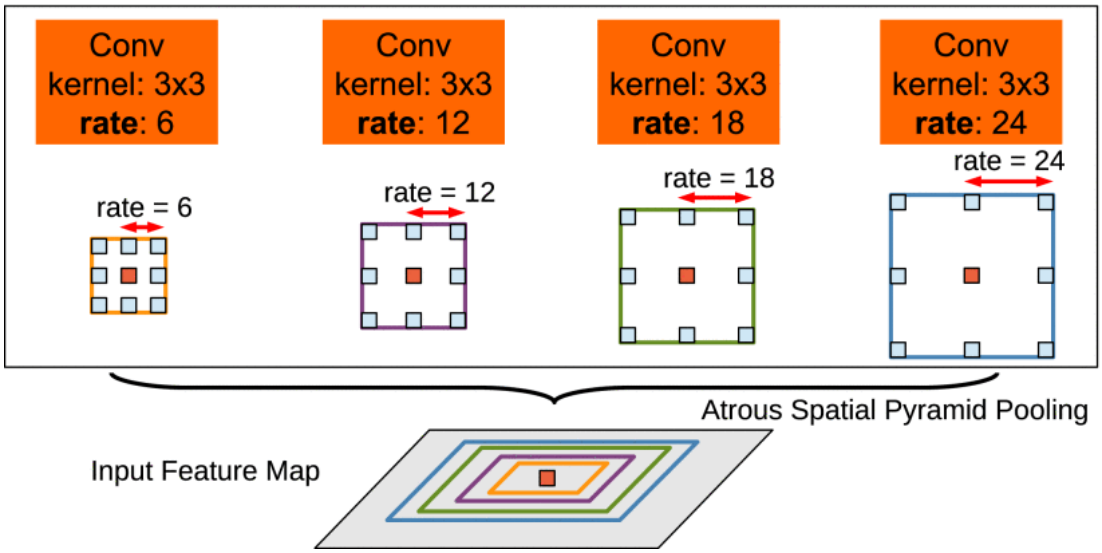
Pro další experimenty jsem si vybral dataset LoveDA. Ten klade svou různorodostí snímaných oblastí velké nároky na zobecňovací schopnost segmentačního algoritmu. Tento dataset jsem hlouběji rozebral a zpracoval statistiky vypovídající o jeho specifických vlastnostech. Kvůli nerovnoměrnému zastoupení jeho tříd jsem pak uvedl definici a vysvětlení relevantních metrik, dle kterých byla posouzena kvalita následně vytvořených a rozebraných algoritmů.

Aplikace metod LBP a HOG nebyla příliš efektivní. Metoda HOG vykázala v mnoha ohledech lepší schopnost segmentace než metoda LBP, která ze své podstaty nedokázala ve svých příznacích pojmout dostatečnou informaci o třídách datasetu. Metoda LBP pak byla překonána také principiálně jednoduchou metodu RGB, jež vykazovala sémanticky přijatelnější výsledky. Lepších výsledků metod by mohlo být dosaženo zařazením určitých metod předzpracování (např. filtrací průměrovým filtrem), případně rozšířením vektoru příznaků o příznaky z jiných metod. Ve srovnání s účinností neuronových sítí se však použití těchto metod s ruční extrakcí příznaků jednoznačně nevyplatí.

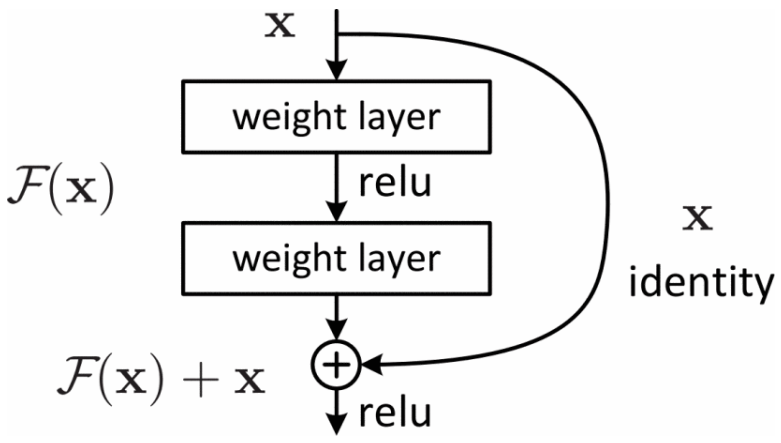
Umělá neuronová síť si v porovnání s metodami HOG a LBP vedla dle očekávání lépe. Její predikce tvořily s vývojem modelů stále smysluplnější shluky, které nakonec překonaly hodnotu 50 % mIoU. Predikce neuronových sítí však mají své limity, což se projevilo i na natrénovaných modelech. To je kromě obtížné podstaty datasetu způsobeno mnoha faktory, z nichž za zmínku stojí nekonzistence

anotátorů, kdy jsou často velké rozdíly mezi stylem anotace některých tříd a oblastí. Segmentace některých tříd je přesto více než dostatečná (např. třída *voda*). Segmentace některých tříd je naopak velmi neuspokojivá napříč všemi modely (např. třída *pustina*). K dosažení lepších predikčních schopností sítě by mohla vést obměna a prozkoumání více možných komponent, jako jsou jiné druhy optimizerů, plánovačů a případně ztrátových funkcí (s možností definovat vlastní). Segmentační schopnost modelů by se pak dala vylepšovat metodami augmentace, kdy bychom zvětšili variabilitu trénovacích dat jejich jednoduchými transformacemi (např. rotací, transpozicí, osovým převrácením). Dalším z mnohých možných vylepšení by bylo použít k predikci váženou kombinaci výstupů více modelů, jež by měly uspokojivé výsledky na určitých třídách.

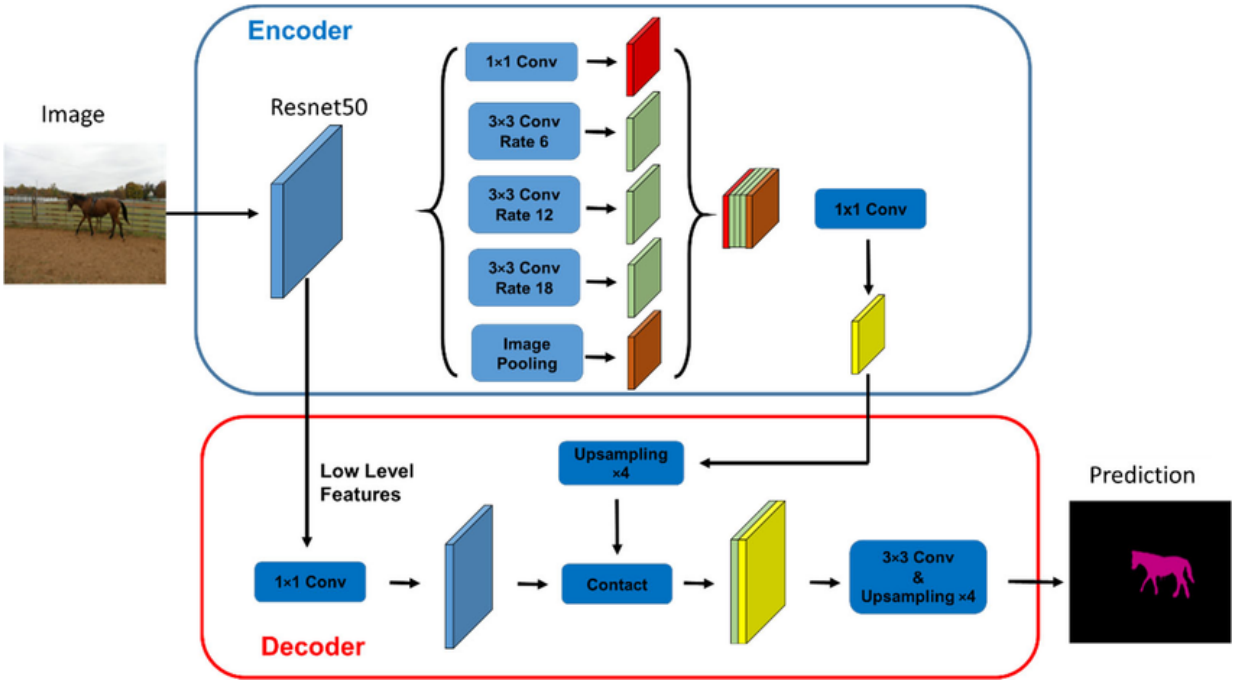
A Přílohy



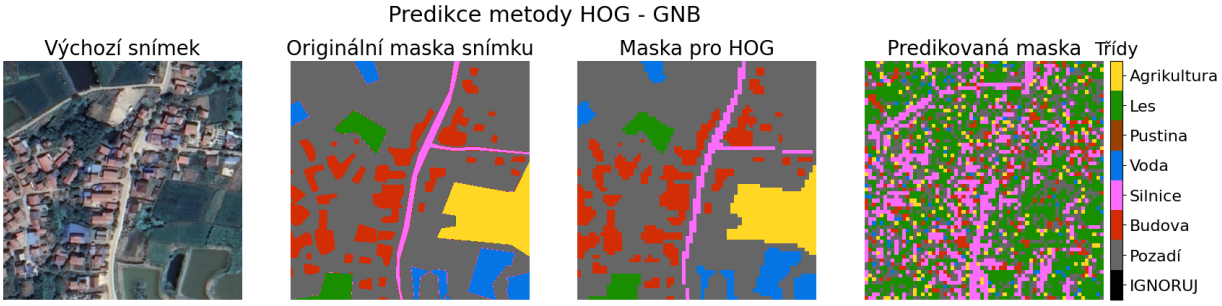
Obrázek A.1: Atrous Spatial Pyramid Pooling (ASPP) [29]. Pro klasifikaci středového pixelu (oranžový) využívá ASPP víceškálové funkce pomocí více paralelních filtrů s různými dilatacemi. Efektivní zorná pole jsou zobrazena v různých barvách [29].



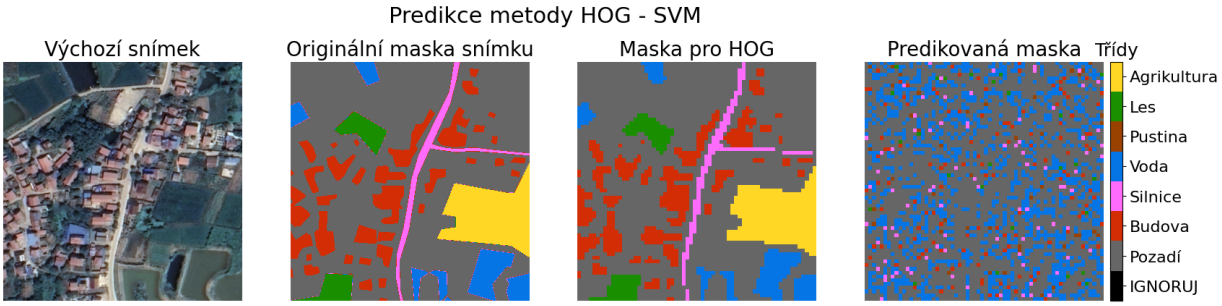
Obrázek A.2: Znázornění reziduálního bloku [].



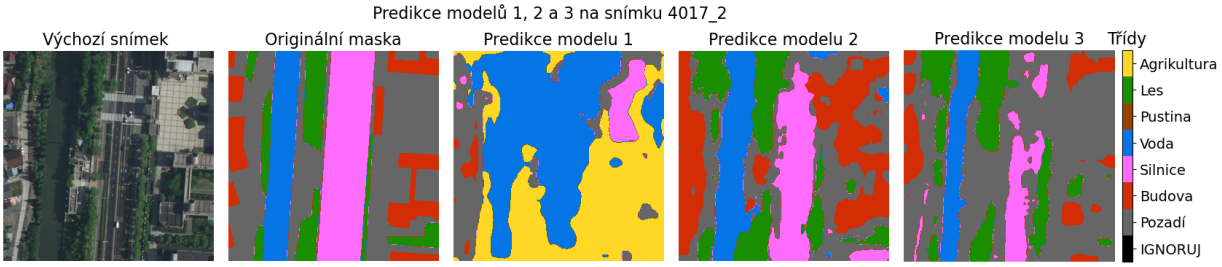
Obrázek A.3: Struktura sítě kombinující DeepLabV3 a ResNet50 [36]



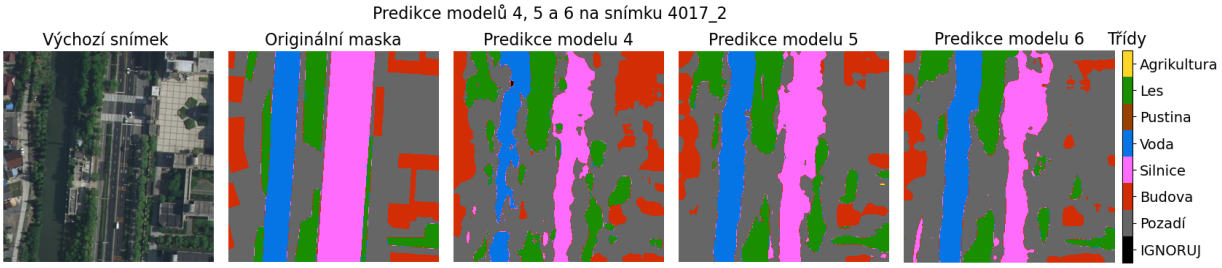
Obrázek A.4: Vizualizace použití HOG - klasifikátoru GNB natrénovaného na celé trénovací množině na obrázek z množiny validační. Maska pro HOG je maska upravená (zmenšená) funkcí modus. Snímek 2588



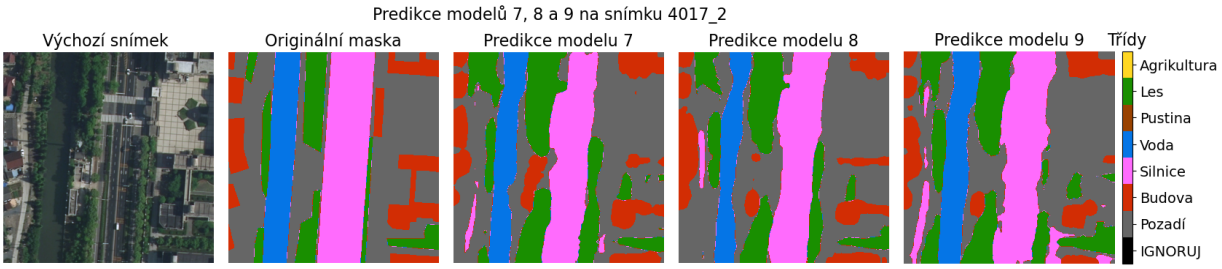
Obrázek A.5: Vizualizace použití HOG - klasifikátoru SVM se 100 iteracemi natrénovaného na celé trénovací množině na obrázek z validační množiny. Maska pro HOG je maska upravená (zmenšená) funkcí modus. Snímek 2588



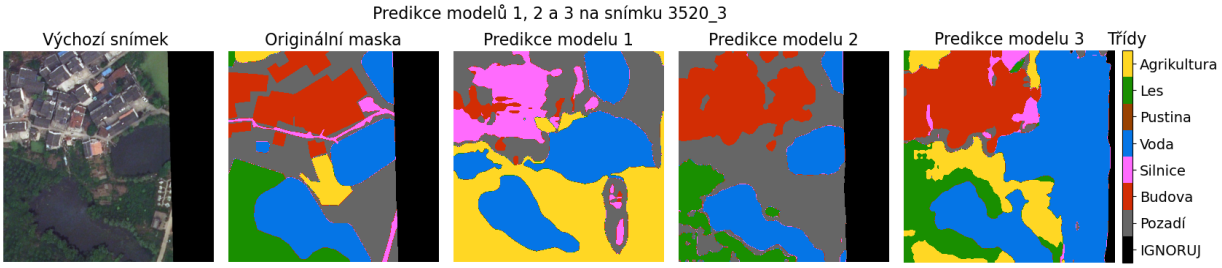
Obrázek A.6: Predikce modelů 1, 2 a 3 na části snímku 4017 s rozlišením 512×512 pixelů.



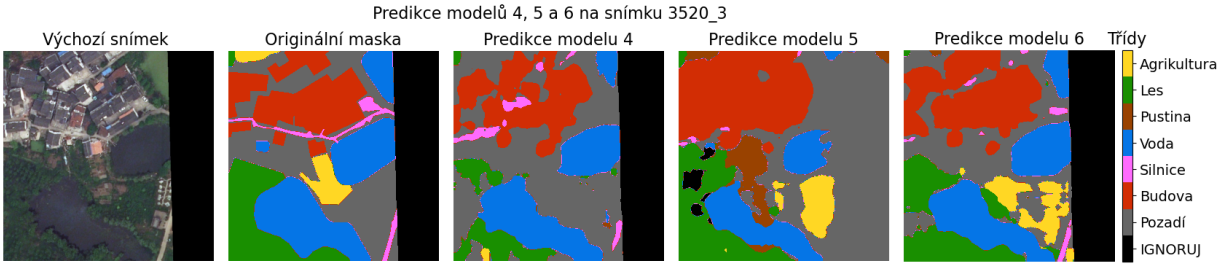
Obrázek A.7: Predikce modelů 4, 5 a 6 na části snímku 4017 s rozlišením 512×512 pixelů.



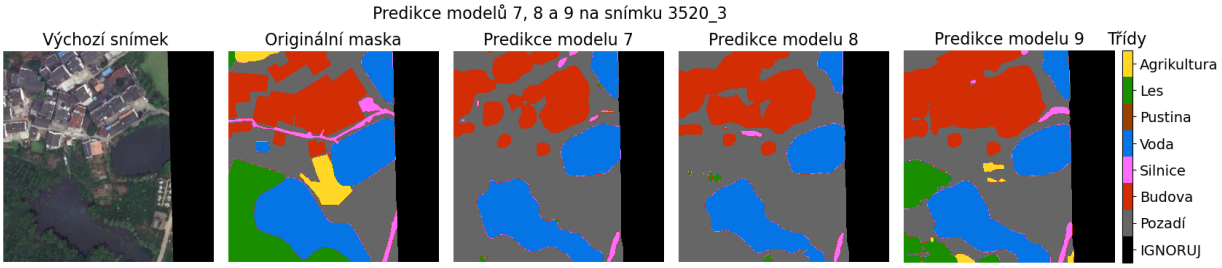
Obrázek A.8: Predikce modelů 7, 8 a 9 na části snímku 4017 s rozlišením 512×512 pixelů.



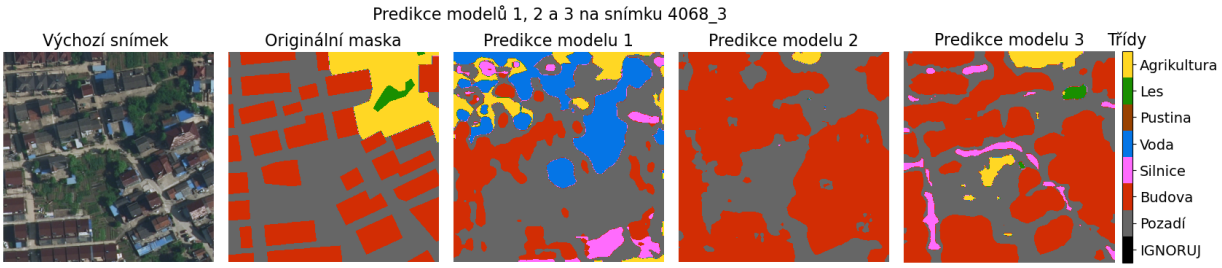
Obrázek A.9: Predikce modelů 1, 2 a 3 na části snímku 3520 s rozlišením 512×512 pixelů.



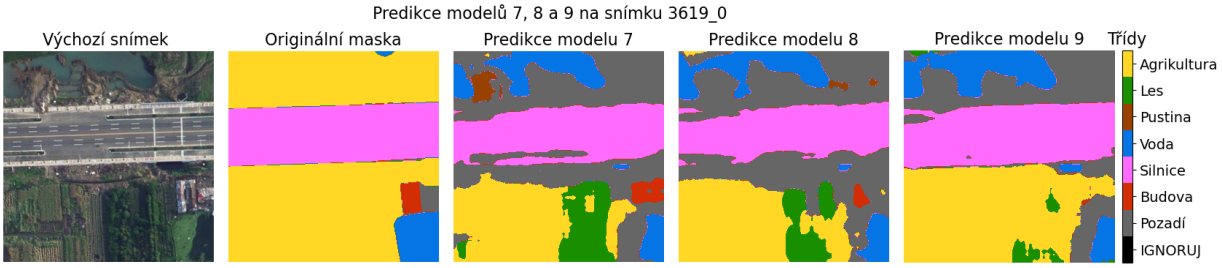
Obrázek A.10: Predikce modelů 4, 5 a 6 na části snímku 3520 s rozlišením 512×512 pixelů.



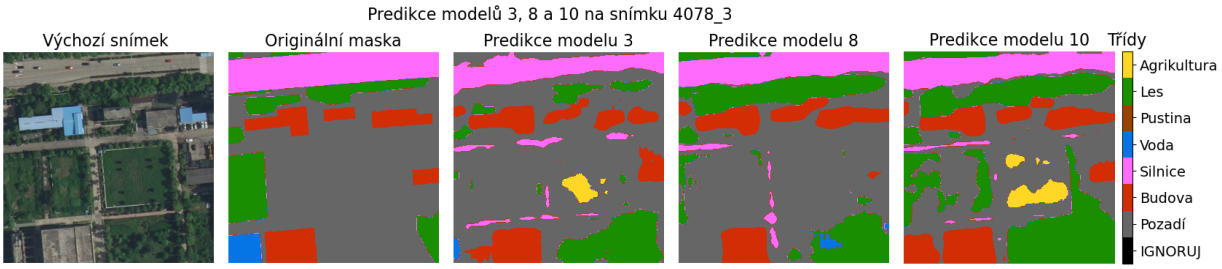
Obrázek A.11: Predikce modelů 7, 8 a 9 na části snímku 3520 s rozlišením 512×512 pixelů.



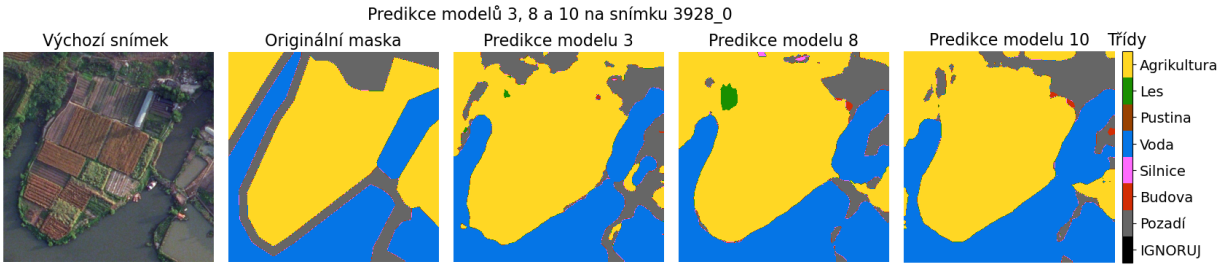
Obrázek A.12: Predikce modelů 1, 2 a 3 na části snímku 4068 s rozlišením 512×512 pixelů.



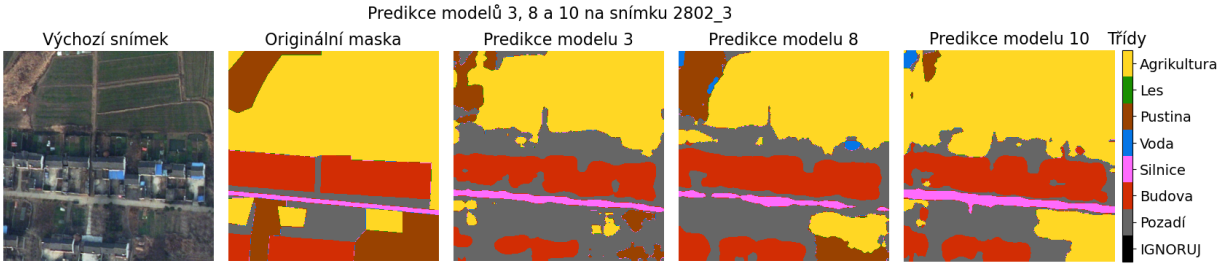
Obrázek A.13: Predikce modelů 7, 8 a 9 na části snímku 3619 s rozlišením 512×512 pixelů.



Obrázek A.14: Predikce modelů 3, 8 a 10 na části snímku 4078 s rozlišením 512×512 pixelů.



Obrázek A.15: Predikce modelů 3, 8 a 10 na části snímku 3928 s rozlišením 512×512 pixelů.



Obrázek A.16: Predikce modelů 3, 8 a 10 na části snímku 2802 s rozlišením 512×512 pixelů.

Literatura

[1] I. L. Turner, M. D. Harley, R. Almar, and E. W. Bergsma, "Satellite optical imagery in coastal engineering," *Coastal Engineering*, vol. 167, p. 103919, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S037838392100079X

[2] Google, "Google earth," 2023, [Software]. Version 7.3.2.5776. Mountain View, CA: Google.

[3] G. Van Rossum and F. L. Drake Jr, *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[4] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[6] NVIDIA, P. Vingelmann, and F. H. Fitzek, "Cuda, release: 10.2.89," 2020. [Online]. Available: https://developer.nvidia.com/cuda-toolkit

[7] R. E. Foundation, "Landcovernet europe: A geographically diverse land cover classification training dataset," Radiant MLHub, 2022.

[8] X.-Y. Tong, G.-S. Xia, Q. Lu, H. Shen, S. Li, S. You, and L. Zhang, "Land-cover classification with high-resolution remote sensing images using transferable deep models," *Remote Sensing of Environment*, vol. 237, p. 111322, 2020.

[9] V. Sainte Fare Garnot and L. Landrieu, "Panoptic segmentation of satellite image time series with convolutional temporal attention networks," *ICCV*, 2021.

[10] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, "Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark," in *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2017.

[11] J. Wang, Z. Zheng, A. Ma, X. Lu, and Y. Zhong, "Loveda: A remote sensing land-cover dataset for domain adaptive semantic segmentation," in *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, J. Vanschoren and S. Yeung, Eds., vol. 1. Curran Associates, Inc., 2021. [Online]. Available: https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/4e732ced3463d06de0ca9a15b6153677-Paper-round2.pdf

[12] A. Van Etten, D. Lindenbaum, and T. M. Bacastow, "Spacenet: A remote sensing dataset and challenge series," *arXiv preprint arXiv:1807.01232*, 2018.

[13] J. Iqbal and M. Ali, "Weakly-supervised domain adaptation for built-up region segmentation in aerial and satellite imagery," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 167, p. 263–275, 2020.

[14] J. Castillo-Navarro, B. Le Saux, A. Boulch, N. Audebert, and S. Lefèvre, "Semi-Supervised Semantic Segmentation in Earth Observation: The MiniFrance suite, dataset analysis and multi-task network study," *Machine Learning*, vol. 111, pp. 3125–3160, 2022. [Online]. Available: https://hal.science/hal-03132924

[15] X.-Y. Tong, G.-S. Xia, and X. X. Zhu, "Enabling country-scale land cover mapping with meter-resolution satellite imagery," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 196, pp. 178–196, 2023.

[16] A. Boguszewski, D. Batorski, N. Ziemba-Jankowska, T. Dziedzic, and A. Zambrzycka, "Landcover.ai: Dataset for automatic mapping of buildings, woodlands, water and roads from aerial imagery," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2021, pp. 1102–1110.

[17] J. Wang, Z. Zheng, A. Ma, X. Lu, and Y. Zhong, "LoveDA: A remote sensing land-cover dataset for domain adaptive semantic segmentation," Oct. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5706578

[18] J. Wang, A. Ma, Y. Zhong, Z. Zheng, and L. Zhang, "Cross-sensor domain adaptation for high spatial resolution urban land-cover mapping: From airborne to spaceborne imagery," *Remote Sensing of Environment*, vol. 277, p. 113058, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0034425722001729

[19] A. Pavao, I. Guyon, A.-C. Letournel, X. Baró, H. Escalante, S. Escalera, T. Thomas, and Z. Xu, "Codalab competitions: An open source platform to organize scientific challenges," *Technical report*, 2022. [Online]. Available: https://hal.inria.fr/hal-03629462v1

[20] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[21] T. Chan, G. Golub, and R. LeVeque, "Updating formulae and a pairwise algorithm for computing sample variances," Stanford University, Stanford working paper STAN-CS-79-773, 1979.

[22] Z. Guo, L. Zhang, and D. Zhang, "A completed modeling of local binary pattern operator for texture classification," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1657–1663, 2010.

[23] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1. Ieee, 2005, pp. 886–893.

[24] A. Ajit, K. Acharya, and A. Samanta, "A review of convolutional neural networks," in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, 2020, pp. 1–5.

[25] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[26] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6230–6239.

[27] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[28] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv:1706.05587*, 2017.

[29] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[30] E. Khoshnevisan, H. Hassanpour, and M. M. AlyanNezhadi, "Profile face recognition based on elements by normalizing global and local features," in *2022 8th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS)*, 2022, pp. 1–6.

[31] R. Azad, M. Heidari, M. Shariatnia, E. K. Aghdam, S. Karimijafarbigloo, E. Adeli, and D. Merhof, "Transdeeplab: Convolution-free transformer-based deeplab v3+ for medical image segmentation," in *Predictive Intelligence in Medicine*, I. Rekik, E. Adeli, S. H. Park, and C. Cintas, Eds. Cham: Springer Nature Switzerland, 2022, pp. 91–102.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[33] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[35] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Doll'a r, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[36] L. Kou, M. Sysyn, S. Fischer, J. Liu, and O. Nabochenko, "Optical rail surface crack detection method based on semantic segmentation replacement for magnetic particle inspection," *Sensors*, vol. 22, p. 8214, 10 2022.