Semestrální práce

Segmentace obrazu pomocí prahování

Vypracoval : Lískovský David

Předmět : XDEEP

Datum : 05.12.2025

Obsah

Úvod – Předmět práce

Cílem této práce je demonstrovat možnosti využití neuronové sítě U-Net pro automatické rozpoznávání objektů na digitálních obrazech a jejich převod do podoby segmentačních masek. Segmentace představuje úlohu, jejímž cílem je přiřadit každému pixelu obrazu příslušnost buď k relevantnímu objektu, nebo k pozadí.

Z praktického hlediska lze tento proces chápat jako převod vizuální informace do strukturované podoby: model „interpretuje“ vstupní obraz a generuje k němu masku, která odpovídá skutečnému tvaru objektu.

Tato práce se soustředí výhradně na problematiku segmentace, nikoliv na klasifikaci či detekci objektů. Použitý dataset obsahuje dvojice datových vzorků: vstupní obraz a odpovídající binární masku. Úkolem neuronové sítě je naučit se vztah mezi těmito reprezentacemi, tj. rozpoznat a reprodukovat obrys objektu na fotografii.

Pro řešení byla zvolena architektura U-Net, která se v oblasti segmentace osvědčila jako robustní a široce využívaný model. Její předností je schopnost současně zachytit globální strukturu obrazu i jemné detaily hran, a to i při práci s relativně malými datasety.

V rámci experimentů se práce zaměřuje na vliv různých parametrů tréninku na kvalitu segmentace, zejména na velikost datasetu, počet tréninkových epoch, rozlišení vstupních obrazů a další faktory. Hodnocení výsledků probíhá prostřednictvím metriky Intersection over Union (IoU), která představuje standardní nástroj pro měření přesnosti segmentačních masek.

Výstupem práce je systematický přehled toho, jaké reprezentace se model U-Net skutečně naučí, jaká je jeho přesnost při segmentaci a které parametry mají na výsledky nejvýznamnější vliv.

2. Dataset a předzpracování

Dataset použitý v této práci obsahuje přibližně 940 dvojic obrazových dat. Dataset je původně RGB, ale pipeline konvertuje snímky na jednokanálové (grayscale) kvůli jednoduchosti a nižší paměťové náročnosti. Implementace U-Netu přijímá jednokanálový vstup. Masky představují referenční řešení, které neuronová síť využívá během učení jako vzor. Úkolem sítě je naučit se generovat masku, jež co nejvěrněji odpovídá skutečnému tvaru objektu na obraze.

2.1 Charakter dat

Použitá data nejsou syntetická, nýbrž reálná, což je pro segmentační úlohu zásadní. Obrázky zachycují objekty v různých podmínkách, které zahrnují:

- variabilní pozadí,
- odlišné světelné podmínky,
- rozdílné měřítko objektu,
- mírný šum a variace ostrosti.

Tyto faktory činí segmentační úlohu náročnější než v případě akademických datasetů, kde jsou objekty obvykle jednoznačně odděleny od pozadí. Zároveň však ukazují praktický směr práce: cílem je model schopný segmentovat objekty i v podmínkách blízkých reálnému prostředí.

2.2 Předzpracování obrázků

Aby bylo možné data efektivně využít při trénování na grafické kartě a aby síť přijímala jednotný vstup, procházejí všechny snímky i masky stejným předzpracováním.

2.2.1 Změna velikosti

Vstupní snímky i masky jsou přeškálovány na společnou velikost:

- IMG_SIZE = 128,
- případně 256 v některých experimentech.

Menší rozlišení urychluje trénink a snižuje paměťové nároky, avšak může vést ke ztrátě detailů. V experimentální části je patrné, že příliš vysoké rozlišení může naopak výkon zhoršit, pokud model nedisponuje dostatečnou kapacitou.

2.2.2 Normalizace

Obrázky jsou převedeny na hodnoty v rozsahu 0–1 podle vztahu:

$$img = \frac{img}{255}$$

Normalizace přispívá ke stabilizaci tréninkového procesu neuronové sítě.

2.2.3 Binarizace masek

Masky jsou převedeny na binární hodnoty:

$$mask = mask > 0).float()$$

Tím vzniká jednoznačné rozdělení na:

- 1 = objekt,
- 0 = pozadí.

2.2.4 Rozdělení na trénovací a validační sadu

Dataset je rozdělen v poměru:

- 80 % tréninková část,
- 20 % validační část.

Validační sada se nepoužívá pro učení, ale výhradně pro vyhodnocení modelu. Tento postup zajišťuje objektivní měření přesnosti pomocí metriky IoU.

2.3 Částečné využití datasetu v experimentech

Pro analýzu vlivu velikosti datasetu na výsledky je využíván parametr N_SUBSET, který vybírá náhodnou podmnožinu dat. V experimentech jsou použity hodnoty:

- 200 vzorků,
- 400 vzorků,
- 800 vzorků.

Tento postup má dva hlavní cíle:

- umožňuje posoudit, kolik dat je pro U-Net skutečně dostačujících,
- poskytuje vhled do toho, jak tréninkový proces reaguje na změnu velikosti trénovací množiny.

2.4 Další úpravy dat (augmentace)

V základní verzi tréninku byly použity pouze jednoduché transformace, jako horizontální a vertikální převrácení obrazu. Tyto úpravy podporují zobecnění modelu, avšak nepřinášejí výrazné zlepšení výsledků.

Pro budoucí práci doporučuji využít rozsáhlejší sadu augmentací (doporučenou Albumentations), zejména:

- změny kontrastu a jasu,
- přidání náhodného šumu,
- rotační transformace,
- lokální geometrické deformace.

Tyto postupy mohou významně zvýšit robustnost modelu vůči variabilitě reálných dat.

3. Architektura U-Net

U-Net je konvoluční neuronová síť navržená specificky pro úlohy segmentace obrazu. Její hlavní předností je schopnost současně zachytit globální strukturu scény i jemné lokální detaily, které jsou pro přesnou segmentaci zásadní. Název U-Net vychází z charakteristického tvaru architektury, připomínajícího písmeno „U“.

Síť se skládá ze dvou symetrických částí:

- kontraktivní (sestupné),
- expanzivní (vzestupné),

které jsou propojeny tzv. skip-connections. Tyto propojení zajišťují, že síť neztrácí prostorové informace při postupném zmenšování rozlišení.

3.1 Kontraktivní část – extrakce rysů obrazu

První polovina U-Netu postupně redukuje rozlišení vstupního obrazu. Tento proces probíhá střídáním dvou základních operací:

1. Konvoluční vrstvy – extrahují stále komplexnější rysy obrazu, jako jsou hrany, textury či tvary.
2. Max-pooling – snižuje rozlišení na polovinu, čímž umožňuje síti zachytit širší kontext.

Každý blok kontraktivní části obsahuje několik konvolucí, které společně reprezentují informaci o tom, „co“ se na obraze nachází, nikoliv však „kde“ přesně.

3.2 Expanzivní část – lokalizace objektu

Druhá polovina architektury má opačnou funkci: obnovuje rozlišení na původní velikost a určuje přesnou polohu objektu.

Používají se zde:

- transponované konvoluce (up-sampling) – zvětšují mapu rysů,
- konvoluce – zpřesňují lokální detaily.

Výsledkem je výstupní obraz masky se stejným rozlišením, jaké měl původní vstupní snímek.

3.3 Skip-connections – zachování detailů

Klíčovým prvkem architektury jsou skip-connections, tedy přímé propojení mezi vrstvami kontraktivní a expanzivní části.

Při sestupu síť získává abstraktní informace s velkým kontextem, avšak ztrácí jemné detaily. Skip-connections tyto detaily přenášejí přímo do expanzivní části, kde se znovu využívají.

Díky tomu:

- model zachycuje globální tvar objektu,
- současně rozlišuje jemné hranové detaily.

Tento mechanismus je jedním z důvodů, proč U-Net často dosahuje lepších výsledků než jiné segmentační architektury, zejména u menších datových sad.

3.4 Výstupní vrstva

Poslední vrstva U-Netu má obvykle následující podobu:

- 1×1 konvoluce – redukuje počet kanálů na jeden,
- sigmoid aktivace – převádí hodnoty na interval 0–1.

Výstupem je pravděpodobnostní mapa, kde každý pixel udává pravděpodobnost příslušnosti k objektu. Následně je tato mapa převedena na binární masku pomocí prahování (hodnota 0.4).

3.5 Implementace použitá v práci

V implementaci je použit **zjednodušený U-Net** s třemi konvolučními bloky v encoderu (dvě pooling operace) a dvěma kroky up-samplingu, doplněnými skip-connections. Výstupní vrstva vrací jednokanálový logits bez aplikace sigmoidu; sigmoid se používá až při evaluaci. Nejedná se o variantu s předtrénovaným enkodérem. Tento fakt má přímý dopad na dosažené výsledky, což je diskutováno v pozdější části práce. U-Net bez silného enkodéru je vhodný pro experimentální účely, avšak jeho výkon se obvykle saturuje dříve než u moderních variant.

3.6 Výhody a nevýhody zvolené architektury

Výhody

- jednoduchá implementace a stabilní trénování,
- nízké nároky na velikost datasetu,
- vhodnost pro experimentální porovnávání,
- intuitivní interpretace výstupů.

Nevýhody

- omezená kapacita pro složitější textury a variabilní datové sady,
- horší generalizace bez předtrénovaného enkodéru,
- citlivost na rozlišení vstupu – příliš velké obrázky mohou výkon zhoršit (viz experimentální část).

4. Trénovací nastavení a hyperparametry

Pro posouzení chování architektury U-Net v různých podmínkách byly provedeny experimenty s několika typy hyperparametrů. Hyperparametry představují nastavení, která neurčuje síť sama, ale zadává je uživatel. Lze je chápat jako parametry ovlivňující citlivost, rychlost či měřítko v technickém systému.

4.1 Velikost vstupního obrázku (IMG_SIZE)

Pro trénink byly použity dvě varianty rozlišení:

- **128 × 128 px**,
- **256 × 256 px**.

Menší rozlišení usnadňuje trénink a snižuje paměťové nároky, zatímco větší rozlišení umožňuje zachovat jemnější detaily, avšak vyžaduje větší kapacitu modelu nebo rozsáhlejší dataset.

Experimenty ukázaly, že U-Net bez předtrénovaného enkodéru dosahuje lepších výsledků při rozlišení 128 px. Při 256 px již model nedisponuje dostatečnou kapacitou a výkon klesá.

Toto zjištění potvrzuje, že vyšší rozlišení nemusí vždy vést k lepším výsledkům, pokud síť nedokáže dodatečné detaily efektivně využít.

4.2 Velikost batch (BATCH)

Batch určuje počet obrázků zpracovaných sítí v jednom kroku učení. Testované hodnoty byly:

- 4,
- 8,
- 16,
- 32.

Každá velikost má své specifické vlastnosti:

- **Malý batch (4–8)** → větší variabilita gradientu, což může být výhodné u menších datasetů.
- **Velký batch (16–32)** → stabilnější učení, avšak někdy potlačuje jemné detaily.

Výsledky ukázaly, že:

- pro menší dataset (200–400 vzorků) je vhodnější batch 4–8,
- pro větší dataset (800 vzorků) funguje dobře i batch 16–32,
- extrémně velký batch nepřináší zlepšení a může výkon dokonce zhoršit.

4.3 Počet epoch (EPOCHS)

Epochy označují počet průchodů sítí přes celý trénovací dataset. Testované hodnoty byly:

- 25 epoch,
- 50 epoch,
- 100 epoch.

Z výsledků vyplývá:

- **25 epoch** → model vykazuje nedostatečné učení (underfitting),
- **50 epoch** → výrazné zlepšení,
- **100 epoch** → nejlepší výsledky v rámci použitých dat.

U menších datasetů síť obvykle potřebuje více epoch, aby dokázala zobecnit typické hranice objektů.

4.4 Rozdělení datasetu (VAL_SPLIT)

Dataset byl rozdělen v poměru:

- **80 % tréninková část**,
- **20 % validační část**.

Validační sada slouží výhradně k vyhodnocování kvality výsledků prostřednictvím metriky IoU. Síť se z validačních dat neučí, což zajišťuje objektivní hodnocení.

4.5 Funkce ztráty (Loss)

Pro trénink byl použit **BCEWithLogitsLoss** s váhováním kladné třídy pomocí parametru *pos_weight*, tj. variantou BCE pracující přímo s logits. Váhování kompenzuje nevyváženost foreground–background v datech. Váhový koeficient *pos_weight* byl určen z průměrného podílu foreground pixelů v trénovacích maskách.

Omezení BCEWithLogitsLoss: U datasetů s výrazně nevyváženým poměrem (velká oblast pozadí, malý objekt) má tato funkce tendenci upřednostňovat přesnost pozadí na úkor objektu. V praxi to může vést k tomu, že síť nepředpovídá menší objekty dostatečně spolehlivě.

Z tohoto důvodu je v závěru práce doporučena alternativa **Focal Loss**, která penalizuje chyby na objektu více než chyby na pozadí.

4.6 Optimalizátor a learning rate

Pro trénink byl použit **Adam optimizer** s výchozím learning rate = 1e-3.

Learning rate určuje rychlost učení sítě:

- příliš vysoká hodnota vede k nestabilitě,
- příliš nízká k pomalému učení.

V této práci byl learning rate konstantní. Moderní praxe však doporučuje jeho postupné snižování pomocí tzv. **learning rate scheduleru**, který umožňuje:

- v počátečních fázích rychlé hledání řešení,
- v pozdějších fázích jemné dolaďování.

4.7 Ostatní parametry

- **Seed (SEED = 42)** – zajišťuje reprodukovatelnost náhodných operací.
- **Prahování predikce (PRED_THR)** – po aplikaci sigmoid aktivace je maska binarizována prahovou hodnotou, nejčastěji 0.4.

Hodnota 0.4 mírně zvýhodňuje pixely, které síť označuje jako objekt, i když si není zcela jistá. V praxi tato volba vedla k lepším výsledkům IoU než striktní práh 0.5.

5. Evaluace pomocí metriky IoU

Pro vyhodnocení kvality segmentace byla použita metrika Intersection over Union (IoU). IoU představuje standardní způsob měření přesnosti segmentačních masek a je široce využívána v moderních studiích zabývajících se segmentací obrazu. Její hlavní předností je jednoduchost a jednoznačnost: vyjadřuje míru překrytí predikované masky se skutečnou maskou.

5.1 Definice metriky IoU

Pro dva binární obrazce (A) (predikce) a (B) (ground truth) je IoU definováno jako:

$$> IoU = \frac{|A \cap B|}{|A \cup B|} >$$

Tedy poměr plochy průniku ke sjednocení.

- **Průnik** – pixely, které síť označila jako objekt a zároveň jsou objektem ve skutečné masce.
- **Sjednocení** – pixely, které patří do objektu alespoň v jedné z masek.

Hodnota IoU se pohybuje v intervalu:

- **0** = masky se nepřekrývají,
- **1** = masky se shodují dokonale.

5.2 Vhodnost IoU pro segmentaci

IoU zohledňuje plochu objektu a není ovlivněno množstvím pixelů tvořících pozadí. To je zásadní, protože dataset použitý v této práci obsahuje:

- rozsáhlé oblasti pozadí,
- relativně malé objekty.

Pokud bychom použili prostou pixelovou přesnost (accuracy), síť by mohla dosahovat hodnot 95–99 % pouhým předpovězením prázdné masky. IoU naproti tomu penalizuje nedostatečné překrytí velmi výrazně, a proto poskytuje objektivnější hodnocení.

5.3 Výpočet IoU pro jednotlivé obrázky

IoU bylo počítáno přímo z binárních masek. Následující funkce v jazyce Python vypočítává IoU pro jeden obrázek:

$def\ iou\_score(pred, target)$:

"""$Vypočítá\ Intersection\ over\ Union\ (IoU)\ pro\ jeden\ obrázek.$"""

$smooth\ =\ 1e-6$

$pred\ =\ pred.float()$

$target\ =\ target.float()$

$\#\ Průnik\ a\ sjednocení$

$intersection\ =\ (pred\ *\ target).sum()$

$union\ =\ pred.sum()\ +\ target.sum()\ -\ intersection$

$return\ (intersection\ +\ smooth)\ /\ (union\ +\ smooth)$

5.4 Hodnocení modelu na celé validační sadě

Pro porovnání různých experimentů nebylo IoU měřeno pouze na jednom obrázku. Pro každé nastavení tréninku byly vypočteny:

- **průměrná hodnota IoU** na všech validačních vzorcích,
- **směrodatná odchylka IoU**,
- **IoU jednotlivých obrázků**.

Vyhodnocovací funkce použitá v experimentech:

```
def evaluate_unet(model, dl, device, thr = 0.4):
    """Vyhodnotí model na celém DataLoaderu a spočítá průměrné IoU a std."""
    model.eval()
    iou_list = []
    with torch.no_grad():
        for x, y in dl:
            x, y = x.to(device), y.to(device)
            # Predikce (logits)
            logits = model(x)
            prob = torch.sigmoid(logits)
            pred_bin = (prob > thr).float()
            for i in range(x.size(0)):
                iou = iou_score(pred_bin[i, 0], y[i, 0])
                iou_list.append(iou.item())
    iou_mean = np.mean(iou_list)
    iou_std = np.std(iou_list)
    return iou_mean, iou_std
```

IoU je vypočteno pro každý vzorek a následně je určena průměrná hodnota i směrodatná odchylka.

5.5 Binarizace predikce (prahování)

Výstupem U-Netu je mapa pravděpodobností v rozsahu 0–1. Aby bylo možné vypočítat IoU, je nutné převést ji na binární masku. V této práci byl použit práh:

- **0.4**

Tato hodnota byla empiricky zvolena, protože poskytovala lepší výsledky než standardní hodnota 0.5. Nižší práh umožňuje zachytit objekty, u nichž si síť není zcela jistá, a které by jinak mohly být vynechány.

5.6 Interpretace hodnot IoU

V segmentačních úlohách lze hodnoty IoU interpretovat následovně:

- **0.1–0.2** – velmi nízká kvalita, model zachytí pouze základní tvary,
- **0.2–0.4** – přijatelné výsledky pro malé datasety nebo jednoduchý U-Net,
- **0.5+** – kvalitní segmentace,
- **0.7–0.9** – špičkové výsledky dosažené rozsáhlými modely a velkými datasety.

V této práci se hodnoty postupně zlepšovaly s rostoucí velikostí datasetu, počtem epoch a vhodným nastavením hyperparametrů. Detailní výsledky jsou rozpracovány v následující kapitole experimentů.

6. Experimenty a interpretace výsledků

Pro lepší porozumění chování architektury U-Net při různých podmínkách učení byla provedena série experimentů. Každý experiment měnil pouze vybrané parametry, zatímco ostatní zůstávaly konstantní. Výsledky byly vyhodnoceny pomocí průměrné hodnoty metriky IoU a směrodatné odchylky na validační sadě (viz kapitola 5).

Cílem experimentů nebylo pouze porovnat číselné výsledky, ale především pochopit, jaké faktory ovlivňují kvalitu segmentace a proč.

Testovány byly následující oblasti:

1. vliv velikosti trénovacího datasetu,
2. vliv počtu epoch,
3. vliv velikosti batch,
4. vliv rozlišení vstupního obrazu,
5. stabilita tréninku a jeho průběh.

Všechny experimenty byly provedeny na totožné implementaci U-Netu a vyhodnoceny jednotnou metodikou.

6.1 Vliv velikosti datasetu (N_SUBSET)

Byly testovány podmnožiny o velikosti 200, 400 a 800 vzorků.

Výsledky:

Dataset	Epochs	Batch	IMG_SIZE	IoU ± std
200	25	4	128	**0.1447 ± 0.1493**
200	50	8	128	**0.2167 ± 0.1727**
400	50	8	128	**0.1687 ± 0.1267**
800	100	16	128	**0.2440 ± 0.1381**

Interpretace:

- **200 vzorků**: síť zachytí pouze základní tvar objektu, výsledky jsou velmi rozptýlené.
- **400 vzorků**: výkon se mírně stabilizuje, ale hranice zůstávají nepřesné.
- **800 vzorků**: IoU výrazně roste, rozptyl klesá; síť dokáže zobecnit variabilitu tvarů a pozadí.

6.2 Vliv počtu epoch

Testované hodnoty: 25, 50 a 100 epoch.

Pozorované chování:

- **25 epoch**: síť se naučí základní struktury, hranice jsou nepřesné.
- **50 epoch**: výrazné zlepšení, IoU roste o 0.05–0.07.
- **100 epoch**: nejlepší výsledky, síť se stabilizuje a zpřesňuje hrany.

Interpretace: U-Net bez předtrénovaného encoderu potřebuje více epoch, protože začíná „od nuly“ a dataset je malý. 100 epoch je realistické minimum.

6.3 Vliv velikosti batch (4, 8, 16, 32)

Shrnutí:

- **Batch 4–8**: vhodné pro malé datasety, vyšší variabilita gradientu → lepší generalizace.
- **Batch 16**: optimální kompromis pro větší dataset (800 vzorků).
- **Batch 32**: nepřináší zlepšení, někdy výkon zhoršuje.

Velký batch zprůměruje variace tvarů, síť se učí příliš hladce a ztrácí jemné detaily.

6.4 Vliv rozlišení vstupu (128 vs. 256 px)

Výsledky:

IMG_SIZE	N	Batch	Epochs	IoU ± std
128	800	16	100	**0.2440 ± 0.1381**
256	800	32	100	**0.2182 ± 0.1234**

Interpretace:

- Větší rozlišení (256 px) nepřineslo vyšší IoU, naopak výkon klesl.
- Model nemá dostatečnou kapacitu, aby využil dodatečné detaily.
- Up-sampling část nedokáže přesně rekonstruovat složité hrany.

6.5 Stabilita a průběh tréninku

Typická křivka loss:

- rychlý pokles v prvních 5 epochách,
- postupné zlepšování do epochy 30–50,
- mírné kolísání validace (malý dataset),
- stabilizace kolem epochy 70–100.

Výkyvy validační loss nejsou známkou overfittingu, ale variabilitou jednotlivých obrázků.

6.6 Shrnutí experimentů

Co nejvíce pomáhá:

1. větší dataset (800 vs. 200),
2. více epoch (100 vs. 25),
3. správně zvolený batch,
4. nižší IMG_SIZE pro základní U-Net.

Co výkon zhoršuje:

1. příliš velké rozlišení (256 px),
2. příliš velký batch (32) u malé sítě,
3. příliš málo dat (200 vzorků).

Celkové zjištění:

Model se naučil segmentovat základní tvary objektů, avšak jeho kapacita nestačí na detailní rekonstrukci. Hodnota IoU kolem 0.24 je realistická pro nepředtrénovaný U-Net na malém datasetu s reálnými snímky.

7. Diskuse výsledků a možnosti dalšího zlepšení

Výsledky experimentů ukazují, že architektura U-Net je schopna segmentovat objekty i na relativně malém reálném datasetu, avšak její výkon je omezen. Nejlepší dosažená hodnota metriky IoU činí přibližně 0.24, což odpovídá segmentaci zachycující základní tvar objektu, nikoliv však jemné hranové detaily. Tento výsledek je v souladu s podmínkami experimentu: použitím nepředtrénovaného U-Netu, omezeným počtem vzorků a variabilním pozadím.

7.1 Klíčová zjištění

- Síť se naučila rozpoznávat obecný tvar objektu, avšak hranice zůstávají rozmazané.
- Velikost datasetu má zásadní vliv na výkon – nárůst z 200 na 800 vzorků vedl k výraznému zlepšení IoU.
- Vyšší rozlišení (256 px) výkon nezlepšilo, naopak zatížilo model a vedlo ke zhoršení výsledků.
- Stabilizace hranic vyžaduje delší trénink – kolem 70.–100. epochy se výsledky ustalují.

7.2 Praktická interpretace IoU

Hodnota IoU ≈ 0.24 ukazuje, že model má správnou představu o poloze a rozsahu objektu, což může být dostačující pro úlohy předzpracování či lokalizace. Pro přesné technické využití by však bylo nutné model rozšířit. Výsledek odpovídá běžné praxi: nepředtrénované U-Nety na malých datasetech dosahují IoU kolem 0.2–0.3, zatímco vyšší hodnoty (0.5–0.9) vyžadují rozsáhlejší datasety, silnější encodéry a pokročilé metody tréninku.

7.3 Doporučení pro zlepšení

Na základě experimentů i doporučení přednášejícího lze doporučit následující postupy:

1. **Nasazení Focal Loss** – vhodnější než BCE pro malé objekty; zvyšuje IoU a zpřesňuje hranice.
2. **Použití silnějšího encodéru** – přináší lepší extrakci rysů a stabilitu.
3. **Rozšířená augmentace (Albumentations)** – náhodné změny kontrastu, rotace, šum či elastic transformace výrazně zvyšují robustnost.
4. **Learning rate scheduler** – umožňuje jemné dolaďování modelu v pozdějších fázích tréninku.
5. **Ensemble nebo test-time augmentace** – kombinace více modelů zlepšuje výslednou predikci.
6. **Post-processing masek** – morfologické operace či vyhlazení hran mohou zvýšit přesnost i bez změny architektury.

7.4 Plán pro budoucí práci

- Nasadit U-Net s ResNet34 encoderem a Focal Loss (očekávané IoU: 0.45–0.55).
- Přidat bohaté augmentace pomocí Albumentations (+0.05 až +0.15 IoU).
- Rozšířit dataset reálnými nebo augmentovanými vzorky.
- Otestovat modernější architektury (UNet++, DeepLabV3+, SegFormer).
- Optimalizovat hyperparametry pomocí scheduleru a menšího batch.
- Zvážit multi-class nebo multi-label segmentaci při rozšíření úlohy.

7.5 Shrnutí

Architektura U-Net prokázala schopnost segmentovat objekty i na malém datasetu, avšak její výkon je limitován absencí silného encodéru a omezeným množstvím dat. Hodnota IoU ≈ 0.24 je konzistentní s očekáváním a ukazuje na potenciál modelu pro základní lokalizační úlohy. Pro dosažení vyšší přesnosti je vhodné využít pokročilejší ztrátové funkce, předtrénované encodéry, rozsáhlejší augmentace a moderní architektury.