



**MATEMATICKO-FYZIKÁLNÍ  
FAKULTA  
Univerzita Karlova**

**BAKALÁŘSKÁ PRÁCE**

Karel Vlk

**Real-time analýza a validace obrazu z  
webových kamer**

Katedra teoretické informatiky a matematické logiky

Vedoucí bakalářské práce: RNDr. Kateřina Macková

Studijní program: Informatika

Praha 2024

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne .....  
Podpis autora

Chtěl bych poděkovat své vedoucí RNDr. Kateřině Mackové za vedení mé bakalářské práce, za cenné rady a připomínky, které mi poskytla. Velké díky patří všem, kteří se podíleli na testování použitelnosti naší aplikace. A samozřejmě zvláštní poděkování mé rodině a přítelkyni za to, že mi stáli po boku během těžkých a napjatých chvil.

Název práce: Real-time analýza a validace obrazu z webových kamer

Autor: Karel Vlk

Katedra: Katedra teoretické informatiky a matematické logiky

Vedoucí bakalářské práce: RNDr. Kateřina Macková, Katedra teoretické informatiky a matematické logiky

**Abstrakt:** V dnešní době digitální transformace nabývá na důležitosti využití webových kamer jako zdrojů informací o venkovních podmínkách a to především díky jejich široké dostupnosti a potenciálu v oblasti meteorologie. Práce se zaměřuje na analýzu a validaci snímků z webových kamer v reálném čase, která je zásadní pro automatizované poskytování informací o aktuálních meteorologických jevech uživatelům po celém světě. Hlavním cílem práce je dosáhnout efektivní analýzy a validace snímků poskytující informace s maximální přesností a minimálním zpožděním. V rámci práce je představen úvod do dané problematiky a přehled stávajících metod a technologií. Hlavní části práce se soustředí na výběr metod a přístupů pro zpracování dat, které jsou zásadní pro návrh efektivního systému. Implementovaný systém, založený na vybraných sofistikovaných technikách strojového učení, je navržen tak, aby vynikal ve zpracování dat v reálném čase a klade důraz na dosažení nejvyšší účinnosti a efektivity.

**Klíčová slova:** analýza obrázků, detekce objektů, klasifikace obrazu, klasifikace počasí, počítačové vidění, hluboké učení, konvoluční sítě

Title: Analysis and validation of webcam images in real time

Author: Karel Vlk

Department: Department of Theoretical Computer Science and Mathematical Logic

Supervisor: RNDr. Kateřina Macková, Department of Theoretical Computer Science and Mathematical Logic

Abstract: Nowadays, digital transformation emphasizes the use of webcams as sources of information about outdoor conditions, primarily due to their wide availability and potential in meteorology. The work focuses on the analysis and validation of real-time webcam images, which is crucial for the automated provision of information about current meteorological events to users worldwide. The main goal of the work is to achieve efficient analysis and validation of images, providing information with maximum accuracy and minimal delay. Within the scope of the work, an introduction to the issue and an overview of existing methods and technologies are presented. The main parts of the work concentrate on selecting methods and approaches for data processing, which are essential for designing an efficient system. The implemented system, based on selected sophisticated machine learning techniques, is designed to excel in real-time data processing and emphasizes achieving maximum efficiency and effectiveness.

Keywords: image processing, object detection, image classification, weather classification, computer vision, deep learning, convolutional neural network

# Obsah

<b>1</b>	<b>Úvod</b>	<b>9</b>
1.1	Cíle práce . . . . .	9
<b>2</b>	<b>Souhrnná analýza a identifikace problémových okruhů</b>	<b>11</b>
2.1	Kontrola obrazu . . . . .	11
2.2	Validace obsahu obrazu . . . . .	13
2.3	Klasifikace obrazu . . . . .	13
2.3.1	Klasifikace počasí . . . . .	14
2.3.2	Klasifikace scenérie . . . . .	14
2.4	Detekce směru kamery podle slunce . . . . .	15
2.5	Shrnutí . . . . .	15
<b>3</b>	<b>Metody řešení identifikovaných problémových okruhů</b>	<b>16</b>
3.1	Metody kontroly obrazu . . . . .	16
3.1.1	Metody detekce snímků bez relevantního obsahu . . . . .	16
3.1.2	Metody detekce částečně stažených snímků . . . . .	17
3.1.3	Metody detekce vertikálně poškozených snímků . . . . .	18
3.2	Metody detekce objektů . . . . .	19
3.2.1	Dvoufázové modely . . . . .	20
3.2.2	Jednofázové modely . . . . .	24
3.2.3	Komparace výkonu modelů . . . . .	28
3.3	Metody klasifikace obrazu . . . . .	31
3.3.1	Hluboké učení v klasifikaci obrazu . . . . .	31
3.3.2	Statistické metody strojového učení v klasifikaci obrazu . . . . .	35
3.4	Shrnutí . . . . .	37
<b>4</b>	<b>Datasetsy</b>	<b>38</b>
4.1	Vlastní datasetsy . . . . .	38
4.1.1	Dataset pro klasifikaci scenerie . . . . .	38
4.1.2	Dataset pro klasifikaci počasí . . . . .	39
4.1.3	Dataset pro detekci oblohy . . . . .	40
4.1.4	Dataset pro detekci slunce . . . . .	40
4.2	Cizí datasetsy . . . . .	41
4.2.1	MSCOCO 2017 dataset . . . . .	41
4.2.2	COCO-Text . . . . .	41
4.2.3	SUN 2012 dataset . . . . .	42
4.2.4	Dataset pro detekci obličejů . . . . .	42
4.2.5	Dataset pro detekci státních poznávacích značek vozidel . . . . .	42
4.2.6	Dataset pro detekci nahoty . . . . .	43
4.3	Syntetické datasetsy . . . . .	43
4.3.1	DataDreamer . . . . .	43
4.3.2	Dataset pro detekci snímků bez relevantního obsahu . . . . .	45
4.3.3	Dataset pro detekci částečně stažených snímků . . . . .	45
4.3.4	Dataset pro detekci vertikálně poškozených snímků . . . . .	45
4.4	Shrnutí . . . . .	46

<b>5 Experiments</b>	<b>47</b>
5.1 Experiments s CV validátory pro získání potřebných prahů . . . . .	47
5.1.1 Experiments pro určení prahových hodnot pro algoritmus detekce fotek bez relevantního obsahu . . . . .	47
5.1.2 Experiments pro určení prahových hodnot pro algoritmus detekce částečně stažených fotek . . . . .	48
5.1.3 Experiments pro nalezení nejlepšího klasifikátoru do algoritmu na detekci vertikálně poškozených obrázků . . . . .	48
5.2 Experiments s YOLOv8s na optimální detekci objektů . . . . .	50
5.2.1 Výběr a použití datasetů . . . . .	50
5.2.2 Metodologie tréninku . . . . .	51
5.2.3 Metodologie evaluace . . . . .	52
5.2.4 Průběh tréninku . . . . .	52
5.2.5 Analýza a evaluace finetuningu YOLOv8s modelů . . . . .	52
5.2.6 Analýza výsledků experimentů . . . . .	56
5.2.7 Shrnutí . . . . .	57
5.3 Experiments na klasifikaci scenérie . . . . .	57
5.3.1 Metodologie . . . . .	57
5.3.2 Vývoj a přístup ke klasifikaci scenerie . . . . .	58
5.3.3 Evaluace modelů . . . . .	60
5.4 Experiments na klasifikaci počasí . . . . .	61
5.4.1 Vývoj a přístupy . . . . .	61
5.4.2 Evaluace přístupů . . . . .	62
5.5 Celkové shrnutí experimentů . . . . .	62
<b>6 Systém, implementace a využití</b>	<b>64</b>
6.1 Architektura . . . . .	64
6.1.1 Controller . . . . .	64
6.1.2 Předzpracování CV validátory . . . . .	66
6.1.3 Detekce objektů a klasifikace . . . . .	66
6.1.4 Validace na základě objektů . . . . .	66
6.1.5 Klasifikace scenérie . . . . .	66
6.1.6 Agregace výsledků . . . . .	66
6.2 Použité technologie . . . . .	67
6.3 Analýza časů zpracování . . . . .	68
6.3.1 Použité hardwarové konfigurace . . . . .	68
6.3.2 Výsledky měření . . . . .	69
6.4 Modely v prototypu systému . . . . .	69
6.5 Diskuse o implementaci . . . . .	70
6.6 Využití a adaptabilita systému . . . . .	70
6.6.1 Zemědělství a monitorování vegetace . . . . .	70
6.6.2 Real-time monitorování meteorologických situací . . . . .	70
6.6.3 Monitorování dopravní situace . . . . .	70
6.6.4 Shrnutí využití . . . . .	71
6.7 Shrnutí . . . . .	71
<b>Závěr</b>	<b>72</b>
<b>Literatura</b>	<b>74</b>

<b>Seznam obrázků</b>	<b>77</b>
<b>Seznam tabulek</b>	<b>78</b>
<b>A Přílohy</b>	<b>79</b>
A.1 Přehled elektronických příloh . . . . .	79

# 1 Úvod

V posledních desetiletích došlo k technologickému pokroku, který výrazně změnil způsob, jakým interagujeme se světem kolem nás. Velký vliv na to měl především nárůst dostupnosti a využití webových kamer. Tato zařízení se stala všudypřítomným nástrojem, jenž nachází uplatnění v široké škále aplikací od osobní komunikace přes bezpečnostní systémy až po monitorování dopravy a meteorologických jevů. Tyto aplikace mají vysoké požadavky na okamžitý přístup k informacím a neustálou potřebou aktualizovaných dat, které umožňují rychle a efektivně reagovat na měnící se podmínky v reálném čase (real-time<sup>1</sup>).

Schopnost real-time zpracování a interpretace vizuálních dat je zdrojem cenných informací pro další aplikace, a také umožňuje rychlou reakci na nově vznikající situace. U monitorování počasí umožňuje tato analýza okamžitě identifikovat změny v meteorologických podmínkách. U bezpečnostních systémů zase může zabránit potenciálním hrozbám. V obou případech je zřejmé, že spolehlivost a rychlosť zpracování dat jsou nezbytné.

S narůstající závislostí na těchto technologiích přichází také výzva: *Jak zajistit, že analýza a validace obrazu jsou prováděny s potřebnou přesností a v reálném čase?* Zde nachází uplatnění pokročilé techniky strojového učení a počítačového vidění. Konvoluční neuronové sítě a další metody hlubokého učení nabízejí nástroje pro extrakci důležitých informací dat a to s přesností a rychlosťí, které byly dotedl nedosažitelné. Tyto technologie umožňují nejen identifikovat a klasifikovat objekty v obrazových datech, ale také provádět složitější úkoly, jako je detekce počasí nebo analýza dopravních situací - vše v reálném čase.

Tato bakalářská práce se zaměřuje na využití pokročilých technik pro real-time analýzu a validaci obrazu z webových kamer se změřením na ty, které informují uživatele o aktuálních meteorologických podmínkách v jimž vybraných lokalitách. Věnuje se analýze pokročilých metod strojového učení a jejich využití pro automatizaci získávání informací. Cílem této práce je vyvinout systém, který bude schopen efektivně a přesně analyzovat snímky v reálném čase a identifikovat a validovat v nich obsažené klíčové informace. Tím přispěje k rozvoji technologií, které umožní lidem lépe porozumět a reagovat na svět kolem nich.

## 1.1 Cíle práce

Hlavním cílem této práce je provést experimenty zaměřené na real-time analýzu a validaci obrazu z webových kamer se zvláštním důrazem na monitorování meteorologických podmínek a volba nejlepších metod do implementace prototypu systému. Experimenty zahrnují rozsáhlý průzkum a analýzu existujících metod strojového učení a počítačového vidění, kde je věnována speciální pozornost konvolučním neuronovým sítím a jejich schopnosti efektivně zpracovávat data v reálném čase.

Na základě teoretických poznatků následuje fáze experimentálního vývoje, během níž jsou vybrané metody optimalizovány a adaptovány na specifické potřeby

---

<sup>1</sup>Real-time znamená, že systém nebo proces je schopen reagovat na události okamžitě, nebo v přísně definovaném časovém rámci.

analýzy a validace obrazu. Cílem experimentů je identifikovat konfigurace, které nabízí nejlepší rovnováhu mezi přesností a rychlostí, aby bylo možné dosáhnout vysoké efektivity zpracování v minimálním čase.

Součástí práce je také vývoj demonstrativního prototypu systému, který slouží jako praktická platforma pro ověření a demonstraci výsledků experimentálního vývoje. Prototyp umožňuje v reálném čase rychlou analýzu, detekci nežádoucího obsahu a klasifikaci obrazu, čímž zajistí poskytování přesných a okamžitě aktualizovaných informací o počasí a scenerii.

Dalším krokem je hodnocení a validace vyvinutého prototypu. Toto hodnocení je provedeno prostřednictvím komplexního testování na různorodých reálných a syntetických datasetech. Cílem je ověřit účinnost, přesnost a spolehlivost systému. Analýza výsledků testování slouží jako základ pro diskusi o potenciálních vylepšeních a rozšířeních systému.

V neposlední řadě se práce zaměřuje na diskusi možností dalšího výzkumu a aplikace vyvinutých metod v praxi. Je zkoumáno, jak rozšířit funkčnost systému pro další oblasti použití a jak přispět k praktickému využití těchto technologií ve prospěch širší společnosti.

Tato práce má za cíl propojit teoretický výzkum a jeho praktickou aplikaci, čímž přináší přínos pro společnost. Záměrem je ukázat, jak lze teoretické metody a experimentální vývoj efektivně uplatnit ve skutečném světě, a tak významně přispět k lepšímu porozumění a interakci s naším okolím.

## 2 Souhrnná analýza a identifikace problémových okruhů

V této kapitole se zaměříme na představení souboru technických výzev a problémů, které je třeba vyřešit pro efektivní a účinné zpracování obrazu z webových kamer. Klíčem k úspěchu je identifikace a implementace robustních řešení pro kontrolu, validaci, klasifikaci a další zpracování obrazových dat. Představujeme zde řadu podproblémů, jejichž řešení je pro tento účel nezbytné.

Prvním z nich je kontrola obrazu, kde musíme zajistit, že obrazová data jsou vhodná pro další zpracování. To zahrnuje detekci poškození obrazu, jako je vertikální zkreslení nebo pouze částečná staženost, a identifikaci neplatných snímků, například z vypnutých kamer bez relevantního obsahu.

V druhém kroku ověřujeme obsah obrazu s cílem zjistit, zda nezahrnuje nevhodné prvky jako přítomnost nahoty, lidí a jejich obličejů, státních poznávacích značek vozidel nebo vozidel samotných. Zároveň se zaměřujeme na identifikaci prvků v obrazu, které ukazují na jeho relevanci a užitnou hodnotu. V našem případě je to detekce oblohy, která značí, že se jedná o venkovní scénu vhodnou pro analýzu počasí. Tento proces je klíčový pro zajištění, že obraz skutečně obsahuje informace relevantní pro naše potřeby a neobsahuje materiál, který by mohl být nežádoucí či irrelevantní.

Třetím krokem je klasifikace obrazu. Klasifikace obrazu je proces, ve kterém přiřazujeme obraz do relevantní skupiny na základě meteorologických podmínek nebo typu scenerie v něm obsažené.

Posledním krokem je detekce směru kamery, která umožňuje pochopit orientaci kamery vůči zaznamenané scéně a je klíčová pro interpretaci zaznamenaných dat, zejména při aplikacích monitorování počasí a venkovních podmínek.

Po agregaci výsledků z jednotlivých kroků procesu získáváme komplexní pohled na analyzovanou scénu spolu s informací, zda snímek prošel validací a je možné s ním případně dál pracovat.

V následujících částech kapitoly se detailněji věnujeme jednotlivým výše uvedeným krokům. Analyzujeme dostupné stávající metody a technologie pro jejich řešení a navrhujeme způsoby, jak tyto metody adaptovat a integrovat do našeho systému pro komplexní analýzu a validaci obrazu z webových kamer.

### 2.1 Kontrola obrazu

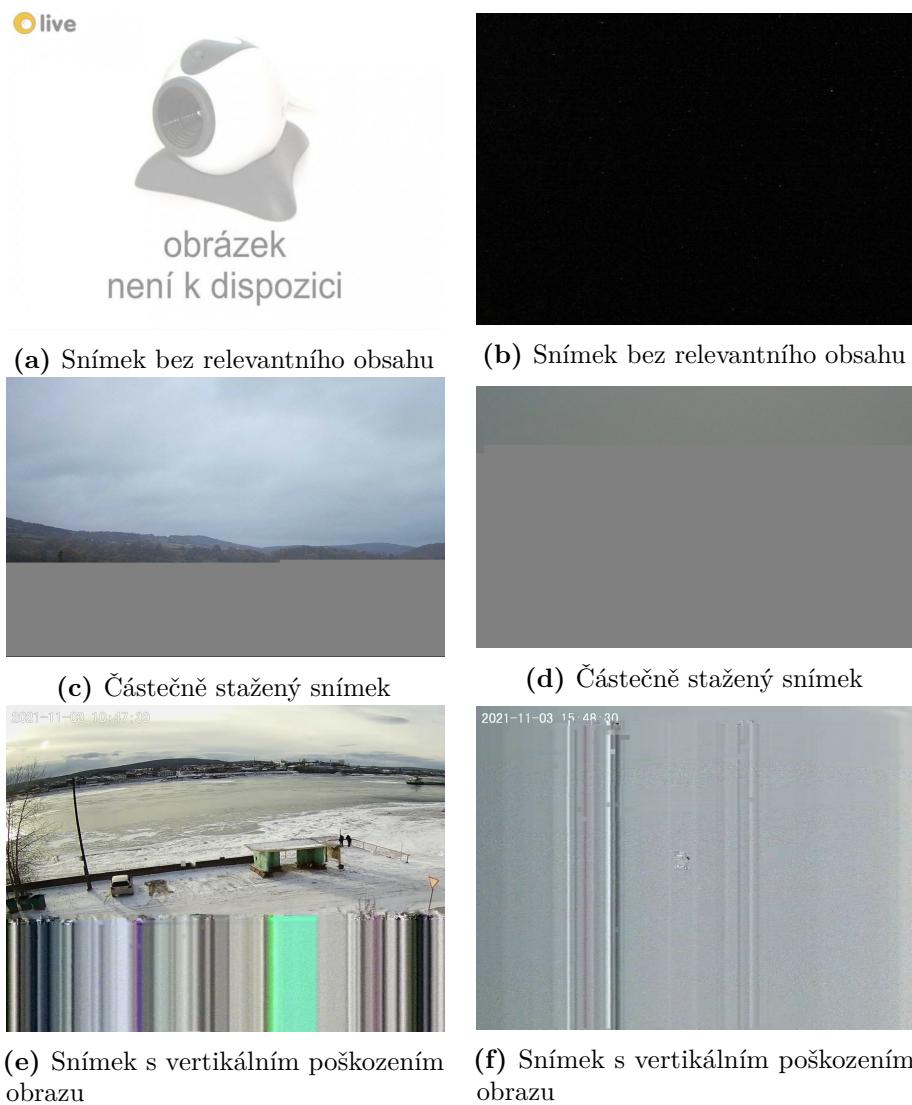
Prvním krokem je kontrola obrazu, která má za úkol identifikovat a vyřadit snímkы, jež nejsou dostatečně kvalitní, a proto nejsou vhodně pro další analýzu. Problémy s kvalitou obrazu mohou vzniknout jak z technických, tak z enviromentálních omezení a mohou zahrnovat snímkы bez významného obsahu, částečně stažené obrazy nebo obrazy vertikálně poškozené.

Prvním problémem jsou snímkы, které neobsahují žádný užitečný obsah. Například pokud poskytovatel vypne webovou kameru a na jejím místě se zobrazí na jednobarevném pozadí text „obrázek není k dispozici“ (viz Obrázek 2.1a) nebo pokud je snímek zcela jednobarevný bez jakýchkoliv rozpoznatelných objektů (viz Obrázek 2.1b). Tento jev lze identifikovat pomocí analýzy hodnot pixelů a

ověření, zda převažující barva nepřekračuje určený práh, což by indikovalo absenci relevantního obsahu.

Další problém představují částečně stažené obrazy, kdy z různých důvodů nedojde ke kompletnímu přenosu dat a část obrazu chybí (viz Obrázek 2.1c a Obrázek 2.1d). Identifikace těchto případů se obvykle provádí analýzou gradientů, která pomáhá odhalit nečekané změny v obrazových datech naznačující jejich nedokončený charakter.

Poslední problém je vertikální poškození obrazu, které se specificky projevuje jako deformace nebo přerušení v pravidelnosti obrazových řádků a může být způsobeno chybami při přenosu dat nebo poruchou samotné kamery (viz Obrázek 2.1e a Obrázek 2.1f).



**Tabulka 2.1** Příklady snímků, které neprojdou úvodním krokem kontroly

Řešení těchto problémů je zásadní pro zajištění kvality a relevace obrazových dat před jejich dalším zpracováním, protože pomáhá optimalizovat výkon, aby se snímky, které neprojdou úvodní kontrolou, dále nezpracovávaly. Tak je zvýšena celková spolehlivost a přesnost systému.

## 2.2 Validace obsahu obrazu

Pro účely validace obsahu obrazu je zásadní využití techniky detekce objektů, která umožňuje specifické vyhledávání a rozpoznávání objektů v obrazových datech. Tato metoda je důležitá pro ověření přítomnosti konkrétních prvků nebo nežádoucího obsahu následujících kategorií:

### Kategorie potenciálně nevhodného obsahu

- Nahota
- Osoby zabírající významnou část obrazu
- Viditelné obličeje
- Dopravní prostředky zabírající významnou část obrazu
- Viditelné státní poznávací značky vozidel: soukromí a bezpečnostní rizika
- Texty, loga a reklamy: nežádoucí z právních nebo etických důvodů

V obrázcích je také nezbytné identifikovat kategorie, které obsahují informaci o počasí a bez nichž nemá další zpracování snímku význam.

### Nezbytný prvek pro validaci

- Obloha: základní prvek pro analýzu a klasifikaci meteorologických podmínek.

Jednou z hlavních výzev je rozmanitost a specifickost hledaných objektů. Objekty jsou často velmi odlišné a nekompatibilní mezi sebou, což ztěžuje nalezení jediného datasetu, který by obsahoval všechny relevantní kategorie. Využití separátních volně dostupných datasetů pro jednotlivé kategorie objektů je proto nejpraktičtějším přístupem. Tato strategie umožňuje specializovanou a přesnou detekci pro každou kategorii zvlášt, což zvyšuje celkovou účinnost a spolehlivost procesu validace.

Alternativním řešením je pak vytvoření jednoho rozsáhlého syntetického datasetu, který by integroval všechny požadované kategorie. Tento přístup by umožnil generování a anotaci obrazových dat na míru specifickým potřebám našeho systému zahrnující širokou škálu scénářů a relevantních objektů pro proces validace.

## 2.3 Klasifikace obrazu

Klasifikaci obrazu můžeme rozdělit do dvou kategorií: *Klasifikace meteorologických podmínek (počasí)* a *klasifikace typu scenerie*. Tato detekce poskytuje ucelený pohled na analyzovanou scénu.

### **2.3.1 Klasifikace počasí**

Klasifikace meteorologických podmínek vyžaduje definici specifických kategorií, které zahrnují:

- Jasno (obloha bez mraků)
- Slunečno
- Oblačno
- Deštivo
- Mlhavo
- Sněživo

Definování počasí je složitý úkol, zejména když se pokoušíme o jeho rozpoznání z kamerových snímků, které často zobrazují různorodé scenerie. S cílem efektivně řešit tento problém jsme se rozhodli přistoupit k vytvoření vlastního datasetu, který je navržen s ohledem na potřeby klasifikace počasí a je tvořen snímky, které jsou typologicky podobné těm, jež poskytují venkovní webové kamery.

### **2.3.2 Klasifikace scenérie**

Klasifikace scenérie umožňuje kategorizovat každý obraz do více skupin současně. Definujeme následující klasifikační skupiny:

- Letiště
- Pláž
- Budova
- Město
- Pobřeží
- Les
- Interiér (vnitřní prostory)
- Jezero
- Krajina
- Meteo (scény s dominující oblohou umožňující posouzení aktuálního počasí)
- Hora
- Přístav
- Řeka
- Sportovní areál

- Náměstí
- Doprava
- Vesnice

Vzhledem k absenci existujícího datasetu, který by umožňoval přiřazení více tříd k jednomu obrazu a pokrýval by námi definované kategorie, je nezbytné vytvoření vlastního datasetu pro tento účel obdobně jako u klasifikace počasí.

## 2.4 Detekce směru kamery podle slunce

Posledním krokem analýzy je určení směru natočení kamery. Hlavní roli zde hraje detekce slunce na obrazu a stanovení jeho pozice. Potenciálními problémy jsou odrazy slunečního světla na vodních hladinách nebo jiných lesklých plochách. Další komplikací je přítomnost jiných intenzivních světelných zdrojů jako pouliční lampy nebo světlomety vozidel. Zmíněné světelné zdroje mohou vést k nesprávným detekcím působící jako falešně pozitivní nálezy způsobené tím, že sluneční záře je jen jeden z mnoha souborů ostrých bodů světla na obrazu. Hlavními rozlišovacími faktory jsou, že slunce na obloze může být jen jedno, je typicky kulatého tvaru a má odstín směřující do žluta, což pomáhá v jeho identifikaci mezi ostatními světelnými zdroji. Tyto rozlišovací faktory spolu s nutnou podmínkou polohy na obloze tvoří předpoklady pro úspěšnou identifikaci a omezují možnost chybné detekce. Následně můžeme určit směr natočení kamery. Tento krok je závislý na lokalizaci a čase, kdy a kde byl snímek pořízen, což jsou informace obvykle dostupné v EXIF<sup>1</sup> metadatech snímku.

## 2.5 Shrnutí

V této kapitole jsme prováděli komplexní analýzu a identifikaci klíčových problémových okruhů spojených s validací a klasifikací obrazových dat z webových kamer. Pro validaci obsahu obrazu jsme došli k tomu, že nejlepším přístupem je využití technik detekce objektů k definování a rozpoznávání nežádoucích objektů. Toto zahrnuje specifikaci objektů, které by na snímku neměly být přítomny (například nahota nebo obličeje). S tím je spojena také identifikace prvků, které jsou pro relevantnost nezbytné - obloha. Dále jsme se zaměřili na klasifikaci obrazů do kategorií zahrnujících meteorologické podmínky jako jsou například jasno, slunečno nebo oblačno a typy scenerie jako jsou například letiště, města či přírodní krajiny, což nám umožňuje ucelenější pochopení obsahu snímků. Posledním krokem byla detekce směru kamery, kde hlavní roli hraje určení pozice slunce na obrazu, což je zásadní pro správné orientování.

---

<sup>1</sup>EXIF je specifikace formátu metadat určená pro soubory obrázků.

# 3 Metody řešení identifikovaných problémových okruhů

V této kapitole poskytujeme teoretický základ a kontext metod potřebných pro náš systém. Poskytneme detailní pohled na jednotlivé problematiky uvedené v předchozí kapitole včetně návrhů řešení. Veškeré navržené přístupy budou následně podrobeny detailnímu testovaní a evaluaci v části o experimentech. Součástí kapitoly je také přehled souvisejících prací, které se týkají podobných témat.

## 3.1 Metody kontroly obrazu

V této části se věnujeme problematice kontroly kvality obrazových dat, přičemž se zaměřujeme na tři oblasti: *analýzu obrazů bez relevantního obsahu*, *detekci časťecně načtených snímků* a *detekci vertikálních poškození obrazu*. Jedním z aspektů při řešení těchto problémů je volba mezi tradičními metodami počítačového vidění (CV) a modernějšími přístupy založenými na hlubokém učení. Ačkoliv hluboké učení, zejména konvoluční neuronové sítě (CNN), dosahuje významných úspěchů v mnoha aplikacích, není vždy ideálním řešením pro všechny scénáře. Jednou z hlavních výhod CV je schopnost efektivního zpracování obrazu s minimálními nároky na množství trénovacích dat. Tyto přístupy jsou založeny na pevně definovaných pravidlech a algoritmech, které lze efektivně implementovat a optimalizovat pro specifické úlohy zpracování obrazu. Vzhledem k těmto faktorům se tradiční metody počítačového vidění jeví jako vhodná volba pro zpracování a analýzu obrazových dat v kontextech, kde jsou hlavními omezeními dostupnost dat a kde je možné problém a kontext jeho výskytu přesně definovat, což je případ kontroly obrazu.

### 3.1.1 Metody detekce snímků bez relevantního obsahu

V této části probíráme detekci snímků, které neobsahují užitečný obsah. Takové situace nastávají, když je na místě očekávaného obrazu prezentován pouze statický obraz bez rozpoznatelných objektů nebo když snímek obsahuje jednolitou barvu bez jakýchkoli detailů.

#### **VoidSeeker: Algoritmus pro detekci snímků bez relevantního obsahu**

Cílem algoritmu VoidSeeker je analyzovat obrazová data a určit, zda snímek obsahuje relevantní vizuální informace. Navrhujeme proces, který se skládá z několika kroků zaměřených na kvantifikaci obsahu obrazu prostřednictvím statistické analýzy a detekce hran. Tímto způsobem je možné rozlišit mezi snímkami s užitečným obsahem a snímkami, které jsou pro další zpracování nevhodné.

#### **Kroky námi navrhovaného algoritmu:**

1. Stanovení prahových hodnot pro standardní odchylku intenzity pixelů a procentuální zastoupení hran v obrazu. Tyto prahy slouží k rozlišení mezi snímky s dostatečným obsahem a těmi bez něj na základě rozdílů v rozložení intenzit a výskytu hran.
2. Načtení obrazových dat a převedení na šedotónový obraz, což umožňuje jednoduší analýzu intenzit pixelů bez nutnosti zohledňovat barvu. Šedotónová reprezentace zjednodušuje výpočetní procesy.
3. Na šedotónovém obrazu se spočítá standardní odchylka intenzit pixelů, která poskytuje informaci o variabilitě intenzity v obrazu. Nízká standardní odchylka naznačuje malou variabilitu typickou pro jednobarevné snímky.
4. S využitím algoritmu Canny [1] se v obrazu detekují hrany. Poměr počtu pixelů, které představují hrany vůči celkovému počtu pixelů v obrazu poskytuje měřítko pro určení, zda snímek obsahuje rozpoznatelné objekty.
5. Pokud standardní odchylka a procentuální zastoupení hran nepřekračují stanovené prahové hodnoty, snímek je označen jako nevhodný pro další zpracování.

### 3.1.2 Metody detekce částečně stažených snímků

Tato část popisuje problematiku, kdy data nejsou kompletně stažena nebo jsou během přenosu poškozena, kvůli čemuž může část obrazu chybět nebo být narušena.

#### FragGuard: Algoritmus detekce částečně stažených snímků

Navrhujeme algoritmus FragGuard založený na metodách zpracování obrazu: *Gaussovo rozostření* a *analýza gradientu*. Tento přístup je zaměřen na identifikaci oblastí v obrázcích, které prokazují významné změny v intenzitě, což naznačuje nekompletnost snímků.

#### Kroky námi navrhovaného algoritmu:

1. Načtení obrazových dat a jejich převod do šedotónového formátu.
2. Zpracování šedotónového obrazu Gaussovým filtrem s jádrem o velikosti  $5 \times 5$ , což umožňuje efektivní snížení šumu a zvýšení spolehlivosti detekce významných změn v obrazu.
3. Aplikace operace výpočtu gradientu na rozostřený obraz, které slouží k detekci změn intenzity v obrazu podél jeho vertikální osy. Ty mohou indikovat přítomnost oblastí, jež nejsou kompletní.
4. Identifikace změn v gradientu, které výrazně přesahují zvolené prahové hodnoty. Prahové hodnoty jsou určeny dynamicky na základě distribuce rozdílů gradientů, což poskytuje flexibilní způsob detekce signifikantních přechodů mezi různými částmi obrazu.

5. Analýza posledního významného gradientu a jeho indexu, který je považován za potenciální hranici mezi kompletní a nekompletní částí obrazu. Na základě tohoto kroku se provádí výpočet standardní odchylky intenzit pixelů pod touto hranicí.
6. Výpočet a normalizace standardní odchylky pixelových intenzit pod hranicí identifikovanou v předchozím kroku vzhledem k celkové standardní odchylce obrazu. Porovnání této normalizované hodnoty s předem stanoveným prahem umožnuje určit, zda je obraz částečně stažen. Obraz je považován za částečně stažený, pokud je hodnota normalizované standardní odchylky větší nebo rovna prahové hodnotě.

### **3.1.3 Metody detekce vertikálně poškozených snímků**

V této části se zaměřujeme vertikální poškození obrazu, které může nastat například při chybném přenosu dat nebo problému s dekódováním vedoucím k výskytu nekonzistentních vertikálních oblastí v obrazu. Tyto oblasti mohou být charakterizovány výraznými změnami v intenzitě, textuře nebo barvě, které narušují vizuální integritu obrazu.

#### **VertiScan: Algoritmus pro detekci vertikálně poškozených obrázků**

VertiScan algoritmus zaměřený na identifikaci vertikálně poškozených snímků navrhujeme tak, že implementuje řadu kroků pro analýzu obrazových dat a určení přítomnosti poškození. Důležitým aspektem tohoto procesu je efektivní detekce a kvantifikace změn v obrazových datech, které by mohly naznačovat vertikální poškození.

#### **Kroky námi navrhovaného algoritmu:**

1. Načtení obrazu z obrazových dat a konverze do šedotónové reprezentace, což umožňuje efektivnější analýzu intenzity pixelů.
2. Výpočet standardní odchylky intenzit pixelů ve vertikálním směru pro identifikaci oblastí s vysokou variabilitou intenzity.
3. Použití diference standardních odchylek k lokalizaci největšího poklesu, což naznačuje potenciálně poškozenou oblast.
4. Pokud byla identifikována poškozená oblast, vypočítá se průměrná absolutní diference intenzit pixelů v této oblasti pro kvantifikaci rozsahu poškození.
5. Sumarizace charakteristik standardní odchylky (průměr, standardní odchylka, šíkmost, špičatost) a lokalizace a rozsah poškozené oblasti pro vytvoření vstupního vektoru pro klasifikátor.
6. Predikce klasifikátorem, zda je obraz vertikálně poškozen (1) nebo ne (0) na základě výše uvedených charakteristik.



Obrázek 3.1 Výstup detekce objektů modelu YOLOv8 od Ultralytics Jocher et al. [2]

## 3.2 Metody detekce objektů

Detekce objektů je technologie v oblasti počítačového vidění umožňující odhadovat polohu objektů v obrazu nebo videu. Tento proces zahrnuje také určení jeho přesné lokality a klasifikaci do odpovídající kategorie. Pro usnadnění vizuální identifikace objektů se často využívá ohraničující rámeček umístěný kolem detekovaného objektu (viz Obrázek 3.1).

Historicky byly v oblasti detekce objektů využívány různé metody, jako jsou například Viola–Jonesův detekční rámec založený na Haar features Viola a Jones [3], Scale-invariant feature transform (SIFT) Lowe [4] a Histogram of oriented gradients (HOG) features Dalal a Triggs [5], které umožňovaly efektivní detekci objektů před érou hlubokého učení. Tyto metody hrály svou roli ve vývoji technik počítačového vidění, avšak s příchodem hlubokého učení a zejména s využitím konvolučních neuronových sítí došlo k výraznému posunu ve schopnosti systémů detekovat a klasifikovat objekty v obrazových datech s vysokou přesností a rychlostí.

Detekce se zaměřuje na rozpoznávání širokého spektra objektů ve vizuálních datech, přičemž nejčastějšími příklady jsou psi, kočky, lidé a auta. Skutečný rozsah rozpoznávaného spektra je omezen možnostmi trénování modelu, takže je možné rozpoznat prakticky jakýkoli objekt.

Detekční modely můžeme rozdělit do dvou hlavních kategorií: *dvoufázové* a *jednofázové detektory*. Každá z těchto kategorií má specifické charakteristiky, architekturu a využití, které je definují a odlišují.

Průlomem v oblasti detekování objektů byl model R-CNN (Regions with Convolutional Neural Networks). Tento dvoufázový (two-stage) přístup kombinoval regionální návrhy s konvolučními neuronovými sítěmi pro přesnou detekci a klasifikaci objektů [6]. Postupem času se však objevila potřeba rychlejších a efektivnějších modelů, což vedlo k rozvoji jednofázových (jednostupňových) detektorů jako YOLO (You Only Look Once) Redmon et al. [7] a SSD (Single Shot MultiBox Detector) Liu et al. [8], které dokázaly detekovat objekty v reálném čase s vyšší přesností. Jednofázové modely jsou schopny dosahovat rychlejšího zpracování obrazu díky přímé klasifikaci a lokalizaci objektů bez nutnosti předběžného výběru regionů.

YOLO9000 Redmon a Farhadi [9] jakožto součást rodiny modelů YOLO představuje průlom v této oblasti díky své schopnosti detekovat rozsáhlé spektrum objektů do více než 9000 kategorií, což umožňuje značné rozšíření možností detekce v porovnání s předchozími modely.

### 3.2.1 Dvoufázové modely

Dvoufázové (dvoustupňové) modely jsou koncipovány tak, aby efektivně identifikovaly a lokalizovaly objekty v obrázcích nebo videích prostřednictvím dvou hlavních fází. První spočívá ve výběru potenciálních regionů zájmů (RoIs) a druhá se zaměřuje na klasifikaci a přesnější lokalizaci objektů. Dvoufázové modely jsou základem pro pokročilé algoritmy detekce objektů nabízející vynikající kombinaci přesnosti a flexibilitu pro různé aplikace počítačového vidění, nicméně jsou pomalejší a mají vyšší výpočetní náročnost ve srovnání s jednofázovými kvůli nutnosti zpracování dvou samostatných fází, kde každá vyžaduje významné množství výpočetních zdrojů.

#### Region-based Convolutional Neural Network

Region-based Convolutional Neural Network (R-CNN) Girshick et al. [6] je považován za první dvoufázový model, přinášející významný posun v přesnosti a efektivitě ve srovnání s předchozími metodami. Architektura R-CNN (viz Obrázek 3.2) integruje hluboké konvoluční neuronové sítě pro extrakci vlastností z obrazu s algoritmem vyhledávání regionů, který identifikuje potenciální regiony zájmů. Tento proces zahrnuje následující kroky:

- 1. Selekce regionů zájmů (Region Proposal):** V prvním kroku R-CNN používá algoritmus pro generování návrhů regionů (například Selective Search (SS) Uijlings et al. [10]), které mohou obsahovat hledané objekty. Tento krok je nezbytný pro redukci počtu oblastí, které budou následně analyzovány, a zároveň pro zachování vysoké pravděpodobnosti, že všechny relevantní objekty budou pokryty. SS kombinuje hierarchické segmentace obrazu a strategie pro slučování na základě podobnosti, pomocí nichž generuje potenciální regiony objektů. Tento proces lze matematicky popsat pomocí funkce podobnosti  $S$ , která určuje míru, s jakou by dva regiony  $R_i$  a  $R_j$  měly být spojeny. Podobnost se obvykle měří na základě barvy, textury, velikosti nebo tvaru, a je definována jako:

$$S(R_i, R_j) = \alpha \cdot S_{color}(R_i, R_j) + \beta \cdot S_{texture}(R_i, R_j) + \gamma \cdot S_{size}(R_i, R_j) + \delta \cdot S_{shape}(R_i, R_j) \quad (3.1)$$

kde  $S_{color}$ ,  $S_{texture}$ ,  $S_{size}$ ,  $S_{shape}$  jsou složky podobnosti založené na barvě, textuře, velikosti, tvaru a  $\alpha$ ,  $\beta$ ,  $\gamma$ ,  $\delta$  jsou váhové koeficienty pro tyto složky.

- 2. Extrakce vlastností pomocí CNN:** Pro extrakci vlastností R-CNN používá předtrénovanou CNN. Vstupní obraz regionu je převeden do pevně dané velikosti (např.  $224 \times 224$  pixelů) a následně je zpracován CNN. Výstupem CNN je vektor vlastností  $F$  pro každý region. Pokud uvažujeme CNN s  $L$  vrstvami, transformaci vstupního obrazu  $I$  na vektor vlastností lze zapsat jako:

$$F = f_L(\dots(f_2(f_1(I; \theta_1); \theta_2)\dots); \theta_L) \quad (3.2)$$

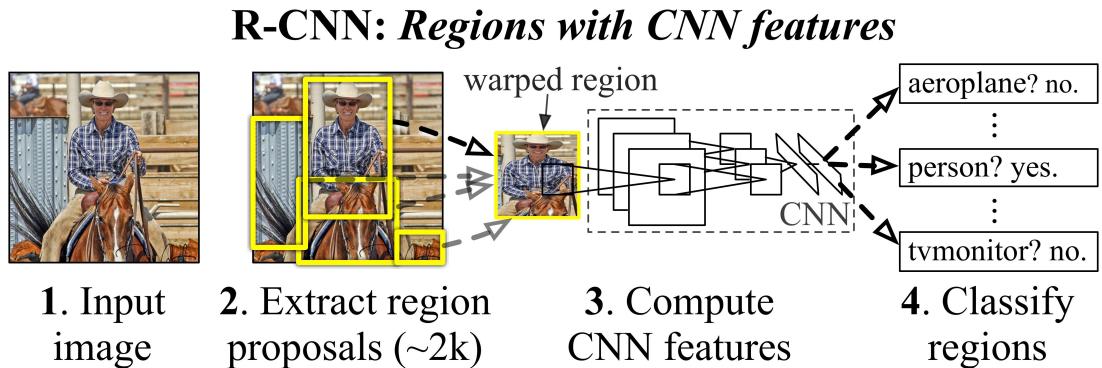
**3. Klasifikace a regrese ohraničujících boxů:** Po extrakci vlastností prochází každý vektor vlastností  $F$  klasifikačním a regresním modelem. Klasifikace se typicky provádí pomocí SVM Cortes a Vapnik [11], který pro každou třídu  $c$  vypočítá skóre  $S_c$ , zatímco regrese upřesňuje polohu ohraničujícího boxu. Klasifikační model pro třídu  $c$  a regresní model pro upřesnění ohraničujícího boxu jsou definovány jako:

$$S_c = w_c^T F + b_c \quad (3.3)$$

$$B = w_{reg}^T F + b_{reg} \quad (3.4)$$

kde  $w_c$  a  $b_c$  jsou parametry klasifikátoru pro třídu  $c$ ,  $w_{reg}$  a  $b_{reg}$  jsou parametry regresního modelu.  $B$  představuje korigované parametry ohraničujícího boxu (např. střed, šířka, výška).

Mezi problémy tohoto modelu patří zejména *vysoká výpočetní náročnost*, jelikož pro každý region se musí provést samostatný průchod CNN, a *neflexibilita SVM klasifikátorů*, které musí být natrénovány odděleně od CNN. Fast R-CNN Girshick [12] a Faster R-CNN Ren et al. [13] se snaží tyto nedostatky řešit pomocí efektivnějších algoritmů pro výběr regionů a integrace procesů učení do jediného end-to-end tréninkového procesu.



**Obrázek 3.2** Architektura R-CNN. Zdroj: <https://paperswithcode.com/method/r-cnn>

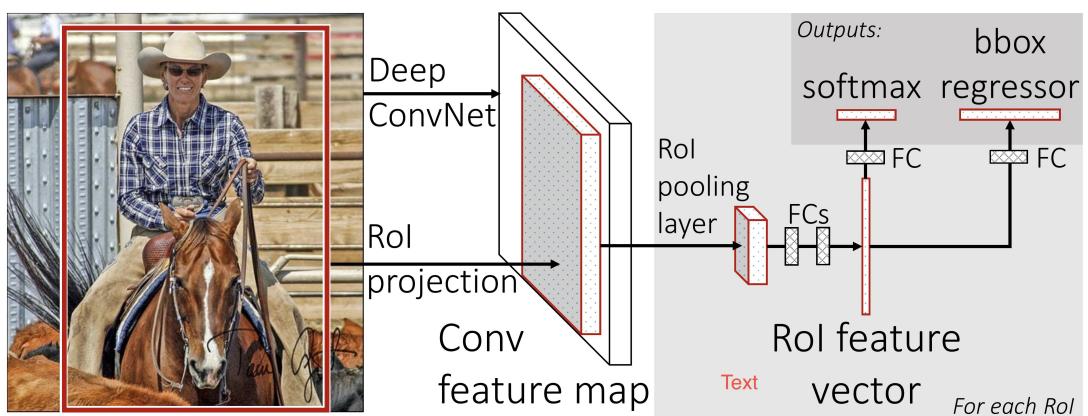
### Evoluce dvoufázové detekce objektů

Modely Fast R-CNN a Faster R-CNN představují signifikantní vylepšení původního R-CNN v efektivitě a přesnosti detekce objektů.

**Fast R-CNN** Fast R-CNN Girshick [12] (viz Obrázek 3.3) snižuje vysokou výpočetní náročnost a neefektivní tréninkový proces pomocí následujících vylepšení:

- **Sdílená extrakce vlastností:** Na rozdíl od R-CNN, která provádí extrakci vlastností pro každý navržený region zájmu (Region of interest, RoI) zvlášť, Fast R-CNN zpracovává celý vstupní obraz jednou jedinou CNN k vytvoření komplexní mapy vlastností. Na tuto mapu se pak aplikují RoI, což výrazně zrychluje proces extrakce.

- **RoI pooling vrstva:** RoI Pooling vrstva ve Fast R-CNN transformuje extrahované vlastnosti každého RoI na vektor pevné velikosti  $C \times C$ , což umožňuje použití těchto vektorů v plně propojených vrstvách, bez ohledu na původní velikost a tvar. Tento proces funguje tak, že každá RoI na mapě vlastností je rozdělena na mřížku  $C \times C$  buněk, kde  $C$  je pevně stanovená velikost výstupního vektoru vlastností. Pro každou buňku  $(i, j)$  mřížky se následně provede max pooling, kde se vybere maximální hodnota z vlastnosti spadajících do této buňky, což vede k vytvoření vektoru vlastností s pevnou velikostí. Ty se pak využívají pro klasifikaci a regresi ohraničujících boxů, což umožňuje precizní lokalizaci a identifikaci objektů v obraze.
- **Integrace klasifikace a regrese ohraničujících boxů:** Na rozdíl od R-CNN, kde se pro klasifikaci a regresi používaly oddělené modely, Fast R-CNN integruje obě tyto úlohy do jednoho výstupního vrstevního bloku, což zjednoduší tréninkový proces a zlepšuje celkovou přesnost modelu.



**Obrázek 3.3** Architektura Fast R-CNN. Zdroj: <https://paperswithcode.com/method/fast-r-cnn>

**Faster R-CNN** Rozvoj Fast R-CNN také vedl k vytvoření Faster R-CNN Ren et al. [13] (viz Obrázek 3.5), mezi jehož klíčová vylepšení patří:

- **Region proposal network (RPN):** RPN využívá mapu vlastností získanou z předchozí CNN k predikci ohraničujících boxů a jejich „objektivnosti“. Pro každou pozici na mapě vlastností, RPN využívá sadu anchor boxů (viz Obrázek 3.4) různých velikostí a poměrů stran (např.  $\frac{1}{2}, 1, 2$ ). Jak je diskutováno v Liu et al. [8], využití většího množství anchor boxů různých poměrů stran vede k lepším výsledkům. Pro každý z těchto anchor boxů RPN generuje dva typy výstupů: *skóre objektivity* ( $P_{obj}$ ), určující pravděpodobnost, že daný anchor box obsahuje objekt a *regresní predikce* ( $P_{req} = (dx, dy, dw, dh)$ ), určující posunutí a změněnou velikost anchor boxu, aby lépe odpovídala objektu.

Pokud  $A = (x_a, y_a, w_a, h_a)$  označuje původní rozměry anchor boxu a  $P_{reg}$  jsou regresní predikce, cílový ohraničující box  $B = (x_b, y_b, w_b, h_b)$  je vypočítán jako:

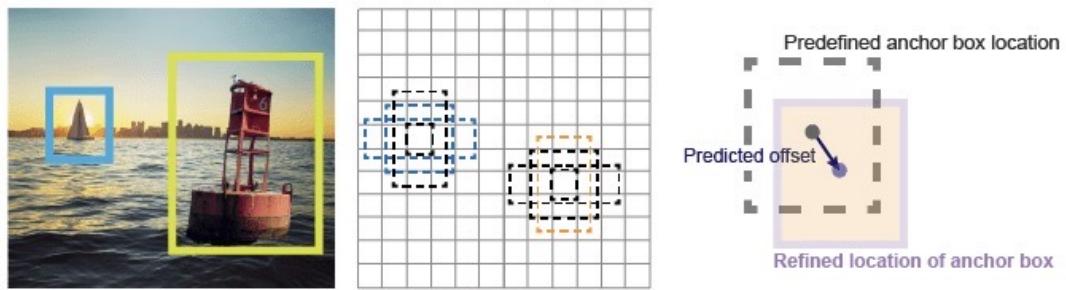
$$x_b = x_a + w_a * dx \quad (3.5)$$

$$y_b = y_a + w_a * dy \quad (3.6)$$

$$w_b = w_a * \exp(dw) \quad (3.7)$$

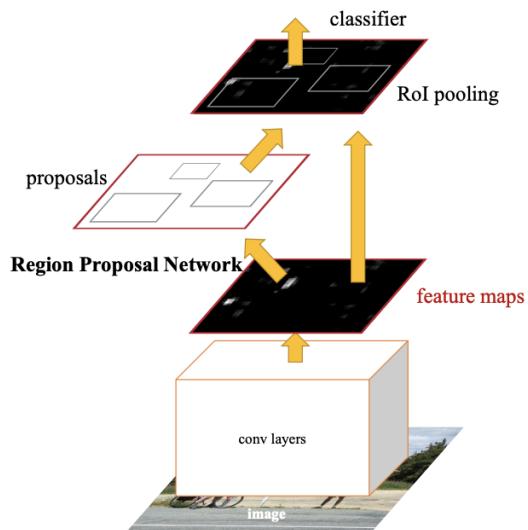
$$h_b = h_a * \exp(dh) \quad (3.8)$$

kde  $(x_b, y_b)$  a  $(w_b, h_b)$  jsou souřadnice respektive rozměry upraveného anchor boxu.



**Obrázek 3.4** První obrázek znázorňuje ground-truth ohraničující boxy, druhý ukazuje ukotvovací rámečky (anchor boxes) v předem stanovených polohách na mapě rysů a třetí demonstруje proces doladění polohy ukotvovacího rámečku s využitím predikovaného posunu. Zdroj obrázku: <https://paperswithcode.com/method/faster-r-cnn>

- **End-to-end trénink:** Integrace RPN do detekčního pipeline umožňuje trénovat celý model end-to-end, což vede k lepší optimalizaci a efektivitě, protože vstupní data jsou přímo mapovaná na požadovaný výstup bez potřeby mezíkroků.



**Obrázek 3.5** Architektura Faster R-CNN. Zdroj: <https://paperswithcode.com/method/faster-r-cnn>

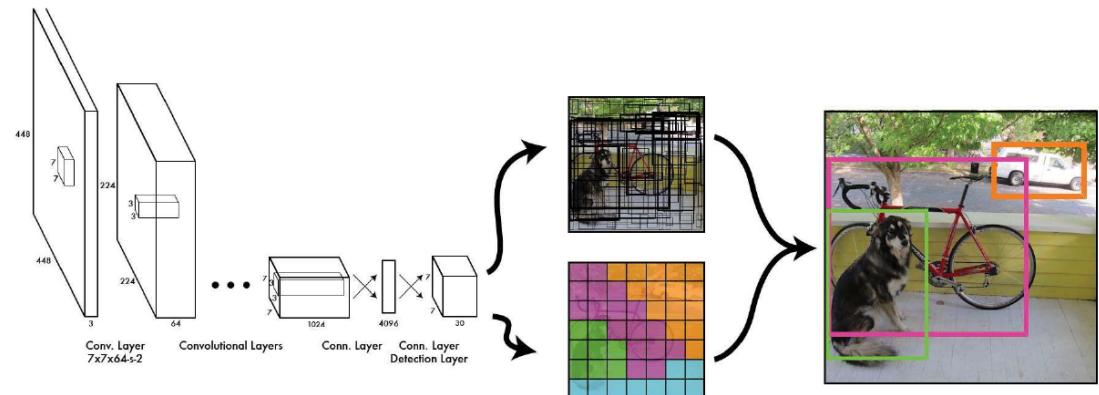
### 3.2.2 Jednofázové modely

Jednofázové (jednostupňové) modely integrují detekci a klasifikaci do jednoho procesního kroku. Tento přístup je rychlejší a přitom zachovává vysokou úroveň přesnosti. Přestože jednofázové modely mohou občas zaostávat za dvoufázovými modely v přesnosti, zejména v případě detekce malých objektů nebo objektů ve složitých scénách, jejich ostatní výhody jako nízká latence, větší efektivita a snadnější škálovatelnost z nich činí atraktivní volbu. V následující části se budeme věnovat specifickým realizacím, kterými jsou iterace YOLO, včetně YOLOv1, YOLOv8 a YOLO-World, a dále modelu SSD.

#### You Only Look Once (YOLO)

YOLO Redmon et al. [7] je jednofázový detekční model, který simultánně předpovídá klasifikaci a lokalizaci objektů v obrázku s vysokou rychlostí a přesností prostřednictvím jedné neuronové sítě používající jednoduchý regresní model.

Architektura YOLO (viz Obrázek 3.6) se skládá ze tří hlavních komponent: *backbone*, *neck* a *head*. Backbone je základní konvoluční síť zodpovědná za extrakci rysů z vstupního obrazu. Skládá se z řady vrstev, které slouží k transformaci a přenášení hlavních rysů k detekčním vrstvám. Ty pak provádí detekci objektů, vytvoření ohraničujících rámečků a klasifikaci tříd. Komponenta *neck* slouží jako zprostředkovatel extrahovaných rysů pro detekční hlavou. Často obsahuje struktury, které integrují a reorganizují rysy z různých úrovní hloubky *backbone* pro zvýšení semantické hodnoty a prostorové přesnosti. Hlava (*head*) v modelu YOLO je odpovědná za konečnou predikci zahrnující klasifikaci tříd objektů a lokalizaci pomocí ohraničujících rámečků.



**Obrázek 3.6** Architektura původní první verze modelu YOLO. Zdroj: <https://towardsdatascience.com/yolov1-you-only-look-once-object-detection-e1f3ffec8a89>

YOLO lze popsat následovně: Nechť  $I$  je vstupní obraz, který je rozdělen na  $S \times S$  (standardně  $7 \times 7$ ) mřížku. Pro každou buňku mřížky model předpovídá  $B$  ohraničujících rámečků a  $C$  pravděpodobností tříd  $C = p(c_i|Object)$ . Každý rámeček je popsán pěticí hodnot  $(x, y, w, h, confidence)$ , kde první čtyři elementy jsou totožné jako u Faster R-CNN a posledním prvkem pětice je  $confidence$ , která reflektuje jistotu, že rámeček obsahuje objekt a jak přesně je rámeček umístěn a je definovaná vztahem:

$$confidence = p(object) * IoU_{pred}^{truth} \quad (3.9)$$

Výstupní parametry modelu mají následující formu:

$$S \times S \times (5B + C) \quad (3.10)$$

Při uvažování příkladu sítě YOLO, kde každá buňka mřížky předpovídá dva ohraňující rámečky a evaluaci na datasetu sestávajícím z  $C$  tříd, je výstupní parametrizace vyjádřena jako:

$$7 \times 7 \times (5 \times 2 + C) \quad (3.11)$$

Pro efektivní řešení problémů s ohraňujícími rámečky YOLO implementuje dvě strategie: *penalizaci chyb* a *využití metody potlačení ne-maximálních hodnot (NMS)*. Velké penalizace jsou aplikovány na chyby v rámečcích s objektem ( $\lambda_{coord} = 5$ ) a menší penalizace na rámečky bez objektu ( $\lambda_{noobj} = 0.5$ ), což zahrnuje jak přesnost souřadnic a rozměrů rámečků, tak i confidence skóre. NMS dále řeší kolize mezi rámečky tím, že odstraňuje ty s nižším IoU<sup>1</sup> a ponechává pouze největší překryv s pravdivými rámečky. Tento přístup minimalizuje chyby při detekci a zvyšuje celkovou přesnost modelu.

Ztrátová funkce YOLO integruje několik kritických aspektů:

- Ztráta polohy a rozměrů ( $\lambda_{coord}$ ) zdůrazňuje přesnost predikce souřadnic  $x$ ,  $y$  a škálování  $w$ ,  $h$  pro rámečky obsahující objekty s větší váhou pro menší rámečky:

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^B 1_{ij}^{\text{obj}} \left[ (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right] \quad (3.12)$$

- Ztráta confidence skóre pro rámečky s objektem a bez objektu ( $\lambda_{noobj}$ ) rozlišuje mezi správně a chybně detekovanými objekty v rámečcích:

$$\sum_{i=0}^{S^2} \sum_{j=0}^B \left[ 1_{ij}^{\text{obj}} (c_i - \hat{c}_i)^2 + \lambda_{noobj} 1_{ij}^{\text{noobj}} (c_i - \hat{c}_i)^2 \right] \quad (3.13)$$

- Klasifikační ztráta pro rámečky obsahující objekty, která zajišťuje správnou identifikaci třídy objektu:

$$\sum_{i=0}^{S^2} 1_{ij}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \quad (3.14)$$

Tato ztrátová funkce ve spojení s NMS umožňuje YOLO dosáhnout vysoké přesnosti a rychlosti detekce a model je schopen efektivně vyřešit problémy spojené s nadměrnými nebo chybně označenými rámečky.

---

<sup>1</sup>IoU (Intersection over Union) je metrika pro hodnocení přesnosti modelů pro detekci objektů a segmentaci obrazu. Měří poměr překrývající se plochy mezi predikcí a „ground truth“ k jejich celkové ploše. Hodnota 1 značí dokonalé překrytí, 0 žádné.

YOLO model je schopný se naučit celkovou distribuci tříd objektů a jejich vzhled, což přináší schopnost generalizace na nové, dříve neviděné obrázky. Také dokáže detekovat objekty s malou chybou lokalizace.

Na základě tohoto modelu vzniklo několik dalších verzí postupně vylepšujících přesnost, rychlosť a schopnost generalizace, například YOLOv2 Redmon a Farhadi [9], YOLOv3 Redmon a Farhadi [14], YOLOv5 Jocher et al. [2], YOLOv7 Wang et al. [15] a YOLOv8 Jocher et al. [2].

## YOLOv8

YOLOv8 Jocher et al. [2] je aktuálně nejlepší state-of-the-art verze YOLO modelů, která se stala populární a široce používanou v praxi díky své dostupnosti. Mezi hlavní technologie, které model používá, patří:

- **Anchor-free detekce:** Tradiční YOLO modely spoléhaly na předdefinované anchor boxy pro identifikaci potenciálních poloh objektů. YOLOv8 používá anchor-free detekci, která přímo predikuje čtyři atributy ohraničujícího rámečku bez nutnosti pevně nastavených kotvících boxů, což umožňuje se dynamicky přizpůsobit různým velikostem a tvarům objektů a zlepšuje jeho schopnost detekce
- **C2f modul (Course-to-Fine):** C2F modul využívá hierarchickou strukturu pro postupné zpřesňování predikcí ohraničujících rámečků od hrubých predikcí k jemnějším a přesnějším výstupů prostřednictvím řady konvolučních a up-sampling vrstev. Tento proces lze matematicky modelovat jako sérii transformací, které postupně zlepšují predikce rámečků s ohledem na jejich lokaci a rozměry
- **GIoU ztrátová funkce:**

$$L_{GIoU} = 1 - \frac{|A \cap B|}{|A \cup B|} + \frac{|C - (A \cup B)|}{|C|} \quad (3.15)$$

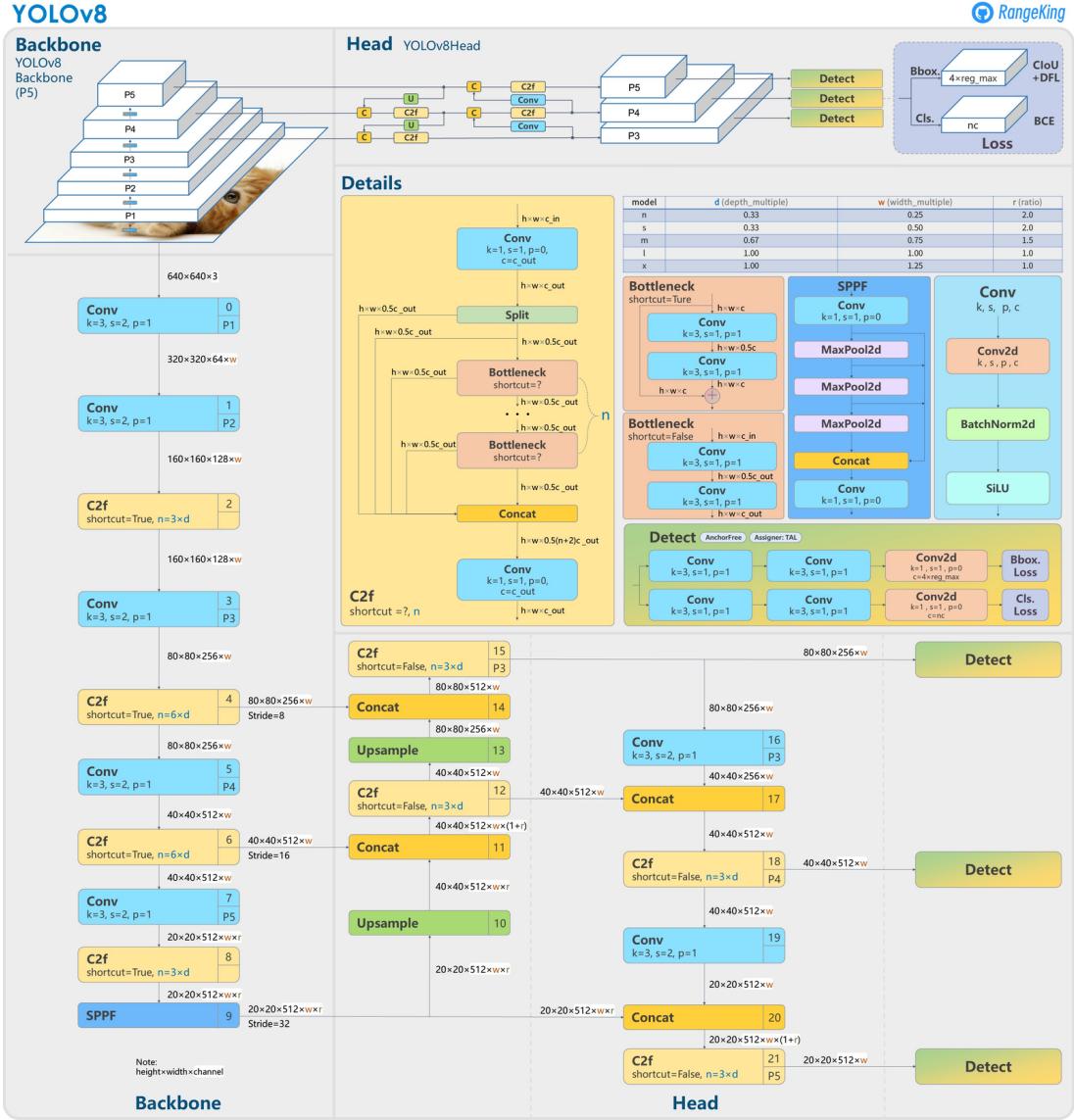
kde  $A$  je predikovaný ohraničující rámeček,  $B$  je pravdivý rámeček a  $C$  je nejmenší obdélník obsahující oba rámečky  $A$  a  $B$ . Tato ztráta nejenže zohledňuje překrytí mezi  $A$  a  $B$ , ale také penalizuje vzdálenost mezi rámečky, pokud se nepřekrývají.

- **DFL ztrátová funkce:** DFL je varianta Focal Loss, která se zaměřuje na minimalizaci rozdílu mezi predikovanou a pravdivou distribucí pravděpodobnosti tříd. Je definována jako:

$$L_{DFL} = - \sum_{c=1}^C y_{o,c} (1 - p_{o,c})^\gamma \log(p_{o,c}) \quad (3.16)$$

kde  $y_{o,c}$  je indikátor, jestli objekt  $o$  patří do třídy  $c$ ,  $p_{o,c}$  je pravděpodobnost, že  $o$  patří do třídy  $c$  a  $\gamma$  je modulující exponent.

- **Decoupled head:** Architektura s oddělenými hlavami pro detekci a klasifikaci s využitím specifických ztrátových funkcí pro každou hlavu umožňuje modelu zvládnout tyto úlohy nezávisle na sobě, což vede k optimalizaci procesu učení.



**Obrázek 3.7** Architektura YOLOv8. Zdroj: <https://blog.roboflow.com/what's-new-in-yolov8/>

## YOLO-World

YOLO série detektorů si získala popularitu pro svou efektivitu a praktičnost v rámci detekce objektů. Její závislost na předem definovaných a natrénovaných kategoriích objektů omezuje použitelnost v otevřených scénářích, kdy není k dispozici dataset pro trénink. YOLO-World Cheng et al. [16] rozšiřuje možnosti YOLO o detekci s otevřeným slovníkem díky předtrénování na velkých datasetech a spojení vizuálních a jazykových modelů. Klíčovým přínosem je nově navržená Re-parameterizable Vision-Language Path Aggregation Network (RepVL-PAN) a region-textová kontrastní ztrátová funkce, které umožňují efektivní interakci mezi vizuálními a lingvistickými informacemi, čímž se podstatně zlepšuje schopnost detekce širokého rozsahu objektů bez nutnosti předchozího učení na specifických kategoriích.

YOLO-World implementuje metodu *prompt-then-detect*. Tato metoda zahrnuje offline slovník s předem zakódovanými kategoriemi. Proces kódování se uskutečňuje

při inicializaci modelu s využitím CLIP encoderu Radford et al. [17], což vede k vytvoření statické vektorové reprezentace pro každou kategorii. Díky tomu může YOLO-World provádět detekce rychleji, protože se eliminuje nutnost vytvářet nové vektorové reprezentace pro kategorie při každé inferenci.

YOLO-World během inferenčního cyklu generuje vektorové reprezentace pro detekované objekty, které se pak porovnávají s kategoriemi v offline slovníku. CLIP encoder, díky svému tréninku na velkém množství dat, je schopen efektivně generalizovat a vytvářet přesné vektorové reprezentace. Pro kategorie objektů zakódované tímto modelem se YOLO-World v tréninku snaží vytvořit podobné vektorové reprezentace s cílem minimalizovat ztrátovou funkci. Díky tomuto procesu YOLO-World produkuje vektorové reprezentace objektů, které jsou podobné těm, které generuje CLIP. Tato podobnost umožňuje efektivní párování mezi detekovanými objekty a předem zakódovanými kategoriemi v offline slovníku. Klíčem k úspěchu je schopnost YOLO-Worldu generovat podobné a dobré generalizované vektorové reprezentace.

### Single-Shot Detector

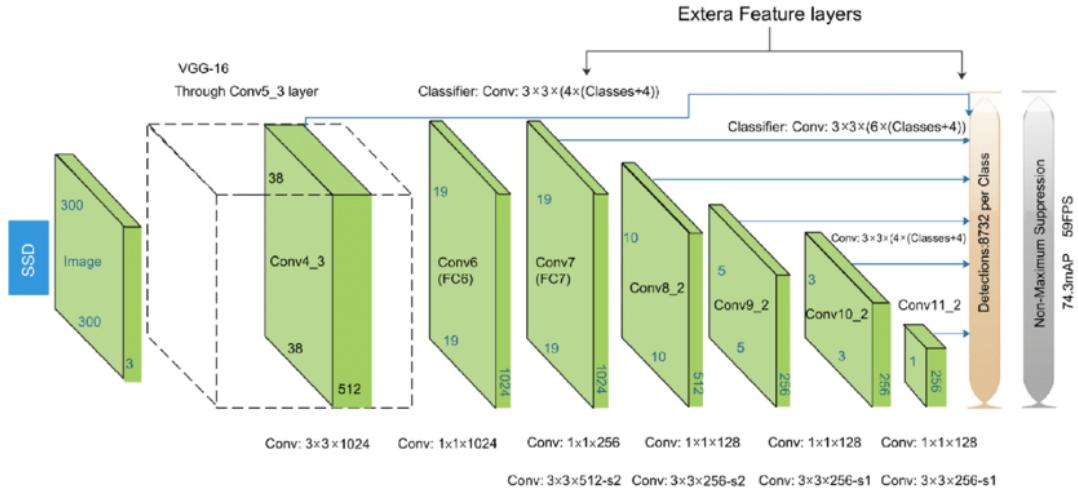
Single-Shot Detector (SSD) Liu et al. [8] kombinuje výhody rychlosti jednofázových modelů s vylepšenou schopností detekce objektů různých velikostí pomocí používání více konvolučních filtrů různých velikostí v posledních vrstvách CNN. Tento přístup umožňuje detekovat velké i malé objekty na různých úrovních rozlišení v jednom průchodu obrazem, čímž překonává omezení modelů YOLO.

Architektura SSD (viz Obrázek 3.8) přistupuje k detekci objektů tím, že na různých úrovních konvoluční sítě předpovídá ohraňující boxy a pravděpodobnosti tříd. Pro každou z vrstev sítě predikuje tensor o rozměrech  $M \times N \times (4 + C) \times K$ , kde  $M$  a  $N$  jsou rozměry prostorové mapy vlastností, 4 reprezentuje parametry ohraňujícího boxu ( $x, y, w, h$ ),  $C$  je počet tříd a  $K$  je počet boxů predikovaných na každou pozici mapy vlastností. SSD používá ztrátovou funkci, která kombinuje *lokalizační ztrátu* obvykle využívající Smooth L1 mezi predikovanými a skutečnými boxy pro objekty a *klasifikační ztrátu*, kterou je obvykle cross-entropy, pro určení přesnosti třídy objektu v daném boxu. SSD rovněž používá techniku hard negative mining (HNM) pro snižování falešně pozitivních predikcí.

Existují různé rozšiřující varianty architektury SSD jako MobileNetV2 Sandler et al. [18] určený pro dosažení větší efektivity na mobilních zařízeních, nebo EfficientDet Tan et al. [19] využívající vícestupňových architektur.

### 3.2.3 Komparace výkonu modelů

V rámci aplikací vyžadujících zpracování obrazu v reálném čase je pro nás systém pro real-time analýzu a validaci obrazu z webových kamer zásadní nejen přesnost detekce, ale také rychlosť, s jakou model dokáže obrazy zpracovávat. Tato část se věnuje porovnání výkonu modelů Faster R-CNN, SSD a YOLO na datasetu MSCOCO Lin et al. [20] z roku 2017 s ohledem na dvě nejdůležitější metriky: *přesnost* a *rychlosť*.



**Obrázek 3.8** Architektura SSD. Zdroj: [https://www.researchgate.net/publication/360720627\\_Interrogating\\_the\\_Installation\\_Gap\\_and\\_Potential\\_of\\_Solar\\_Photo](https://www.researchgate.net/publication/360720627_Interrogating_the_Installation_Gap_and_Potential_of_Solar_Photo)

### Mean Average Precision

Mean Average Precision (mAP) je standardní metrika pro evaluaci modelů detekce objektů, která zohledňuje *přesnost (precision)* a *úplnost (recall)*. Přesnost je definována jako poměr správně identifikovaných objektů k celkovému počtu detekcí provedených modelem. Úplnost měří, kolik relevantních objektů bylo modelem správně identifikováno z celkového počtu relevantních objektů ve zkoumaném datasetu. Hodnota mAP je vypočtena jako průměr hodnot Average Precision (AP) pro všechny třídy objektů, kde AP pro každou třídu je určena integrací křivky přesnosti v závislosti na úplnosti při různých prahových hodnotách detekce. Matematicky lze AP pro jednu třídu vyjádřit jako průměr přesností vypočítaných v každém bodě křivky přesnosti a úplnosti:

$$AP = \frac{1}{n} \sum_{k=1}^n \text{Precision}(k) \quad (3.17)$$

kde  $n$  je počet prahových hodnot. mAP je pak průměr těchto AP hodnot pro všechny třídy.

### Floating-point Operations Per Second

Floating-point Operations Per Second (FLOPs) udává počet výpočtů s plovoucí desetinnou čárkou, které model vykoná za sekundu. Jedná se o metriku určující výpočetní náročnost modelů. Ve strojovém učení a zejména u modelů pro detekci objektů informuje o tom, kolik operací je nutné pro analýzu jednoho vstupního snímku. Model s vysokým počtem FLOPs může vykazovat pomalejší zpracování a vysší nároky na výpočetní zdroje. Modely s nižším počtem FLOPs jsou obecně rychlejší, a proto jsou i vhodnější pro aplikace vyžadující okamžité zpracování obrazu.

Model	vstup	mAP	rychlosť	parametry	FLOPs
YOLOv8n	640px	37.3%	0.99ms	3.2M	8.7B
YOLOv8s	640px	44.9%	1.20ms	11.2M	28.6B
YOLOv8m	640px	50.2%	1.83ms	25.9M	78.9B
SSD ResNet152 V1	640px	35.4%	80ms	-	-
SSD MobileNet V2	640px	28.2%	39ms	-	-
Faster R-CNN IRN V2	640px	37.7%	206ms	-	-
EfficientDet-D2(640)	640px	41.7%	14.8ms	8.1M	11.0B
EfficientDet-D3(640)	640px	44.0%	18.7ms	12.0M	24.9B

**Tabulka 3.1** Porovnání výkonnosti modelů na detekci objektů

Prezentovaná data (viz Tabulka 3.1) jsou shromážděna z více zdrojů<sup>2</sup> <sup>3</sup> <sup>4</sup>. Modely YOLOv8 byly testovány na grafické kartě Nvidia A100. Při testování byl použit framework TensorRT. EfficientDet modely byly evaluovaly na grafické kartě Nvidia V100 bez použití TensorRT. Ostatní modely byly testovány na strojích, u kterých nebyly uvedeny specifikace. Vzhledem k těmto rozdílům je porovnání výsledků pouze orientační.

Pro výběr nejlepšího modelu pro náš systém pečlivě zvažujeme různé aspekty, které zahrnují přesnost, rychlosť, výpočetní náročnosť a specifika testovacího hardwaru. Modely YOLOv8 poskytují významnou výhodu v rychlosti a efektivitě zpracování, což je zásadní pro aplikace vyžadující zpracování obrazu v reálném čase. YOLOv8s, s mAP 44.9% a rychlosťí 1.20ms, se jeví jako nejlepší volba, protože i přes svou vyšší výpočetní náročnosť je schopen dosáhnout vynikající rovnováhy mezi přesností a rychlosťí díky optimalizaci pro vysoký výkon na špičkovém hardwaru.

Při porovnání výkonnosti modelů YOLOv8 s SSD a Faster R-CNN vynikají YOLOv8 modely ve všech sledovaných metrikách. Ačkoliv je možné pozorovat adekvátní přesnost u modelů SSD a Faster R-CNN, jejich rychlosť zpracování a výpočetní efektivita jsou znatelně nižší. Kvůli neznámému hardwaru nemůžeme považovat rychlosť za zcela relevantní faktor při srovnání s ostatními modely. Avšak i kdybychom tento faktor pominuli, SSD a Faster R-CNN jsou stále výrazně horší, což je značně alarmující. Z těchto důvodů nebyly vybrány pro další použití.

EfficientDet je potenciálně zajímavou volbou kvůli jeho dobrému kompromisu mezi přesností a výpočetní náročností. YOLOv8s ho ale předčil v rychlosti zpracování se stále vysokou úrovňí přesnosti. Pro budoucí výzkumné práce s rozdílnými požadavky by EfficientDet mohl představovat velmi atraktivní volbu.

FLOPs jako metrika rychlosť poskytuje hlubší vhled do výpočetní efektivity modelů bez závislosti na specifickém hardwaru. Tato metrika nám umožnila objektivně posoudit, že i přes relativně vysokou výpočetní náročnost YOLOv8s nabízí optimální výkon pro real-time aplikace v kontextu našeho systému. FLOPs a počet parametrů některých modelů nebyly v předložených zdrojích dostupné

<sup>2</sup>Zdroje EfficientDet výsledků: <https://github.com/google/automl/tree/master/efficientdet>, <https://medium.com/codex/review-efficientdet-scalable-and-efficient-object-detection-ed9ebc70f873>

<sup>3</sup>Zdroj YOLOv8 výsledků: <https://github.com/ultralytics/ultralytics>

<sup>4</sup>Zdroj výsledků ostatních modelů: [https://github.com/tensorflow/models/blob/master/research/object\\_detection/g3doc/tf2\\_detection\\_zoo.md](https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md)

nebo specifikovány, což omezuje schopnost provést úplně komplexní srovnání mezi všemi zkoumanými modely.

YOLOv8s je tedy považován za nevhodnější volbu pro náš systém, byť ostatní modely skýtají potenciál pro jiné aplikace a výzkumné účely.

### 3.3 Metody klasifikace obrazu

Klasifikace obrazu patří do oblasti počítačového vidění a umožňuje automaticky kategorizovat obsah obrazových dat do předem definovaných tříd. Rozvoj těchto metod byl v posledních letech značně urychljen díky pokrokům v oblasti strojového učení. Významným milníkem se stalo vítězství modelu AlexNet Krizhevsky et al. [21] v soutěži ImageNet Large Scale Visual Recognition Challenge, které posunulo hranice možností konvolučních neuronových sítí a otevřelo cestu k dalšímu intenzivnímu výzkumu v této oblasti. Tento průlom spolu s nástupem hlubokého učení umožnil dosahovat významných úspěchů v přesnosti a spolehlivosti systémů pro klasifikaci obrazu.

#### 3.3.1 Hluboké učení v klasifikaci obrazu

Metody hlubokého učení se staly dominantním přístupem díky jejich schopnosti automaticky extrahat a učit se reprezentace dat přímo z dat, což eliminuje potřebu manuálního výběru a návrhu vlastností. Konvoluční neuronové sítě (CNN), které jsou speciálním typem hlubokých neuronových sítí navržených pro zpracování obrazových dat, představují základní stavební blok pro většinu moderních systémů klasifikace obrazu. Základem CNN je matematická operace konvoluce, při které se dvě funkce kombinují, aby vytvořily třetí, která reprezentuje, jak se tvar jedné funkce mění v důsledku druhé. Konvoluční vrstvy v CNN aplikují sadu filtrů (kernelů) na vstupní obraz nebo na výstupy předchozích vrstev, aby detekovaly specifické vlastnosti jako jsou hrany, textury nebo jiné vzory. Tento proces umožňuje modelům hlubokého učení se automaticky naučit hierarchii vlastností, které jsou relevantní pro specifické úlohy klasifikace, což vede k vysoké přesnosti a generalizaci schopnosti na nových datech.

V konvoluční vrstvě se filtr  $F$  o rozměrech  $h \times w \times d$ , kde  $h$  je výška,  $w$  je šířka a  $d$  je hloubka filtru, aplikuje na vstupní data  $I$  operací konvoluce, která je definovaná jako:

$$A_{ijk} = (F * I)_{ijk} = \sum_m \sum_n \sum_o F_{mno} \cdot I_{(i+m)(j+n)(k+o)} \quad (3.18)$$

kde  $A_{ijk}$  je výsledná aktivace,  $*$  označuje operaci konvoluce.

Významnou roli v CNN hraje pooling vrstva sloužící k redukci rozměrů vstupního obrazu a ke zvýšení odolnosti proti posunutí vstupních dat. Max pooling, nejčastější forma pooling vrstvy, redukuje velikost aktivace vybíráním maximální hodnoty z předdefinovaného okna. Pro okno o rozměrech  $2 \times 2$  je výstup  $P$  definován jako:

$$P_{ij} = \max(A_{2i:2i+1,2j:2j+1}) \quad (3.19)$$

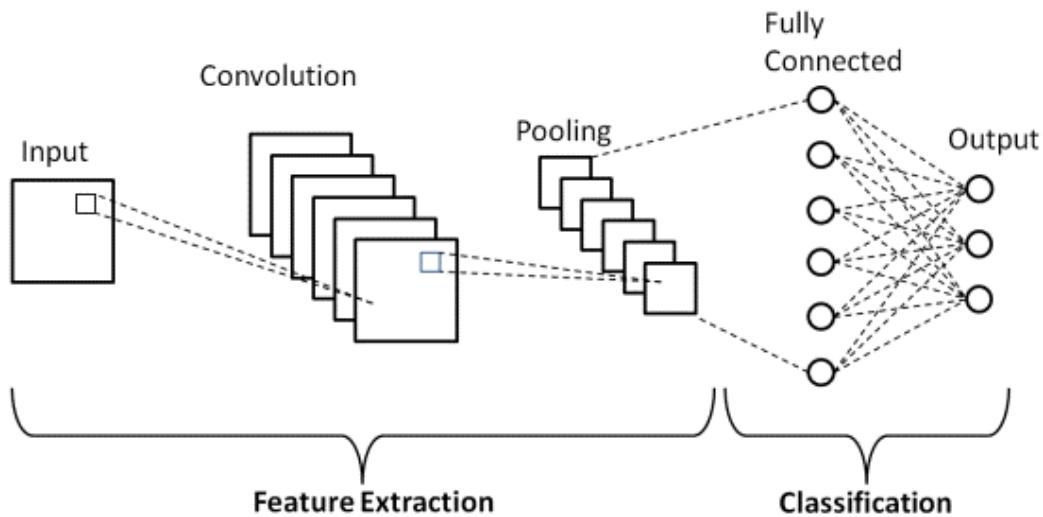
kde  $A$  je aktivace z předchozí konvoluční vrstvy.

Kromě max pooling existují i další typy pooling operací, jako je average nebo L2-norm pooling. Obě tyto pooling operace se liší pouze funkcí, která mapuje hodnoty z předdefinovaného okna na jednu hodnotu.

Po sérii konvolučních a pooling vrstev následují plně propojené vrstvy, které integrují vlastnosti detekované na různých částech obrazu a umožňující klasifikaci. Vstup do plně propojené vrstvy je vektorizovaný (převeden na jednorozměrný vektor), a každý neuron v této vrstvě je propojen s každým vstupem. Výstup  $y$  z plně propojené vrstvy je dán vztahem:

$$y = \sigma(Wx + b) \quad (3.20)$$

kde  $W$  je matici vah,  $x$  je vstupní vektor,  $b$  je bias, a  $\sigma$  je aktivační funkce, kterou je často sigmoid nebo softmax.



**Obrázek 3.9** Základní architektura konvoluční neuronové sítě. Zdroj: <https://www.upgrad.com/blog/basic-cnn-architecture>

Sigmoid je typ aktivační funkce definovaná jako:

$$\text{sigmoid}(z) = \frac{1}{1 + e^{-z}} \quad (3.21)$$

kde  $z$  je lineární kombinace vstupů, tj.  $z = Wx + b$ .

Sigmoid se používá obvykle v poslední vrstvě pro binární klasifikační úlohy, kde je výstupem pravděpodobnost nálezení vzorku do jedné z dvou tříd. Dalším použitím je multi-labelová klasifikace, kde pro každou z možných tříd je použita sigmoid funkce umožňující vzorku náležet současně do více tříd.

Druhým typem aktivační funkce je softmax, který je definován následovně:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad (3.22)$$

kde  $z$  je tatáž lineární kombinace jako v případě sigmoid funkce.

Softmax funkce je typicky používaná v poslední vrstvě hlubokých neuronových sítí pro klasifikaci, kde potřebujeme určit, do které jedné z mnoha vstupních tříd vzorek patří.

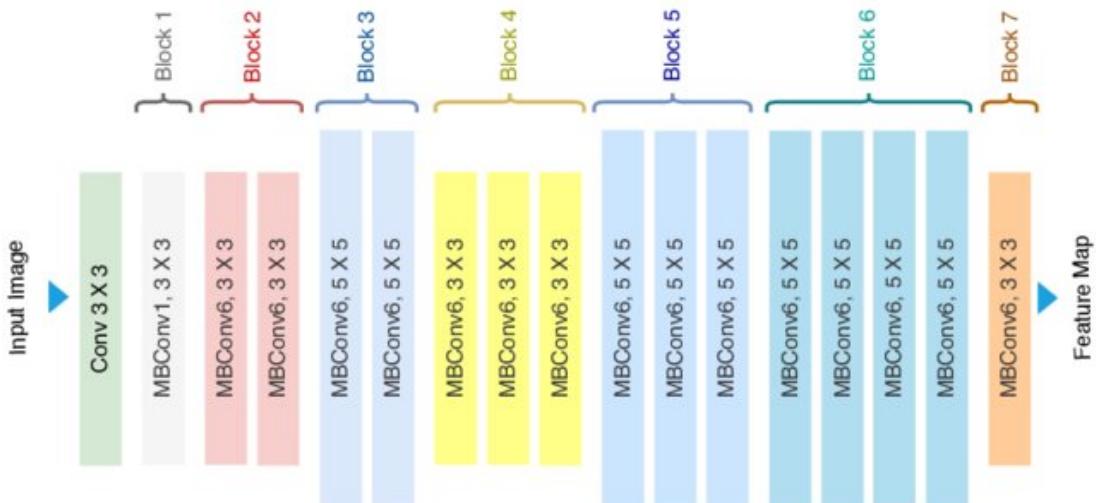
Moderní konvoluční neuronové sítě přinesly revoluci ve zpracování a klasifikaci obrazu, zlepšily přesnost a efektivitu v mnoha aplikacích a otevřely dveře k novým přístupům v oblasti počítačového vidění.

Významné architektury, které změnily odvětví:

- AlexNet Krizhevsky et al. [21]: Jeden z prvních modelů, který prokázal význam hlubokého učení v ImageNet soutěži. Jeho úspěch podnítil zájem o hluboké učení ve vědecké komunitě.
- VGGNet Simonyan a Zisserman [22]: Známý svou jednoduchou, ale hlubokou architekturou. Zlepšil rozpoznávání obrazu díky použití velmi malých konvolučních filtrů.
- ResNet He et al. [23]: Představil koncept reziduálního učení se *skip connections*, což umožnilo trénování velmi hlubokých sítí a výrazně zlepšilo přesnost klasifikace.

Vyjmenované architektury položily základy pro vývoj dalších inovativních modelů a technik, které kontinuálně posouvají hranice toho, co je možné v počítačovém vidění dosáhnout. Ukazatelem kvality modelů je přesnost definovaná jako poměr správně klasifikovaných případů k celkovému počtu případů. Mezi aktuální state-of-the-art modely patří:

- **EfficientNet**: Pro realtime aplikace je model *EfficientNet B0* Tan a Le [24] ideální volbou, nabízející vynikající rovnováhu mezi přesností a efektivitou. Nedílnou součástí architektury EfficientNet (viz Obrázek 3.10) jsou *MBConv bloky* (Mobile Inverted Bottleneck) Sandler et al. [18], které představují vylepšení tradičních konvolučních bloků používaných v mobilních sítích, jako je *MobileNet*. MBConv bloky využívají mechanismus inverzního reziduálního spojení spolu s lineárním bottleneckem a separabilními konvolucemi, které umožňují efektivní škálování šířky, hloubky a rozlišení sítě bez značného zvýšení výpočetní náročnosti. Předtrénovaný model na datasetu ImageNet s přesností 77.692%, při pouhých 3.9B FLOPs, a dostupností na PyTorch Hubu představuje nejlepší volbu pro aplikace, které vyžadují rychlé a přesné zpracování obrazu v reálném čase.
- **Vision Transformer (ViT)**: Vision Transformer (ViT) Dosovitskiy et al. [25] je založený na transformerech, které byly původně vyvinuté pro zpracování přirozeného jazyka, avšak ukazují slibné výsledky i v oblasti počítačového vidění. ViT přistupuje k obrazu jako k sekvenci tokenů (malých oblastí obrazu) a aplikuje na ně architekturu podobnou klasickým transformerům. Pro situace, kde je možné akceptovat mírně vyšší výpočetní náročnost výměnou za výhody transformerové architektury, je *ViT Tiny* s přesností 81.474% při 44.9B FLOPs výbornou volbou. Tento model je rovněž dostupný na PyTorch Hubu a nabízí alternativní řešení pro efektivní zpracování obrazu s vysokou přesností v reálném čase.
- **YOLOv8-CLS**: YOLOv8s-CLS Jocher et al. [2] je klasifikační model založený na základní konvoluční sítě sloužící k extrakci rysů architektury YOLOv8 známé z detekce objektů a adaptované pro efektivní klasifikaci obrazu. Tento model, konkrétně ve verzi „s“ - YOLOv8s, dosahuje přesnosti



**Obrázek 3.10** Architektura EfficientNet B0. Zdroj: [https://www.researchgate.net/publication/344410350\\_Classification\\_and\\_understanding\\_of\\_cloud\\_structures\\_via\\_satellite\\_figures?lo=1](https://www.researchgate.net/publication/344410350_Classification_and_understanding_of_cloud_structures_via_satellite_figures?lo=1)

73.8% na datasetu ImageNet s nízkými výpočetními náročnostmi, což ho činí vhodným pro rychlé zpracování obrazu v real-time aplikacích. V porovnání s modely jako je EfficientNet B0 a Vision Transformer (ViT) Tiny, které mohou nabízet vyšší přesnost, ale s většími výpočetními náročnostmi, YOLOv8s-CLS poskytuje výborný kompromis mezi přesností, efektivitou a rychlostí.

- **CLIP (Contrastive Language–Image Pre-training)**: OpenAI vyvinulo model CLIP Radford et al. [17] jako revoluční přístup k multimodálnímu pochopení obsahu, který umožňuje modelu učit se vizuální koncepty přímo z textového popisu. Model CLIP funguje na principu kontrastního učení, kde se model snaží asociovat obrázky s relevantními popisy v přirozeném jazyce. Díky trénování na rozmanitému datasetu z internetu, který obsahuje miliony obrázků spárovaných s textem, se CLIP vyznačuje výjimečnou schopností generalizace a aplikovatelnosti na široké spektrum úloh bez potřeby dalšího učení. Ačkoli CLIP představuje významný posun ve způsobu, jakým lze modely umělé inteligence trénovat a používat pro interakci mezi vizuálními a jazykovými informacemi, jeho výpočetní náročnost a poměrně vysoké latence činí tento model nevhodným pro aplikace v reálném čase, kde je důležitá rychlá odezva. Tento faktor omezuje jeho přímé využití v našel systému, avšak jeho schopnosti při tvorbě robustních a flexibilních AI systémů, které rozumějí širokému spektru vizuálních a textových dat, jsou bezprecedentní.

Zmíněné modely *EfficientNet B0*, *ViT Tiny* a *YOLOv8-CLS* předtrénované na datasetu *ImageNet* dostupné na Pytorch Hubu<sup>5</sup> vybíráme jako nejlepší potenciální adepty na výsledné řešení real-time klasifikace pro náš vyvýjený systém. Oba modely nabízí konkurenční výsledky na benchmarku ImageNet a poskytují vysokou úroveň přesnosti při zachování nízké výpočetní náročnosti. Oba modely představují v rámci svých kategorií ten nejlepší poměr mezi přesností a výpočetní náročností.

<sup>5</sup>Webový zdroj PyTorch Hubu: <https://pytorch.org/hub/>

Modely vybíráme jen a přímo z PyTorch Hubu s ohledem na jejich jednoduchou implementaci do systémů. Tato platforma umožňuje vývojářům snadno integrovat nejnovější modely umělé inteligence do svých projektů bez potřeby rozsáhlé konfigurace nebo trénování od základu.

### 3.3.2 Statistické metody strojového učení v klasifikaci obrazu

Ve světle současných metod strojového učení si tradiční statistické přístupy ke klasifikaci obrazu jako rozhodovací stromy, support vector machines (SVM), a vícevrstvé perceptrony (MLP) udržují své místo v určitých aplikacích. Tyto metody jsou zvláště vhodné v situacích, kde je možné na základě předem získaných informací o obrazu jasně definovat kategorie. Klíčem k úspěchu těchto metod je využití vlastností (features), které jsou dostatečně informativní a relevantní pro danou úlohu klasifikace.

Vlastnosti, na kterých tyto statistické metody operují, musí mít jasný význam a kontext. Na rozdíl od hlubokého učení, které může pracovat přímo s pixely obrazu a extrahovat z nich relevantní vlastnosti v průběhu tréninku, tradiční metody vyžadují, aby vlastnosti byly definovány a extrahovány předem. V praxi to může znamenat využití algoritmů pro detekci hran, textur, barevných histogramů, nebo jiných charakteristik, které nesou informace samy o sobě a jsou schopny popsat obsah obrazu v užitečném formátu pro klasifikaci.

Jednou z výhod těchto tradičních metod je, že obvykle nevyžadují tak rozsáhlé množství dat pro trénink. Díky explicitní závislosti na předem definovaných vlastnostech je možné modely naučit rozpoznávat vzory a kategorizovat obrazy s menším množstvím tréninkových dat. Nicméně kvalita a relevance extrahovaných vlastností přímo ovlivňuje schopnost modelu provádět přesnou klasifikaci a jejich nedostatečné provedení může zhoršit výsledky modelů.

#### Vícevrstvé Perceptrony

Vícevrstvé perceptrony jsou základem mnoha moderních architektur neuronových sítí a patří mezi základní modely pro hluboké učení. Strukturálně se skládají z jedné vstupní vrstvy, několika skrytých vrstev a jedné výstupní vrstvy. Každá vrstva obsahuje neurony, které jsou plně propojené s neurony v předchozí a následující vrstvě. Pro každý neuron, výstup je vypočítán jako vážená suma jeho vstupů, k níž je přidán bias a na tento součet je aplikována nelineární aktivační funkce.

Matematicky lze výstup  $y_i^l$  neuronu  $i$  ve vrstvě  $l$  vyjádřit jako:

$$y_i^l = \phi \left( \sum_j w_{ij}^l x_j + b_i^l \right) \quad (3.23)$$

kde  $w_{ij}^l$  jsou váhy spojení mezi neuronem  $j$  v předchozí vrstvě a neuronem  $i$  ve vrstvě  $l$ ,  $b_i^l$  je bias neuronu  $i$ ,  $x_j$  jsou vstupy do neuronu, a  $\phi$  je aktivační funkce.

MLP jsou schopné modelovat složité nelineární vztahy díky své schopnosti vytvářet komplexní hranice rozhodnutí. Tuto schopnost lze připsat především nelineárním aktivačním funkcím, jako jsou *ReLU*, *sigmoida* nebo *tanh*, které umožňují sítím učit se nelineární transformace. Výběr aktivační funkce a konfigurace

architektury sítě (počet vrstev a neuronů) jsou důležité aspekty návrhu MLP, které ovlivňují jejich výkonnost a schopnost generalizace.

Jedním z problémů MLP je přeúčení, což je situace, kdy model příliš dobře zapadá do tréninkových dat a nedokáže efektivně generalizovat na nová neviděná data. Tento problém se často řeší technikami regularizace, jako je L1 a L2 regularizace, nebo metodami jako dropout, které náhodně „vypínají“ některé neurony během tréninku, čímž se riziko snižuje.

## Rozhodovací Stromy a Evoluce k XGBoost

Rozhodovací stromy jsou populární metoda strojového učení, která se používá pro úlohy klasifikace a regrese. Jádrem metody je stromová struktura, kde každý uzel reprezentuje „otázku“ na některé z příznaků dat, každá hrana k potomku reprezentuje odpověď na otázku a listy stromu reprezentují predikční výstupy nebo třídy. Vytváření rozhodovacího stromu zahrnuje výběr příznaků, který efektivně „rozděluje“ datovou sadu, a kritérium pro zastavení, které určuje, kdy se má uzel stát listem. Běžně používané metriky pro výběr atributů zahrnují:

Informační zisk, který je založen na entropii:

$$IG(D, A) = H(D) - \sum_{v \in Values(A)} \frac{|D_v|}{|D|} H(D_v) \quad (3.24)$$

Gini index pro měření čistoty rozdělení:

$$Gini(D) = 1 - \sum_{i=1}^n p_i^2 \quad (3.25)$$

kde  $H(D)$  je entropie datové sady D,  $D_v$  jsou podmnožiny D rozdělené podle hodnoty příznaku A a  $p_i$  je pravděpodobnost třídy i v datové sadě D.

Evolucí klasických rozhodovacích stromů je pak XGBoost Chen a Guestrin [26] (eXtreme Gradient Boosting), což je pokročilá implementace gradient boosting algoritmu. XGBoost využívá rozhodovací stromy jako základní prediktory a iterativně je přidává do souboru (ensemble) s cílem minimalizovat zadanou ztrátovou funkci.

Základem XGBoost je myšlenka, že přidávání stromů do modelu může být formulováno jako gradientní sestup v prostoru funkcí. Pro každý strom se optimalizuje:

$$Obj^{(t)} = \sum_{i=1}^n l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \quad (3.26)$$

kde  $l$  je ztrátová funkce,  $\hat{y}_i^{(t-1)}$  je predikce modelu do  $t-1$  iterace,  $f_t$  je strom přidaný k  $t$ -té iteraci a  $\Omega(f_t)$  je termín regularizace, který penalizuje složitost stromu.

XGBoost přináší významné vylepšení v oblasti výpočetní efektivity a predikční výkonnosti díky regularizaci pomocí penalizace za velkou komplexnost modelu, což pomáhá zabránit přeúčení. Dalším benefitem je implementace efektivní paralelizace a cache optimalizace, které značně zkracují čas potřebný pro trénink modelů na velkých datových sadách.

## Support Vector Machines (SVM)

Support Vector Machines (SVM) Cortes a Vapnik [11] jsou nástrojem pro řešení úloh klasifikace a regrese. SVM se snaží najít nadrovinu (nebo soubor nadrovin v prostoru vyšších dimenzí), která nejlepším způsobem odděluje data do různých tříd. Významným konceptem u SVM je maximalizace marže mezi třídami, což je vzdálenost mezi nadrovinou a nejbližšími datovými body z každé třídy, známými jako support vektory.

Základní idea SVM spočívá v transformaci vstupních dat do více dimenzionálního prostoru pomocí kernelového triku a následném nalezení optimální rozdělující hyperroviny v tomto prostoru. Cílem je najít parametry  $w$  a  $b$  nadroviny  $w \times x + b = 0$  takové, že marže mezi třídami je maximalizována. To lze formulovat jako minimalizaci  $\frac{1}{2}||w||^2$  s omezeními  $y_i(w \cdot x_i + b) \geq 1$  pro všechny datové body  $x_i$ , kde  $y_i$  jsou třídy, do kterých klasifikujeme.

SVM používá kernelové funkce k transformaci vstupních dat do více dimenzionálního prostoru, kde je snazší nalézt lineární oddělovací hranici. Běžné kernely zahrnují polynomiální, radiální bázové funkce (RBF) a sigmoidní kernely

SVM jsou známé svou robustností a efektivitou, zejména v úlohách s vysokou dimenzionalitou a kde je počet vzorků menší než počet dimenzí. Díky použití kernelového triku jsou schopny efektivně modelovat i velmi složité hranice rozhodnutí.

## 3.4 Shrnutí

V této kapitole jsme se zaměřili na přístupy řešení specifických problémů v oblasti zpracování obrazových dat. Diskutovali jsme o metodách kontroly obrazu, detekce objektů a klasifikace obrazu, poskytující jak teoretické základy, tak praktické implementace pro každou z těchto oblastí.

Pro kontrolu obrazu jsme zvážili jak tradiční metody počítačového vidění, tak moderní přístupy založené na hlubokém učení. Specificky jsme se zaměřili na tři hlavní problémy: analýzu obrazů bez relevantního obsahu, detekci částečně načtených snímků a detekci vertikálních poškození obrazu. Pro každý z těchto problémů jsme navrhli specifický algoritmus: *VoidSeeker*, *FragGuard* a *VertiScan*.

V oblasti detekce objektů jsme představili dvě hlavní kategorie detektorů: dvoufázové a jednofázové. Diskutovali jsme o vývoji modelů od R-CNN po YOLOv8, přičemž jsme zvláště vyzdvihli YOLO modely pro jejich rychlosť a efektivitu. Model YOLOv8 byl identifikován jako nejlepší a vybrán pro další experimenty a nalezení optimálního použití. Konkrétně model YOLOv8s (varianta „s“) byl vybrán kvůli optimálnímu poměru rychlosti a přesnosti.

Pro klasifikaci obrazu jsme se zaměřili na hluboké učení a diskutovali o různých architekturách CNN. Modely byly analyzovány z hlediska jejich výkonu a schopnosti generalizace. Byla představena novější řešení jako EfficientNet, Vision Transformer nebo YOLOv8s-CLS, která nabízejí pokročilé možnosti pro klasifikaci obrazu s vysokou přesností a efektivitou a proto jsou dále podrobeny experimentům pro ověření jeho výkonnosti v praxi.

# 4 Datasetsy

V této kapitole poskytujeme podrobný přehled a hodnocení datasetů, které byly využity pro trénink a validaci modelů v rámci vývoje našeho systému. Pečlivý výběr datasetů je nezbytný, neboť úspěch a efektivita modelů strojového učení závisí na kvalitě použitých dat. Pro naše účely datasetsy dělíme do tří hlavních kategorií: *vlastní*, *cizí* a *syntetické*. V následujících odstavcích se zaměříme na každou z těchto kategorií zvlášt, představíme použité datasetsy, vysvětlíme jejich významné přednosti a objasníme, proč byly zvoleny v kontextu specifických požadavků našeho systému.

## 4.1 Vlastní datasetsy

V rámci naší práce jsme vytvořili čtyři vlastní datasetsy, jejichž cílem bylo řešit specifické potřeby a výzvy, s nimiž se náš systém setkává a pro něž neexistují žádné vhodné veřejně dostupné varianty. Rozsah těchto vlastních datasetů není příliš velký z důvodu časové náročnosti jejich vytváření. Obzvláště zajímavým zdrojem pro sběr dat bylo API portálu <https://api.windy.com/>, odkud jsme čerpali obrázky. Tento zdroj je schopný poskytovat aktuální a historická meteorologická data v podobě, která přímo odpovídá našim potřebám. Dále se věnujeme podrobnějším informace o každém z datasetů.

### 4.1.1 Dataset pro klasifikaci scenerie

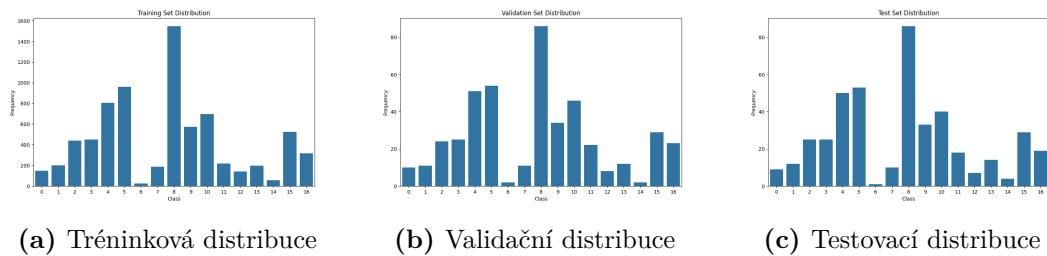
Tento dataset je víceštítkový (multilabelový), což znamená, že umožňuje vzorkům náležet do více kategorií současně, na rozdíl od jednoštítkové (singlelabel) klasifikace, kde vzorek patří pouze do jedné kategorie. Obsahuje snímky různých prostředí s anotacemi, přičemž jeho primáním účelem je poskytnout modelům základ pro efektivní rozpoznávání a klasifikaci typů prostředí na základě vizuálních charakteristik. K dispozici jsou specifikované kategorie zahrnující *letiště*, *pláž*, *budovu*, *město*, *pobřeží*, *les*, *interiér*, *jezero*, *krajinu*, *meteo*, *horu*, *přístav*, *řeku*, *sportovní areál*, *náměstí*, *dopravu* a *vesnici*.

Dataset obsahuje celkem 2 899 anotovaných snímků. Některé kategorie se v datasetu vyskytují jen ojediněle a dataset není vyvážený, což může mít vliv na trénování a výkonnost modelů. Zde je výčet zastoupení anotací v jednotlivých kategoriích:

- letiště: 168
- pláž: 224
- budova: 489
- město: 499
- pobřeží: 908
- les: 1066

- interiér: 27
- jezero: 208
- krajina: 1720
- meteo: 639
- hora: 782
- přístav: 258
- řeka: 157
- sportovní areál: 223
- náměstí: 62
- doprava: 582
- vesnice: 359

Při rozdělení datové sady na trénovací, validační a testovací množiny se uplatnil poměr 80:10:10. Zásadním prvkem tohoto procesu bylo zajištění konzistentní distribuce kategorií napříč všemi množinami (viz Obrázek 4.1).



**Obrázek 4.1** Distribuce datových množin datasetu na klasifikaci scenerie

### 4.1.2 Dataset pro klasifikaci počasí

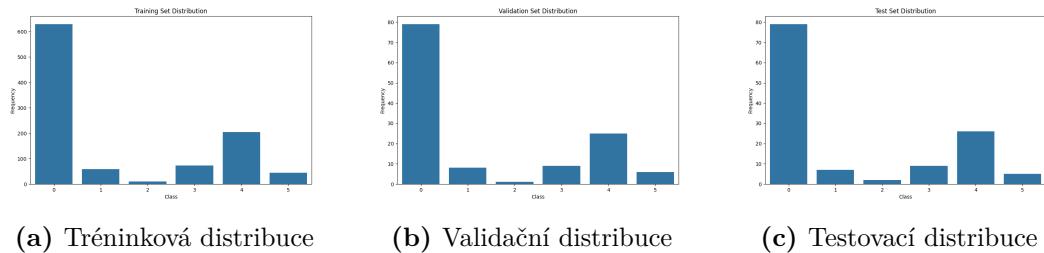
Dataset je určen pro jednoštítkovou klasifikaci a obsahuje celkem 1 277 anotovaných snímků, které reprezentují různé druhy počasí. Každý snímek je přiřazen do jedné z následujících kategorií: *zataženo*, *mlha*, *déšť*, *sníh*, *slunečno*, *neznámé*, což umožňuje modelům přesně identifikovat a klasifikovat aktuální meteorologické podmínky. Zastoupení jednotlivých kategorií v datasetu je následující:

- zataženo: 787
- mlha: 74
- déšť: 13
- sněživo: 91
- slunečno: 256

- neznámé: 56

Distribuce zastoupení opět naznačuje, že dataset není vyvážený, což je opět důležité vzít v úvahu při návrhu a trénování modelů, aby se zabránilo případným zkreslením.

Při strukturování datasetu do trénovacích, validačních a testovacích částí byl zachován poměr 80:10:10, s cílem zajistit evaluaci modelů na datových sadách nezávislých od tréninkové. Důležitým faktorem je rovněž dosažení homogenního zastoupení jednotlivých kategorií v každé části datasetu. Tento přístup podporuje nejen spravedlivé hodnocení výkonnosti modelů, ale i jejich robustnost a přesnost při generalizaci na nová data. Podrobnosti distribuce jsou vizualizovány v doplňujícím obrázku, což umožňuje přímé srovnání s předchozím datasetem, kde byl uplatněn obdobný princip zajištění konzistence distribuce kategorií (viz Obrázek 4.2).



**Obrázek 4.2** Distribuce datových množin datasetu na klasifikaci počasí

#### 4.1.3 Dataset pro detekci oblohy

Dataset se zaměřuje na detekci oblohy v rámci obrazových snímků z webových kamer. Skládá se z 2799 anotovaných snímků, které byly pečlivě vybrány s cílem ochránit rámeček oblast oblohy, který slouží pro určení, zda se jedná o venkovní fotografii, ze které je možné odvodit aktuální meteorologické podmínky. Anotace byla provedena poloautomaticky s následovnou manuální validací a úpravou. Rozdělení datasetu na trénovací, validační a testovací množiny bylo provedeno s poměrem 80:10:10.

#### 4.1.4 Dataset pro detekci slunce

Dataset se zaměřuje na detekci slunce v obrazových snímcích. Obsahuje celkem 818 anotovaných snímků, z čehož 165 snímků pochází z datasetu dostupného na platformě RoboFlow pod názvem *Sun*<sup>1</sup> a 196 snímků je ze souboru *Sun Tracking Photovoltaic Panel*<sup>2</sup>. Zbývající snímkы jsou výsledkem našeho vlastního sběru a anotace. Použité datasety z RoboFlow byly sloučeny dohromady a následně rozděleny na trénovací, validační a testovací množinu v poměru 80:10:10.

<sup>1</sup>Zdroj datasetu Sun: <https://universe.roboflow.com/1009727588-qq-com/sun-nxvfz/dataset/2>

<sup>2</sup>Zdroj datasetu *Sun Tracking Photovoltaic Panel*: <https://universe.roboflow.com/pfe-iset-beja/sun-tracking-photovoltaic-panel/dataset/2>

## 4.2 Cizí datasety

V této práci využíváme širokou škálu cizích datasetů určených pro detekci objektů, textu a specifických vizuálních charakteristik, jako je detekce nahoty, obličejů nebo státních poznávacích značek vozidel. Pro zachování konzistence ve vývoji a evaluaci našich modelů strojového učení, aplikujeme vlastní rozdělení datasetů v poměru 80:10:10 na trénovací, validační a testovací.

### 4.2.1 MSCOCO 2017 dataset

MSCOCO 2017 dataset Lin et al. [20] obsahuje 123 287 obrázků, 80 kategorií a 5 041 096 anotací, které zahrnují různé objekty, scény a situace z každodenního života (viz Tabulka 4.1). Tyto kategorie lze rozdělit do několika hlavních skupin zahrnujících zvířata, vozidla, domácí předměty, sportovní vybavení, jídlo a venkovní objekty. Některé z klíčových kategorií v MSCOCO 2017 datasetu jsou:

- **Zvířata:** pes, kočka, kůň, ovce, kráva, slon, medvěd, zebra, žirafa.
- **Vozidla:** automobil, motocykl, letadlo, autobus, vlak, nákladní auto, lod, semafor, hydrant.
- **Domácí předměty:** stůl, židle, pohovka, lůžko, jídelní stůl, toaleta, TV, notebook, myš, dálkový ovladač, klávesnice, mobilní telefon.
- **Sportovní vybavení:** létající talíř, lyže, snowboard, sportovní míč, létací drak, baseballová pálka, baseballová rukavice, skateboard, surf, tenisová raketa.
- **Jídlo:** banán, jablko, sendvič, pomeranč, brokolice, mrkev, hotdog, pizza, dort, židle.
- **Venkovní objekty:** strom, lavička, parkování, semafor, hydrant, značka stop, parkovací automat.

Použití MSCOCO 2017 datasetu umožňuje efektivně filtrovat anotované osoby a vozidla, které jsou klíčové v oblasti bezpečnosti a ochrany soukromí. MSCOCO 2017 dataset je k dispozici ke stažení na <https://cocodataset.org/>.

### 4.2.2 COCO-Text

COCO-Text je specializovaný dataset zaměřený na text v obrázcích postavený na obrázcích z MSCOCO 2014. Byl vytvořen s cílem poskytnout rozsáhlou sadu

Část dat	Obrázky	Anotace
Trénink	115 367	4 013 946
Validace	5 000	504 509
Test	2 920	522 641

Tabulka 4.1 Počty obrázků a anotací v MSCOCO 2017 datasetu

anotací textu v různých kontextech a situacích v přirozených obrázcích. Námi použitá podmnožina obsahuje 17 414 snímků.

COCO-Text dataset je využíván pro filtrace log, reklam a obecně velkých textů ve vizuálních médiích, což přispívá k čistotě vizuálního obsahu. Zdrojem datasetu je web Kaggle<sup>3</sup>, který zprostředkovává dataset původního tvůrce <https://bgshih.github.io/cocotext/>.

#### 4.2.3 SUN 2012 dataset

SUN 2012 dataset je dataset určený pro úlohy detekce objektů, který poskytuje rozsáhlé množství anotovaných obrázků zahrnující široký rozsah scén a objektů nalezených v přirozeném prostředí. Tento dataset je známý svou obrovskou diverzitou obsahující tisíce kategorií objektů.

Pro potřeby našeho projektu jsme z SUN 2012 datasetu vybrali 70 kategorií, které jsou nejrelevantnější pro naši aplikaci. Tyto kategorie zahrnují:

*zed, okno, podlaha, obloha, stromy, budova, strom, rostlina, rostliny, tráva, cesta, budovy, hora, auto, osoba, krabice, země, chodník, dopravní značka, květiny, plot, mrakodrap, pouliční lampa, mořská voda, dům, zábradlí, voda, obrazovka, keré, kámen, balkón, deska, text, krabice, pole, skála, palma, skály, stezka, lavička, zasněžená hora, říční voda, živý plot, kameny, reflektor, brána, pozemek, auta, skalnatá hora, písečná pláž, stojací lampa, lidé, deska, most, sníh, krb, kulečníkový stůl, letadlo, písek, vodopád, věž, zasněžená zem, přechod, domy, fontána, věšák, bazén, hory, zamračená obloha, palmy.*

Velikost dat v SUN 2012 datasetu pro naše využití obsahující výše uvedené objekty je 16 781 snímků. Výběr SUN 2012 datasetu nám umožňuje lépe pochopit scénu a klasifikovat obrázky na základě přítomných objektů.

#### 4.2.4 Dataset pro detekci obličejů

Dataset pro detekci obličejů, který kombinuje data z dvou klíčových zdrojů – Kaggle<sup>4</sup> obsahujících 13 386 trénovacích a 3 347 validačních obrázků a Roboflow<sup>5</sup> obsahujících 2 211 trénovacích obrázků. Celkově jsme pro naše účely použili 18 944 snímků. Dataset pro detekci obličejů je využíván k identifikaci a anonymizaci obličejů ve vizuálních datech, což je zásadní pro ochranu soukromí jednotlivců.

#### 4.2.5 Dataset pro detekci státních poznávacích značek vozidel

Dataset zaměřený na detekci registračních značek obsahuje celkově 24 242 anotovaných snímků. Tento dataset je důležitý pro detekci a případné zamazávání státních poznávacích značek vozidel s cílem ochrany osobních údajů a soukromí

<sup>3</sup>Zdroj datasetu z Kaggle: <https://www.kaggle.com/datasets/c7934597/cocotext-v20?select=data>

<sup>4</sup>Zdroj dat z Kaggle: <https://www.kaggle.com/datasets/fareselmenshawii/face-detection-dataset>

<sup>5</sup>Zdroj dat z Roboflow: <https://universe.roboflow.com/mohamed-traore-2ekkp/face-detection-mik1i/dataset/7>

v digitálním prostředí. Data byla získána z RoboFlow datasetu *License Plate Recognition*<sup>6</sup>.

#### 4.2.6 Dataset pro detekci nahoty

Pro účely detekce nahoty a identifikace nevhodného obsahu byl vybrán dataset *PORN LAB Image Dataset* z platformy Roboflow, který obsahuje 13 043 anotovaných obrázků s již provedenými augmentacemi<sup>7 8</sup>. Tento dataset je hodnotný pro jeho rozmanitost a specifické kategorizace, které zahrnují obnažené části těla: *břicho, ženské prsy, mužské prsy, zadnice, ženské genitálie a mužské genitálie*, což ho činí ideálním pro filtraci nevhodného obsahu, které je klíčové pro udržení standardů vhodnosti a etiky v digitálním obsahu.

### 4.3 Syntetické datasety

Syntetické datasety jsou uměle vytvořené datové sady, které napodobují reálná data a umožňují simulaci specifických situací bez nutnosti sběru reálných dat. Vzhledem k výzvám spojeným s nedostatkem specificky anotovaných datasetů a vylepšení stávajících datasetů, jsme využili také syntetické datasety. Ty umožňují simulovat konkrétní situace a podmínky, které by jinak byly obtížně získatelné. Také umožňuje testovat, jak tato generovaná data ovlivňují výkonnost a spolehlivost našich modelů. Hlavním nástrojem pro tvorbu těchto syntetických datasetů se stala platforma DataDreamer od společnosti Luxonix Sokovnin et al. [27].

#### 4.3.1 DataDreamer

DataDreamer je platforma skládající se z několika vzájemně propojených komponent, které poskytují end-to-end řešení pro generování a anotaci syntetických dat pro detekci objektů. Jednotlivé kroky vytváření syntetického datasetu tvořeného generovanými obrázky a anotacemi jsou popsány níže.

1. **Generátor promptů:** Na základě vstupní specifikace vytvoří generátor promptů detailní popisy pro následné generování obrázků. Zvolili jsme pokročilý jazykový model (LM), který je schopen na základě předdefinovaných specifikací vytvářet detailní a koherentní textové popisy.
2. **Generování obrázků:** Modelů generování obrázků vytvoří vysoko kvalitní a realistické obrázky na základě textových popisů. Zvolili jsme SDXL Turbo model<sup>9</sup> využívající state-of-the-art technologie difuzních modelů pro transformaci textových popisů na vizuální data. Díky optimalizacím pro vysokou

<sup>6</sup>Zdroj dat z Roboflow: <https://universe.roboflow.com/roboflow-universe-projects/license-plate-recognition-rxg4e/dataset/4>

<sup>7</sup>Augmentace je technika ve strojovém učení, která uměle zvyšuje velikost a diverzitu tréninkových dat tím, že aplikuje náhodné a realistické transformace na stávající vzorky (např. obrázky), čímž zlepšuje robustnost a výkon modelu.

<sup>8</sup>Zdroj dat z Roboflow: [https://universe.roboflow.com/tiem-yhrf6/porn\\_lab/dataset/1](https://universe.roboflow.com/tiem-yhrf6/porn_lab/dataset/1)

<sup>9</sup>SDXL Turbo byl představený společností Stability.AI v jejich článku: <https://stability.ai/research/adversarial-diffusion-distillation>

rychlost a efektivitu je tento model schopen rychle generovat vysoko kvalitní a realistické obrázky.

3. **Anotace obrázků:** Následuje proces automatické anotace obrázků pomocí zero-shot modelů na detekci objektů. Anotování obrázků v našem případě zajišťuje pokročilý zero-shot detektor objektů od Google OWLv2 Minderer et al. [28], který je schopen identifikovat a klasifikovat objekty v generovaných obrázcích bez potřeby předchozího specifického tréninku.
4. **Validace a testování:** Na závěr může volitelně proběhnout validace a testování vygenerovaných anotovaných obrázků.

## Vytvořené datasety z DataDreameru

Pro naše účely jsme pomocí DataDreameru vytvořili dva syntetické datasety. První obsahuje 10 000 snímků se širokou škálou objektů, které se vyskytují napříč všemi zmíněnými datasety a které chceme detektovat. Druhý obsahuje dalších 10 000 snímků se stejnými kategoriemi jako SUN 2012 a byl speciálně generovaný pro jeho doplnění, což bylo motivováno snahou kompenzovat některé nevýhody SUN 2012 datasetu, zejména počtu kategorií a celkového objemu dat.

DataDreamer jsme upravili tak, aby reflektoval různorodost reálného světa. Při generování popisů pro snímky (prompts) jsme použili uniformní distribuce pro denní a noční čas (*in daytime* a *in nighttime*) a různé podmínky počasí (*jasno, polojasno, slunečno, zataženo, mlhavo a deštivo a sněživo*) pro simulaci různých scénářů, ve kterých mohou venkovní kamery operovat. Cílem bylo získat data, která nejvíce připomínají data z venkovních webových kamerových systémů.

## Popis generovaných scén syntetických snímků

Proces formulace popisu scény pro tvorbu syntetických snímků byl navržen tak, aby co nejvíce reflektoval skutečné podmínky, ve kterých mohou být venkovní kamery používány. Klíčové aspekty tohoto procesu zahrnují výběr meteorologických podmínek a denní doby, což jsou dva základní faktory, které významně ovlivňují vizuální charakteristiky snímků.

Meteorologické podmínky byly definovány škálou možností: *jasno, slunečno, oblačno, částečně zataženo, deštivo, mlhavo*. Denní doba byla rovněž rozdělena na dvě základní kategorie: *den a noc*.

Na základě specifikovaných kritérií a selekce 3 až 8 objektů určených pro detekci v daném datasetu je připraveno zadání pro LM pro každou generovanou scénu. Tato specifikace slouží jako instrukce pro LM, který poté generuje detailní popis scény určený generátoru obrázků. Formulace zadání pro LM je následující: „*Fotografie {objektů} během {denní doby} za {meteorologických podmínek}*“. V této šabloně, *objekty* reprezentují seznam objektů na fotografií s počtem rozpětí mezi 3 a 8. Jak *denní doba*, tak *meteorologické podmínky* jsou stanoveny na základě předem určeného výběru. Tato metodologie generace scén byla vyvinuta za účelem simulace co nejvhodnějších podmínek pro vytváření syntetických snímků.

### 4.3.2 Dataset pro detekci snímků bez relevantního obsahu

Snímkы bez relevantního obsahu můžeme definovat jako obrazy, které neobsahují žádné rozpoznatelné nebo užitečné objekty či informace pro danou aplikaci. Typicky se jedná o snímkы s vysokou mírou uniformity (např. jednobarevné plochy) nebo snímkы obsahující pouze generický text či symboly signalizující chybu či nedostupnost (např. „obrázek není k dispozici“). Identifikace těchto snímků vyžaduje analýzu vizuálních charakteristik, jako jsou barevné distribuce a textové informace.

Pro vytvoření syntetického datasetu snímků bez relevantního obsahu jsme vyvinuli generátor, který umožňuje vytváření obrázků s různými atributy, jako jsou náhodné barvy, texty a jednoduché geometrické tvary. Tyto atributy jsou kritické pro simulaci typických znaků nežádoucích snímků. Důležité vlastnosti generovaných snímků zahrnují:

- **Uniformní barevné plochy:** Reprezentují snímkы bez vizuálního obsahu, například snímkы, kde je obrazový senzor zakrytý nebo kamera je vypnuta.
- **Textové zprávy:** Simulují scénáře, kdy je na snímkу zobrazena chybová zpráva nebo upozornění, typicky indikující nedostupnost obrazového obsahu.
- **Jednoduché geometrické tvary a šum:** Modelují nízkou komplexitu obrazu nebo sníženou kvalitu obrazu, což může být důsledkem technických problémů.

Pro potřeby trénování detekčního modelu generujeme 500 obrázků bez relevantního obsahu a doplňujeme je 500 obrázky pocházejících z datasetu na klasifikaci počasí.

### 4.3.3 Dataset pro detekci částečně stažených snímků

Tento dataset vytváříme modifikací existujících snímků opět pocházejících z datasetu na klasifikaci počasí tak, aby vypadaly jako částečně stažené. Tento proces zahrnuje aplikaci *vkládání jednolitých barevných bloků*. Tento přístup simuluje případy, kde místo očekávaného obrazového obsahu zůstávají prázdné, často jednobarevné oblasti v černém, bílém a nebo šedém odstínu.

Pro zvýšení reálnosti datasetu a zajištění robustnosti modelů implementujeme náhodnou variabilitu v míře a barvě jednolitých barevných bloků, umožňující tak vytvoření širokého spektra částečně stažených snímků.

Opět používáme 500 obrázků, které modifikujeme, aby vypadaly jako částečně stažené a k nim přidáváme 500 obrázků bez poškození (všechny pocházejí z datasetu pro klasifikaci počasí).

### 4.3.4 Dataset pro detekci vertikálně poškozených snímků

Pro vytvoření efektu vertikálně poškozených snímků upravujeme snímkы tak, že do nich vkládáme jednotné bloky počínaje náhodným bodem v obrázku. Tato technika efektivně imituje chyby, které se vyskytují při nesprávném načítání nebo přenosu obrázků, když určité části snímkу nejsou korektně zobrazeny a jsou nahrazeny oblastmi bez vizuálního obsahu nebo s uniformní barvou.

Dataset obsahuje celkem 1000 snímků, kde 500 obrázků bylo modifikováno s cílem simuloval vertikálně poškozené snímky. Zbývajících 500 snímků zůstává nezměněných a slouží jako kontrolní skupina. Všechny snímky pocházejí z původního datasetu určeného pro klasifikaci počasí.

## 4.4 Shrnutí

V této kapitole jsme představili datasety využité pro trénink a validaci modelů, které jsou základem našeho systému. Výběr a struktura datasetů jsou klíčové pro výkonnost modelů, což se odráží ve třech hlavních kategoriích datasetů: *vlastních, cizích a syntetických*.

Vlastní datasety byly vyvinuty pro konkrétní potřeby, pro které neexistují veřejné datasety. Příkladem je „Dataset pro klasifikaci scenerie“ a „Dataset pro klasifikaci počasí“ obsahující specifické kategorie. Tyto datasety byly rozšířeny o „Dataset pro detekci oblohy“ a „Dataset pro detekci slunce“.

Cizí datasety jsou využívány pro širší rozsah detekce a klasifikace objektů a situací. Významné jsou MSCOCO 2017 a SUN 2012, které poskytují širokou paletu vizuálních anotací z reálného světa. Mezi další více specifické datasety pak patří COCO-Text dataset zaměřený na detekci textů, „Dataset pro detekci obličejů“, „Dataset pro detekci státních poznávacích značek vozidel“ a „PORN LAB Image Dataset“ využitelný pro detekci nahoty.

Syntetické datasety, vyvinuté s pomocí platformy DataDreamer a dalších nástrojů, umožňují simulaci specifických scénářů, které nejsou běžně dostupné v reálných datasetech. Tyto datasety jsou klíčové pro testování a zlepšení robustnosti modelů v kontrolovaném prostředí, které napodobuje reálné podmínky s přesně definovanými parametry. Syntetické datasety také pomáhají překonat omezení reálných dat tím, že umožňují experimentaci s různými situacemi a podmínkami, které mohou být obtížně získatelné nebo vzácné ve skutečném světě.

Celkově tato kapitola zdůrazňuje strategický výběr a strukturování datasetů, které jsou základem pro úspěšný vývoj a implementaci našich algoritmů strojového učení. Kombinace vlastních, cizích a syntetických datasetů zajišťuje, že naše modely jsou dobře vybaveny pro řešení širokého spektra úloh ve vizuálním rozpoznávání s vysokou přesností a spolehlivostí.

# 5 Experiments

V této kapitole poskytujeme přehled a analýzu realizovaných experimentů. Experimenty jsou zaměřeny na nalezení neoptimálnějších metod pro zpracování obrazu z webových kamer. Cílem je určit efektivitu různých přístupů ve čtyřech základních oblastech: *validace obrazu*, *detekce objektů*, *klasifikace počasí* a *klasifikace scenérie*. Výsledky experimentů odhalují limitace a účinnosti jednotlivých vyzkoušených metod, na jejichž základě je vybráno nejlepší řešení pro náš systém.

## 5.1 Experiments s CV validátory pro získání potřebných prahů

V této části provádíme experimenty s validátory obrazu za účelem nastavení prahových hodnot pro detekci poškozených a nerelevantních snímků.

### 5.1.1 Experiments pro určení prahových hodnot pro algoritmus detekce fotek bez relevantního obsahu

Proces nalezení prahových hodnot pro algoritmus VoidSeeker (viz Kapitola 3.1.1) zakladáme na statistických metodách a důkladné analýze dat. K určení prahových hodnot a testování používáme celkem 1000 fotografií datasetu pro detekci snímků bez relevantního obsahu (viz Kapitola 4.3.2). Fotografie jsou rozdělené rovnoměrně na snímky s a bez relevantního obsahu. Prvních 800 fotografií používáme k tréninku (určení prahových hodnot), zatímco zbývajících 200 fotografií slouží k evaluaci výsledků. Postupujeme ve dvou základních fázích, kterými jsou *výpočet metrik* a *určení optimálních prahových hodnot*.

V první fázi z datové sady rozdelené na snímky s a bez relevantního obsahu počítáme dvě klíčové metriky - standardní odchylku a procentuální podíl hran. Tyto metriky vybíráme na základě jejich schopnosti kvantifikovat variabilitu a strukturální složitost obrazu, což jsou hlavní faktory pro rozlišení obrazu s a bez relevantního obsahu.

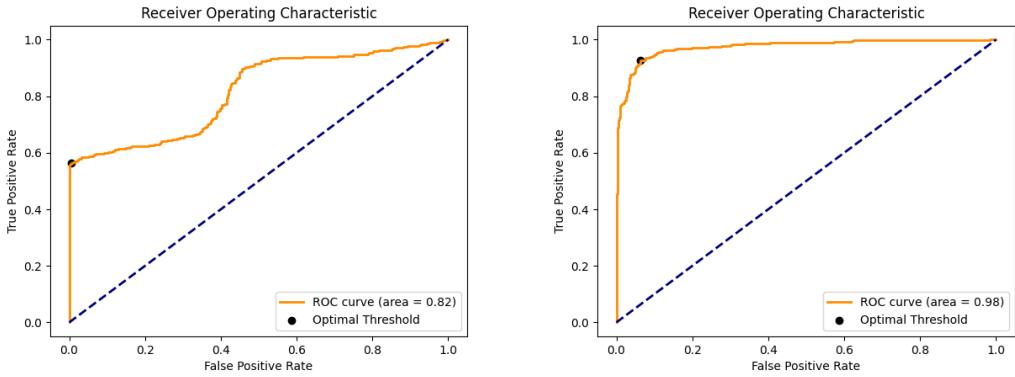
Následně se zaměřujeme na analýzu ROC (Receiver Operating Characteristic) křivky pro každou z těchto metrik s cílem určit optimální prahové hodnoty. Tento krok je založen na vyhledání bodu na křivce, kde je rozdíl mezi pravdivě pozitivní mírou a falešně pozitivní mírou maximální. Využití analýzy ROC a určení AUC<sup>1</sup> (viz Obrázek 5.1) jako standardních nástrojů v oblasti strojového učení nám umožňuje efektivně hodnotit výkonnost klasifikačních modelů a objektivně porovnávat různá nastavení prahových hodnot.

Díky této metodice identifikujeme optimální horní prahové hodnoty pro:

- standardní odchylku: **35**
- procentuální zastoupení hran: **0.12**

---

<sup>1</sup>AUC, neboli plocha pod křivkou, kvantifikuje celkový výkon binárního klasifikátoru s hodnotou mezi 0 a 1, kde 1 značí dokonalou předpověď a 0,5 znamená žádnou diskriminační sílu.



(a) Graf ROC pro určení prahu hranového za- (b) Graf ROC pro určení prahu standardní stoupení. odchylky.

**Obrázek 5.1** Grafy ROC pro detekci obrázků bez relevantního obsahu.

Implementace těchto prahů do algoritmu nám umožňuje dosáhnout přesnosti 94.5% v detekci snímků bez relevantního obsahu. Vysoká úroveň přesnosti s minimálním počtem falešně pozitivních (9 z 200) a falešně negativních (2 z 200) výsledků potvrzuje, že naše kombinace výpočtu metrik a statistické analýzy poskytuje solidní základ pro efektivní detekci.

### 5.1.2 Experimenty pro určení prahových hodnot pro algoritmus detekce částečně stažených fotek

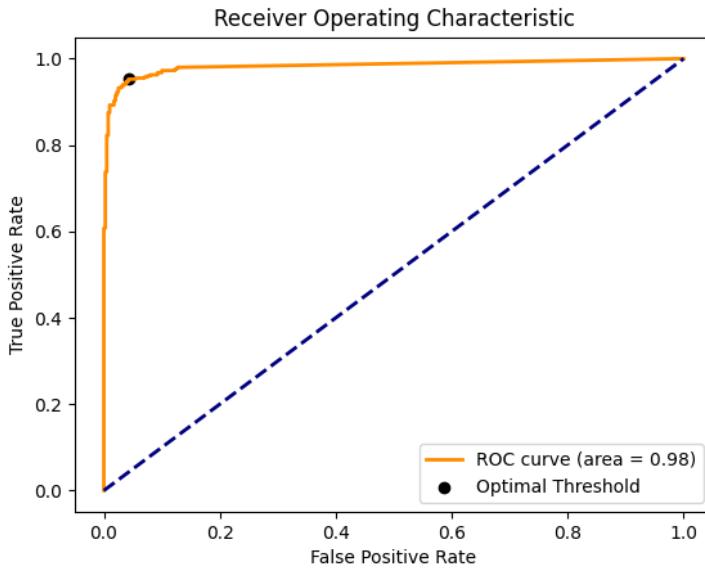
V navazujícím experimentu se zaměřujeme na určení prahové hodnoty normalizované standardní odchylky pixelových intenzit pod detekovanou hranicí algoritmu FragGuard (viz Kapitola 3.1.2). Podobně jako v předchozím případě využíváme pro určení a evaluaci prahových hodnot dataset pro detekci částečně stažených snímků (viz Kapitola 4.3.3) o celkovém počtu 1000 fotografií, rozdělený na tréninkovou (800 fotografií) a testovací (200 fotografií) sadu, obě s rovným zastoupením plně a částečně stažených fotek.

V procesu určování optimální prahové hodnoty normalizované směrodatné odchylky pixelových intenzit opět využíváme analýzu ROC křivky (viz Obrázek 5.2). Na základě této metody určujeme prahovou hodnotu jako **0.052**, která nám umožňuje efektivně odlišit částečně stažené fotky od kompletne stažených.

Prahovou hodnotu volíme na základě maximalizování úspěšnosti detekce při současném minimalizování počtu falešně pozitivních a falešně negativních výsledků. Testování algoritmu na sadě 200 fotografií s tímto nastavením prahu přináší výsledky s 96% úspěšností, přičemž zaznamenáváme 4 falešně pozitivní a 4 falešně negativní výsledky. Tento výsledek ukazuje na vysokou spolehlivost a přesnost našeho přístupu k detekci částečně stažených snímků.

### 5.1.3 Experimenty pro nalezení nejlepšího klasifikátoru do algoritmu na detekci vertikálně poškozených obrázků

V rámci našeho výzkumu zaměřeného na detekci vertikálně poškozených obrázků čelíme výzvě určit, jak nejlépe klasifikovat fotografie na základě extrahova-



**Obrázek 5.2** Graf ROC pro určení prahů z testovací množiny na detekci částečně stažených obrázků.

ných vlastností pomocí námi vyvinutého algoritmu VertiScan (viz Kapitola 3.1.3). Tyto vlastnosti zahrnují statistiky jako jsou průměr, standardní odchylka, šíkmost a špičatost rozložení standardních odchylek jasových hodnot, doplněné o index maximálního poklesu a průměrnou diferenci.

Jedním z hlavních zjištění je, že pro tuto specifickou úlohu nelze efektivně využít prahové hodnoty, jak to je u předešlých algoritmů. Důvodem je variabilita a komplexnost vzorů vertikálního poškození, které nelze adekvátně zachytit pouze jednoduchými prahovými pravidly. Úloha vyžaduje sofistikovanější přístup k modelování vzájemných vztahů mezi extrahovanými vlastnostmi, a proto dále experimentujeme s klasifikačními algoritmy SVM s lineárním jádrem a XGBoost, které volíme na základě jejich schopnosti efektivně zpracovávat vysokodimenzionální data a identifikovat složité vzory v datech.

V experimentech používáme testovací množinu datasetu pro detekci vertikálně poškozených snímků (viz Kapitola 4.3.4) s celkovým počtem 1000 obrázků (poměr 50:50 poškozených a nepoškozených) a z toho 800 obrázků použitých pro trénink a zbylých 200 obrázků pro test. Rozdělení je zde stejné, jako v předchozích částech, aby se zajistila konzistence metodologických přístupů a umožnilo přímé srovnání výsledků.

Na základě experimentů, SVM s lineárním jádrem dosahuje úspěšnosti 83% na testovací množině. Při hledání nejlepší konfigurace pro XGBoost pomocí Grid Search techniky identifikujeme optimální parametry, které vedou k mírnému zlepšení výkonnosti, dosahující úspěšnosti 84% na testovací množině. Mezi vybrané parametry patří:

- colsample\_bytree: 0.7
- gamma: 0
- learning\_rate: 0.01

- max\_depth: 5
- n\_estimators: 300

Tato konfigurace dosahuje skóre 0.838 v průběhu cross-validace. I přesto, že XGBoost překonává SVM z hlediska úspěšnosti, volíme pro implementaci SVM s lineárním jádrem do našeho detekčního systému. Toto rozhodnutí je motivováno především rychlostí SVM, která je pro naše využití klíčová. XGBoost nabízí mírně lepší přesnost, vyžaduje ale výrazně delší dobu pro trénink a optimalizaci, což je v našem případě nevhodné.

## 5.2 Experimenty s YOLOv8s na optimální detekci objektů

V této části experimentů se zaměřujeme na evaluaci modelu YOLOv8s Jocher et al. [2] pro účely detekce objektů zajišťující efektivní filtraci a zpracování obrazu. Model vybíráme díky jeho schopnosti rychle a přesně identifikovat objekty v různých scénách. Experimenty jsou navrženy tak, aby otestovaly výkon modelu v simulovaných reálných podmínkách s důrazem na optimalizaci mezi rychlostí a přesností detekce. Cílem je ověřit, jak dobře dokáže YOLOv8s vyhovět požadavkům našeho systému, a identifikovat případné oblasti pro zlepšení.

### 5.2.1 Výběr a použití datasetů

Naše experimenty s YOLOv8s využívají širokou paletu datasetů, které zahrnují veškeré objekty, jenž jsou relevantní pro naši aplikaci. Používáme těchto 8 datasetů:

1. Dataset pro detekci objektů umožňující porozumění scenérie MSCOCO (označovaný dále jako **COCO**, viz Kapitola 4.2.1)
2. Dataset pro detekci obličejů (označovaný dále jako **FCE**, viz Kapitola 4.2.4)
3. Dataset pro detekci nahoty (označovaný dále jako **NUD**, viz Kapitola 4.2.6)
4. Dataset pro detekci státních poznávacích značek vozidel (označovaný dále jako **LPT**, viz Kapitola 4.2.5)
5. Dataset pro detekci objektů umožňující porozumění scenérie SUN (označovaný dále jako **SCU**, viz Kapitola 4.2.3)
6. Dataset pro detekci oblohy (označovaný dále jako **SKY**, viz Kapitola 4.1.3)
7. Dataset pro detekci slunce (označovaný dále jako **SUN**, viz Kapitola 4.1.4)
8. Dataset pro detekci textu (označovaný dále jako **TXT**, viz Kapitola 4.2.2)

Trénink modelů YOLOv8s provádíme vždy na trénovacích množinách jednotlivých datasetů zvlášt. Evaluace natrénovaných modelů pak probíhá pro každý dataset zvlášt na jeho testovací množině. Tento proces nám umožní získat detailní přehled o výkonnosti modelů v konkrétních detekčních úlohách a posoudit, jak dobře jsou modely schopné generalizovat naučené znalosti na nově viděná data.

## 5.2.2 Metodologie tréninku

Trénink YOLOv8s modelů realizujeme formou finetuningu s následujícími parametry:

- **Batch size:** 4, minimalizace paměťové náročnosti a zajištění efektivity tréninku.
- **Epochs: 150**, s možností předčasného ukončení tréninku (early stopping) po 20 epochách bez zlepšení.
- **Optimalizátor: Auto**, automatická volba mezi SGD, Adam a dalšími na základě výkonu na validační sadě.
- **Learning rate: 0.01**
- **Weight decay: 0.0005**, předcházení přetrénování tím, že penalizuje velké váhy.
- **Warm-up epochs: 3**, postupné zvyšování learning rate na začátku tréninku pro stabilizaci procesu.

Zbylé parametry zůstaly nastaveny ve výchozím stavu.

## Finetuning

Finetuning je technika v oblasti strojového učení, která se využívá k dotrénování předem natrénovaného modelu pro konkrétní úlohu nebo dataset. Tento proces se často používá v aplikacích hlubokého učení, kde trénování modelu od základů vyžaduje značné množství výpočetních zdrojů a dat. Finetuning umožňuje využít znalosti, které model získal během původního tréninku a aplikovat je na nový související problém. Mezi základní principy metody patří:

- **Předtrénovaný model:** Výchozím bodem je model, který byl předem natrénovan na rozsáhlém a obecném datasetu, například ImageNet pro obrazové úlohy.
- **Úprava architektury:** Často se mění poslední vrstvy modelu tak, aby odpovídaly počtu tříd nebo struktuře výstupu cílené úlohy.
- **Finetuning:** Proces zahrnuje další trénink modelu na menším specifickém datasetu s volitelnou úpravou architektury. Tento krok obvykle vyžaduje menší učící rychlosť a méně epoch, jelikož model již obsahuje relevantní porozumění.

Finetuning je dále specifikován pro modely s zamraženým backbonem, kde bylo explicitně uvedeno, že prvních 10 vrstev nebude během tréninku aktualizováno (freeze: 10). Backbone neuronové sítě představuje předtrénovanou konvoluční část sítě, která slouží jako základní stavební kámen pro specializovanější úkoly v oblasti hlubokého učení. Díky rozsáhlému tréninku na masivních datových sadách je backbone schopen extrahat robustní a obecné vizuální reprezentace z vstupních obrazů. Zamrazení modelu či vrstev je technika v transfer learningu, kde se určité parametry během tréninku nezmění, umožňující adaptaci předtrénovaného modelu na novou úlohu s uchováním relevantních rysů.

### 5.2.3 Metodologie evaluace

Evaluace modelů je založena na sadě metrik specifických pro úlohy detekce objektů, jenž zahrnují:

- **Precision (Přesnost):** Míra správně identifikovaných pozitivních příkladů z celkového počtu příkladů, které model označil jako pozitivní.

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \quad (5.1)$$

- **Recall (Úplnost):** Míra správně identifikovaných pozitivních příkladů z celkového počtu skutečných pozitivních příkladů v datech.

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \quad (5.2)$$

- **Mean Average Precision (mAP<sub>50</sub> a mAP<sub>50–95</sub>):** Průměrná přesnost detekce při prahu IoU (Intersection over Union)<sup>2</sup> 0.5 a průměr přes různé prahy IoU od 0.5 do 0.95 s krokem 0.05. mAP poskytuje komplexní hodnocení výkonnosti modelu v detekci objektů.

### 5.2.4 Průběh tréninku

Tréninkový proces zahrnuje aplikaci augmentačních technik pro zvýšení robustnosti modelu vůči různorodosti vstupních dat. Dále využívá techniku early stopping k prevenci přetrenování modelu. Sledování výkonnosti modelu na validační sadě pak umožňuje dynamicky upravovat learning rate a další hyperparametry pro optimalizaci tréninkového procesu.

V průběhu tréninku věnujeme pozornost snižování celkové loss funkce a specifickým aspektům detekce, jako je přesnost lokalizace a schopnost modelu rozlišovat mezi různými třídami objektů. Výsledkem je model schopný efektivně detekovat a klasifikovat objekty s vysokou přesností a úplností při zachování rychlosti inference.

### 5.2.5 Analýza a evaluace finetuningu YOLOv8s modelů

V naší studii se zaměřujeme na rozšířenou analýzu finetuningu a evaluace YOLOv8s modelů pro úlohy detekce objektů. Používáme model YOLOv8s s předtrénovanými váhami na rozsáhlém a diverzifikovaném datasetu COCO, který poskytuje solidní základ pro další finetuning na našich specifických datasetech. Dále popíšeme detailní přístupy k finetuningu.

---

<sup>2</sup>IoU je metrika používaná pro hodnocení přesnosti polohových predikcí objektů v oblasti počítačového vidění vyjadřující poměr překryvu mezi predikovaným a skutečným ohraničujícím boxem. IoU se vypočítá jako

$$IoU = \frac{\text{Plocha průniku predikovaného a skutečného boxu}}{\text{Plocha sjednocení predikovaného a skutečného boxu}} \quad (5.3)$$

## Finetuning na individuálních datasetech

Prvním krokem je finetuning osmi YOLOv8s modelů na osmi různých datasetech, viz Tabulka 5.1. Každý dataset obsahuje unikátní sady anotací, což vyžaduje specifickou adaptaci modelů pro maximální výkonnost. Tento přístup umožňuje modelům lépe rozpoznávat specifické objekty v daných datasetech, což se projevuje vysokými hodnotami precision a recall pro většinu datasetů.

Model	Obrazy	Instance	Precision	Recall	mAP <sub>50</sub>	mAP <sub>50–95</sub>
COCO*	-	-	-	-	-	0.449
FCE	1895	6619	0.942	0.889	0.952	0.737
NUD	1305	4089	0.979	0.975	0.991	0.853
LPT	2425	2717	0.992	0.979	0.990	0.880
SCU	1679	12432	0.585	0.487	0.536	0.395
SKY	281	281	0.986	0.989	0.994	0.977
SUN	79	128	0.950	0.609	0.756	0.643
TXT	1715	7783	0.861	0.827	0.899	0.677

**Tabulka 5.1** Evaluace specializovaných finetunovaných modelů YOLOv8. [\*] Model trénovaný a evaluovaný autory YOLOv8 modelu, kde je známá jen hodnota mAP<sub>50–95</sub>.

## Finetuning se zamraženým backbonem

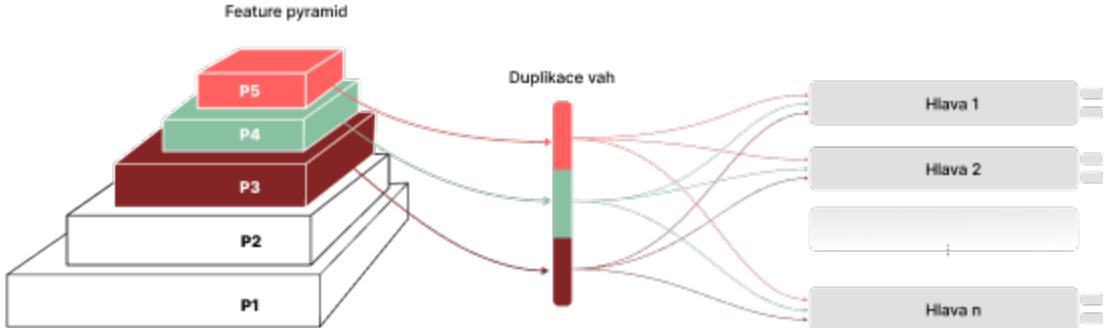
Při finetuningu se zamraženým backbonem pozorujeme obecné snížení metrik ve srovnání s plně finetunovanými modely (viz Tabulka 5.2). Tento výsledek je očekáván, jelikož omezení adaptace vah v backbone vrstvách snižuje schopnost modelu efektivně se přizpůsobit specifikům nových dat. Přesto tento přístup nabízí výhody v podobě rychlejšího tréninku a nižších požadavků na výpočetní zdroje.

Model	Obrazy	Instance	Precision	Recall	mAP <sub>50</sub>	mAP <sub>50–95</sub>
COCO*	-	-	-	-	-	0.449
FCE	1895	6619	0.916	0.851	0.931	0.694
NUD	1305	4089	0.886	0.782	0.880	0.585
LPT	2425	2717	0.982	0.965	0.988	0.785
SCU	1679	12432	0.456	0.429	0.442	0.318
SKY	281	281	0.955	0.992	0.989	0.904
SUN	79	128	0.978	0.609	0.728	0.626
TXT	1715	7783	0.761	0.673	0.761	0.504

**Tabulka 5.2** Evaluace specializovaných finetunovaných modelů YOLOv8 se zamraženým CNN backbonem. [\*] Model trénovaný a evaluovaný autory YOLOv8 modelu, kde je známá jen hodnota mAP<sub>50–95</sub>.

## Model s více specializovanými detekčními hlavami

Při vytváření našeho modelu YOLOv8 s více detekčními hlavami, které jsou optimalizovány pro rozličné úkoly detekce založené na různých datasetech, čerpáme



**Obrázek 5.3** Yolo multihead architektura (připojení na feature pyramid z backbonu).

inspiraci a technické poznatky ze článku *You Only Look at Once for Real-time and Generic Multi-Task* Wang et al. [29] a v rámci něj dostupného zdrojového kódu na GitHubu<sup>3</sup>. Tento zdroj nám poskytuje cenné vhledy do implementace a integrace více detekčních hlav do jednoho komplexního modelu. Tento model kombinuje trénované hlavy z modelů, které jsme předtím finetunovali se zamraženým backbonem předtrénovaným na MSCOCO datasetu, do jednoho víceúlohového modelu.

**Parametry a struktura** S přidáváním dalších detekčních hlav do modelu postupně narůstá počet vrstev a parametrů, což signalizuje zvýšení komplexnosti modelu. Konkrétně přecházíme od modelu s jednou hlavou až po model s osmi hlavami (z důvodu 8 specializovaných datasetů), což nám způsobuje postupný nárůst výpočetních požadavků (viz Tabulka 5.3). Architektura multihead modelu se skládá ze základního sdíleného backbone, který je společný pro všechny detekční hlavy a následně z individuálních vrstev hlav, které jsou specifické pro každý z datasetů (viz Obrázek 5.3).

Hlavy	Vrstvy	Parametry (M)	GFLOPs
1	225	11.17	28.8
2	350	17.25	45.1
3	475	23.34	61.3
4	600	29.43	77.5
5	725	35.51	93.8
6	850	41.60	110.0
7	975	47.69	126.2
8	1100	53.77	142.5

**Tabulka 5.3** Parametry modelu YOLOv8 s 1-8 specializovanými detekčními hlavami

**Výkonnost a doba inference** Multihead model je efektivnější ve srovnání s osminásobkem doby inference jednohlavového modelu díky simultánní detekci pro více úkolů v jednom průchodu obrazem, což je časově výhodnější než sekvenční

<sup>3</sup>YOLOv8 multi-task dostupný na Githubu <https://github.com/JiayuanWang-JW/YOLOv8-multi-task> a popsaný ve článku Wang et al. [29]

spouštění osmi samostatných modelů. Výkonnost modelů s 1-8 hlavami přináší očekávané výsledky (viz Tabulka 5.4).

YOLOv8	Inference			Postprocessing			Total
	Hlavy	Mean	Min	Max	Mean	Min	Max
1	8.662	8.469	17.302	1.355	1.245	10.326	11.741
2	8.893	8.49	22.843	2.578	2.428	11.457	13.354
3	11.848	11.209	23.015	3.697	3.453	7.463	17.503
4	14.652	14.242	24.573	4.638	4.483	7.82	21.13
5	17.661	17.29	23.02	5.645	5.502	6.73	25.138
6	20.671	20.267	47.414	6.693	6.55	16.514	29.232
7	23.83	23.25	51.272	7.836	7.587	19.037	33.555
8	27.057	26.252	66.288	8.911	8.621	22.016	37.851

**Tabulka 5.4** Evaluace času v milisekundách potřebného pro predikci pro YOLOv8 model s 1-8 specializovanými detekčními hlavami

**Konstrukce multihead modelu** Náš vícehlavový YOLO model vzniká sloučením trénovaných hlav z modelů se zamraženým backbonem do jednoho modelu se zachováním váhy každé hlavy. Tento postup zajišťuje, že výkonnost jednotlivých detekčních hlav v multihead modelu zůstává konzistentní s výkonností modelů, které byly samostatně finetunovány, čímž zachováváme specializovanou schopnost detekce pro různé typy objektů.

### Finetuning na agregovaném datasetu

Trénink jediného modelu YOLOv8 na kombinovaném datasetu (viz Tabulka 5.5) byl koncipován jako pokus o vytvoření univerzálního modelu schopného detekovat širokou škálu objektů. Výsledky ukazují, že i přes obecně nižší výkonnost ve srovnání s modely finetunovanými na specifických datasetech tento přístup stále nabízí slibnou cestu pro aplikace, kde je potřeba rozpoznávat objekty z různých domén.

Model	Obrazy	Instance	Precision	Recall	mAP <sub>50</sub>	mAP <sub>50-95</sub>
COCO	496	3541	0.480	0.518	0.520	0.388
FCE	1895	6619	0.852	0.749	0.854	0.596
NUD	1305	4089	0.785	0.663	0.758	0.474
LPT	2425	2717	0.950	0.877	0.946	0.690
SCU	1679	12432	0.371	0.340	0.350	0.242
SKY	281	281	0.922	0.973	0.972	0.868
SUN	79	128	0.627	0.352	0.427	0.228
TXT	1715	7783	0.633	0.547	0.600	0.361

**Tabulka 5.5** Evaluace jediného finetunovaného modelu YOLOv8 trénovaných na spojeném datasetu skládajícím se ze všech datasetů používaných pro jednotlivé modely.

## Syntetický dataset

Využití syntetického datasetu (viz Kapitola 4.3.1) jako prevence zapomínání umožňuje modelům udržovat si schopnost rozpoznávat objekty napříč různými doménami. Následující experiment odhaluje, že i přes určité snížení výkonnosti v důsledku rozdílů mezi reálnými a syntetickými daty tento přístup přináší významný potenciál pro vývoj robustnějších modelů (viz Tabulka 5.6).

Model	Obrazy	Instance	Precision	Recall	mAP <sub>50</sub>	mAP <sub>50–95</sub>
COCO	496	3541	0.345	0.264	0.255	0.180
FCE	1895	6619	0.561	0.430	0.519	0.279
NUD	1305	4089	0.0273	0.0227	0.0154	0.00494
LPT	2425	2717	0.778	0.597	0.680	0.330
SCU	1679	12432	0.210	0.208	0.156	0.105
SKY	281	281	0.867	0.851	0.900	0.700
SUN	79	128	0.578	0.289	0.350	0.185
TXT	1715	7783	0.0735	0.00578	0.0374	0.0171

**Tabulka 5.6** Evaluace jediného finetunovaného modelu YOLOv8 trénovaného na syntetickém datasetu obsahující všechny detekované kategorie jako výše zmíněné speciálizované modely.

## Zeroshot YOLO-World model

Zeroshot YOLO-World model umožňuje rozpoznávat objekty bez potřeby předchozího explicitního tréninku na konkrétních datech. Ačkoliv výsledky (viz Tabulka 5.7) ukazují na významně nižší výkonnost ve srovnání s tradičně finetunovanými modely, tento přístup otevírá nové možnosti pro vývoj adaptabilních a flexibilních systémů detekce objektů.

Model	Obrazy	Instance	Precision	Recall	mAP <sub>50</sub>	mAP <sub>50–95</sub>
COCO*	-	-	-	-	0.522	0.377
FCE	1895	6619	0.544	0.0835	0.301	0.154
NUD	1305	4089	0.0055	0.0088	0.003	0.001
LPT	2425	2717	0.812	0.0206	0.415	0.284
SCU	1679	12432	0.216	0.316	0.186	0.131
SKY	281	281	0.916	0.541	0.739	0.645
SUN	79	128	0	0	0	0
TXT	1715	7783	0.0084	0.0005	0.0043	0.001

**Tabulka 5.7** Evaluace YOLO-World modelu na datasetech. [\*] Model trénovaný a evaluovaný autory YOLO-World modelu.

## 5.2.6 Analýza výsledků experimentů

**Vysoká efektivita finetunovaných modelů** Finetuning osmi separátních modelů YOLOv8s na specifických datasetech dosáhl ve většině případů vysoké

účinnosti, což dokazuje, že specifické přizpůsobení modelu pro určitou úlohu může výrazně zlepšit jeho schopnost správně identifikovat objekty. Výsledky navíc naznačují, že finetunované modely jsou schopné dobře generalizovat naučené znalosti na nově viděná data.

**Omezení zamražených modelů** Modely se zamraženým backbonem obecně dosáhly nižších metrik ve srovnání s plně finetunovanými modely, což bylo očekáváno, jelikož omezení adaptace vah v backbone vrstvách snižuje schopnost modelu efektivně se přizpůsobit specifikům nových dat.

**Časová efektivita Multihead modelu** Multihead model nabízí výrazné zlepšení efektivity zvláště v kontextu času inference, kde simulace detekce pro více úkolů v jednom průchodu obrazem přináší časové úspory ve srovnání se sekvenčním spouštěním několika samostatných modelů.

**Potenciál agregovaných a syntetických datasetů** Trénink modelu na agregovaném datasetu a využití syntetického datasetu ukazuje, že přestože tyto přístupy obecně dosahují nižší výkonnéosti ve srovnání s modely trénovanými na specifických datasetech, tak přinášejí významný potenciál pro aplikace vyžadující rozpoznávání objektů z různých domén.

**Zeroshot detekce - směr budoucího výzkumu** Zeroshot YOLO-World model představuje zajímavý přístup s potenciálem pro budoucí výzkum, i když jeho současná výkonnost je nižší ve srovnání s tradičními modely. Jeho schopnost detekovat objekty bez předchozího explicitního tréninku otevírá nové možnosti pro vývoj adaptabilních a flexibilních systémů detekce.

### 5.2.7 Shrnutí

Na základě našich experimentů a analýz volíme využití osmi separátních finetunovaných modelů YOLOv8s pro účely detekce objektů. Tento přístup poskytuje nejvyšší výkonnost pro specifické detekční úkoly a umožňuje detailnější přizpůsobení modelů pro konkrétní aplikace.

## 5.3 Experimenty na klasifikaci scenérie

Cílem těchto experimentů je vyhodnotit schopnost algoritmů strojového i hlubokého učení klasifikovat scenérii na základě obrázků. Zaměřujeme se na určení nejfektivnějších metod s nejlepším poměrem přesnosti a rychlosti predikce.

### 5.3.1 Metodologie

Pro evaluaci modelů ve víceštítkové klasifikaci jsou klíčové následující metriky:

- **Přesnost (Accuracy):** Definována jako poměr správně identifikovaných štítků k celkovému počtu štítků. Přesnost ve víceštítkovém kontextu není ideální metrikou, jelikož může být zavádějící v případě nerovnoměrně distribuovaných tříd, což je přesně náš případ.

$$Accuracy = \frac{TruePositives + TrueNegatives}{TotalPopulation} \quad (5.4)$$

- **F1 skóre:** F1 skóre je harmonický průměr přesnosti (precision) a úplnosti (recall). Existují dva přístupy k jeho výpočtu v víceštítkové klasifikaci:
  - **F1 Macro:** Vypočítá se průměrné F1 skóre přes všechny třídy, kde každé třídě je přiřazena stejná váha.

$$F1_{Macro} = \frac{2 * (Precision * Recall)}{(Precision + Recall)} \quad (5.5)$$

kde Precision a Recall jsou průměrované přes všechny třídy.

- **F1 Samples:** Průměr F1 skóre je vypočítán pro každý vzorek zvlášť, což lépe reflektuje distribuci štítků ve víceštítkovém scénáři.

$$F1_{Samples} = \frac{1}{N} * \sum_i 2 * \frac{Precision_i * Recall_i}{Precision_i + Recall_i} \quad (5.6)$$

pro i-tý vzorek a N je celkový počet vzorků.

### 5.3.2 Vývoj a přístup ke klasifikaci scenerie

Vývoj přístupu ke klasifikaci scenerie probíhá postupně s akcentem na integraci nejnovějších technologií a metodik ve strojovém učení, přičemž je kladen důraz na dosažení co nejlepšího kompromisu mezi přesností a rychlostí pro potřeby real-time aplikací.

Nejdříve vybíráme jako základní klasifikační model *EfficientNet B0* Tan a Le [24], což je nejmenší člen rodiny EfficientNet modelů vhodný pro real-time klasifikaci. EfficientNet B0, disponující 3.9 miliony parametrů, je optimalizován pro efektivitu, což umožňuje jeho použití i na zařízeních s omezenými výpočetními kapacitami. Trénink tohoto modelu realizujeme na vlastním víceštítkovém datasetu pro klasifikaci scenérie (viz Kapitola 4.1.1). Dataset navrhujeme tak, aby reflektoval specifické potřeby klasifikace scenerie a obsahoval relevantní anotace a rozložení tříd.

Následně zkoumáme možnost využití klasifikovaných objektů z modelu YOLOv8, který je předem natrénovaný na datasetu zaměřeném na porozumění scéně (viz Kapitola 4.2.3). Tento přístup je motivován skutečností, že scenerie lze často efektivně popsat právě množinou detekovaných objektů a jejich vztahy.

#### Předzpracování dat a extrakce příznaků pomocí detekovaných objektů

Pro každý obrázek v datasetu pomocí YOLOv8 predikujeme objekty, na jejichž základě vytváříme slovník relevantních objektů. Tento slovník slouží jako základ pro vytváření vektorů příznaků.

Pro každou fotografiu následně generujeme příznaky na základě výskytu objektů definovaných ve slovníku. Tento proces zahrnuje kroky, jako je identifikace přítomnosti objektů, normalizace jejich ohraňujících boxů vzhledem k velikosti obrázku a výpočet statistik (např. průměrných a standardních odchylek) pro

souřadnice a konfidenční skóre detekovaných objektů. V případě, že objekt ze slovníku ve fotografii není identifikován, jsou na příslušné pozici ve vektoru příznaků vloženy nulové hodnoty.

Nyní představíme detailní popis vytvoření dvou typů příznaků:

- **BB příznaky:** Pro každý detekovaný objekt vytváříme normalizované souřadnice jeho ohraničujících boxů vzhledem k celkovým rozměrům obrázku. Dále pro objekty stejné kategorie vypočítáváme průměrné hodnoty a standardní odchylky těchto normalizovaných souřadnic a konfidenční skóre. Výsledkem je vektor složený z těchto agregovaných statistik pro každý objekt ze statického slovníku.
- **OH příznaky:** Pro jednodušší reprezentaci používáme alternativní *one-hot encoding* přístup, kde přítomnost každého objektu ze slovníku v obrázku je označena 1 a nepřítomnost 0. Tato metoda poskytuje rychlý a přímý způsob reprezentace informací o výskytu objektů, ale nepřináší žádné informace o jejich umístění či jiných vlastnostech.

Výsledkem je, že každá scéna je popisována sadou lokalizovaných a kategorizovaných prvků.

## Architektura MLP modelů

Pro klasifikaci navrhujeme rozšířené verze klasického MLP, které nazýváme RMLP. Architekturu navrhujeme tak, aby zpracovávala bohatší a komplexnější příznaky z obrázků scenerie. Níže je popsána obecná architektura MLP a RMLP modelů:

### MLP pro OH příznaky

- **Vstupní vrstva:** Přijímá vektor one-hot příznaků reprezentující přítomnost či absenci specifických objektů detekovaných v obrázku.
- **Dense vrstvy:** Využívají několik vrstev plně propojených neuronů, což umožňuje modelu zpracovat a rozpozнат vzory ve vstupních datech.
- **Aktivační funkce:** Používá ReLU (Rectified Linear Unit) pro zavedení nelinearity do modelu, což umožňuje zachycení složitějších vzorů.
- **Výstupní vrstva:** Redukuje dimenzi výstupů na počet tříd, které mají být klasifikovány, a provádí finální rozhodování.

### RMLP pro BB příznaky

- **Vstupní vrstva:** Zpracovává vektor příznaků získaných z BB, který zahrnuje normalizované souřadnice a konfidenční skóre.
- **Dense vrstvy s normalizací a regulací:**
  - **Lineární transformace:** Zajišťuje základní transformaci vstupních dat.

- **BatchNorm**: Normalizační vrstva, která stabilizuje učení tím, že normalizuje vstupy do další vrstvy.
- **LeakyReLU**: Variantu aktivační funkce ReLU, která umožňuje malý průtok gradientu i pro záporné vstupy a pomáhá předcházet problému mizejících gradientů.
- **Dropout**: Regularizační technika, která „vypíná“ náhodný výběr neuronů během trénování, čímž pomáhá zabránit přetrénování modelu.
- **Výstupní vrstva**: Stejně jako u OH modelu, vrstva převede zpracované informace na finální klasifikační výsledek.

Rozšířená MLP architektura je vybavena pro hluboké porozumění a interpretaci dat. Vrstvy BatchNorm a Dropout hrají klíčovou roli ve zvýšení generalizační schopnosti modelu a jeho robustnosti proti přetrénování.

### 5.3.3 Evaluace modelů

Evaluace modelů na testovací sadě poskytuje přímý důkaz o výkonnosti různých přístupů (viz Tabulka 5.8). Volba nejlepšího modelu musí vyvažovat přesnost a efektivitu provozu.

*RMLP - BB* vyniká v přesnosti a má konkurenceschopné F1 skóre ve srovnání s ostatními modely, přičemž je výrazně přední v čase inference. Když k tomuto času přidáme dobu inferenčního zpracování modelu YOLOv8 (viz Tabulka 5.4), zůstává celková doba inferenčního procesu stále nižší než u *EfficientNetu B0*, což z něj dělá preferovaný model pro real-time aplikace. Navíc přístup k inferenci, který kombinuje YOLOv8 a RMLP, nabízí flexibilitu a možnost vylepšování modelu bez nutnosti rozšiřování datasetu a tréninku pouze zapomocí definovaných pravidel na základě detekovaných objektů.

*EfficientNet B0* sice dosahuje nejlepšího F1 Samples skóre, vykazuje ale nižší přesnost a výrazně delší časy inference, což je méně vhodné pro aplikace vyžadující rychlé zpracování.

Na základě těchto úvah a s ohledem na potřeby specifické pro klasifikaci scenerie je možné konstatovat, že *Rozšířené MLP - BB* představuje optimální rovnováhu mezi výkonem a časovou efektivitou.

Model	Přesnost	F1 Macro	F1 Samples	Inference*
EfficientNet B0	0.186	0.613	0.674	16
RMLP - BB	0.203	0.557	0.665	2
RMLP - OH	0.117	0.375	0.594	2
SVM - BB	0.082	0.423	0.591	3
SVM - OH	0.048	0.272	0.508	3
XGBoost - BB	0.136	0.606	0.508	15
XGBoost - OH	0.102	0.535	0.437	15

**Tabulka 5.8** Výsledky modelů víceštítkové klasifikace scenérie. [\*] čas inference v milisekundách byl měřen pro EfficientNet B0 na grafické kartě Nvidia Tesla T4 a pro ostatní modely na CPU

## 5.4 Experimenty na klasifikaci počasí

Poslední sadou experimentů se věnujeme klasifikaci meteorologických podmínek pomocí metod hlubokého učení. Při přístupu ke klasifikaci počasí byly využity podobné metodiky a metriky jako u klasifikace scenerie, tedy zohlednění přesnosti, F1 skóre (makro a vážené), a čas inference, které jsou stěžejní pro identifikaci nejvhodnějšího modelu.

### 5.4.1 Vývoj a přístupy

Experimentujeme s několika modely hlubokého učení, kdy každý z nich představuje přístup k řešení problému odvozený z jeho architektonických vlastností, trénovacích strategií a adaptace na konkrétní úkoly klasifikace. Níže je podrobný popis těchto modelů, včetně motivace pro jejich výběr, specifických nastavení a účelu jejich použití.

- **EfficientNet modely:**
  - **EfficientNet B0:** Tento model slouží jako základní verze rodiny EfficientNet, s nižším počtem parametrů a rychlejší inferencí. Byl zvolen pro jeho efektivitu a schopnost dosáhnout solidní přesnosti i na méně komplexních datasetech.
  - **EfficientNet B5:** tento model je podobný modelu výše, obsahuje ale vyšší počet parametrů a větší hloubku, a proto byl zvolen za účelem porovnání, jak škálování modelu ovlivní výkon na stejném datasetu.

Trénink obou modelů na klasifikačním datasetu stavíme na předtrénovaných instancích dostupných na PyTorch Hubu. Náš trénink probíhá ve dvou fázích: nejdříve probíhá se zamraženým předtrénovaným backbonem, aby se využilo transfer learningu, následně celý model dotrénováváme s nižším learning rate pro jemné doladění.

- **YOLOv8-CLS:** YOLOv8-CLS představuje adaptaci detekčního modelu YOLOv8 pro úkoly klasifikace. Využívá se přitom předností architektury YOLO známé svou rychlostí a efektivitou v detekci objektů. K inicializaci používáme předtrénované váhy na datasetu ImageNet Krizhevsky et al. [21], které poskytují pevný základ pro další finetuning specificky orientovaný na klasifikaci meteorologických jevů.

V první fázi model YOLOv8-CLS finetunujeme klasickým stylem s možností aktualizace vah po celé své struktuře. V následující fázi provádíme experiment s přenesením vah z backbone detekčního modelu YOLOv8s, přičemž tento backbone je následně zamražen a dále je trénována výhradně poslední klasifikační vrstva. Tuto strategii motivuje možnost využití detekčních vah pro efektivnější trénink finální klasifikační vrstvy a její potenciální aplikaci v multihead modelu s více detekčními hlavami (viz Kapitola 5.2.5).

- **YOLO-World:** Model YOLO-World postavený na fundamentech zero-shot detekce objektů zůstává ve své základní struktuře nezměněn. Jeho aplikace však spočívá ve specifikaci detekovaných objektů jakožto deskriptorů

meteorologických podmínek, což představuje odchylku od tradičních aplikací modelů objektové detekce, které se soustředí na identifikaci konkrétních fyzických objektů.

Využití YOLO-World v této roli je založeno na schopnosti modelu provádět klasifikaci a lokalizaci objektů v rámci určitých ohraničených rámečků. Kritickým rozšířením této funkcionality je definice objektů jakožto meteorologických stavů - například *slunečné počasí* či *deštivé počasí*. Tato metodika umožňuje modelu přejít od tradiční detekce objektů ke klasifikaci založené na výběru detekovaného objektu značící kategorii počasí s nejvyšším skórem konfidence.

#### 5.4.2 Evaluace přístupů

Na základě srovnání různých modelů pro klasifikaci (viz Tabulka 5.9) trénovaných na stejném jednoštítkovém datasetu (viz Kapitola 4.1.2), YOLOv8-CLS model ukazuje nejlepší celkové výsledky, což zahrnuje nejvyšší přesnost a F1 skóre ve srovnání s ostatními testovanými modely, včetně EfficientNet B0 a B5. Přestože YOLOv8-CLS má mírně delší dobu inference, jeho převaha ve výkonnosti a flexibilita přístupu jej činí preferovanou volbou pro aplikace klasifikace počasí. Experimenty se zero-shot modelem YOLO-World ukázaly že navrhovaný směr může být použitelný, avšak pro naše účely se jeví jako nedostatečný.

Model	Přesnost	F1 Macro	F1 Weighted	Inference*
EfficientNet B0	0.5156	0.4216	0.5886	16
EfficientNet B5	0.6797	0.2229	0.5978	28
YOLOv8-CLS	0.7266	0.6083	0.7193	12
YOLOv8-CLS**	0.6953	0.4471	0.6527	12
YOLO-World	0.5234	0.1934	0.4759	-

**Tabulka 5.9** Výsledky modelů pro klasifikaci počasí. [\*] čas inference v milisekundách měřen na grafické kartě Nvidia Tesla T4. [\*\*] YOLOv8 CLS se zamrazeným backbonem a váhami z YOLOv8 na detekci objektů.

### 5.5 Celkové shrnutí experimentů

V této kapitole jsme prezentovali experimenty zaměřené na zvýšení efektivity metod zpracování obrazu z webových kamer. Výsledky experimentů umožnily identifikaci a výběr nejúčinnějších metod pro detekci a klasifikaci obrazových dat.

Experimenty s algoritmy VoidSeeker a FragGuard pro kalibraci prahových hodnot ukázaly vysokou přesnost v detekci irrelevantních a částečně stažených obrazů (94.5% a 96% úspěšnost). Další testování ukázalo, že algoritmus VertiScan používající SVM s lineárním jádrem dosahuje 83% úspěšnosti, zatímco použití XG-Boost dosahuje mírně vyšší úspěšnosti. Pro rychlejší zpracování je však preferováno SVM.

Dále jsme testovali YOLOv8s modely pro detekci objektů, kde finetuning na specifických datasetech zlepšil přesnost a úplnost detekce. Multihead model

umožnil významné zrychlení inference. Modely se zamraženým backbonem nebo modely trénované na syntetických datasetech vykazovaly nižší výkonnost.

Pro klasifikaci scenerie dosáhl RMLP model přesnost 65.7%, zatímco model YOLOv8-CLS pro klasifikaci počasí dosáhl nejvyšší přesnosti 72.7% a F1 skóre 60.8%. Experimenty s různými modely pro klasifikaci počasí ukázaly, že YOLOv8-CLS představuje optimální volbu s dobou inference 12 ms, což je přijatelné pro náš systém.

Experimenty poskytly důkazy o efektivitě vybraných přístupů umožňující stanovení nejlepších praktik pro další vývoj a nasazení do systémů. Tato pozorování podporují naše rozhodnutí pro finální konfigurace a nastavení systému zaručující vysokou účinnost a spolehlivost v praxi.

# 6 Systém, implementace a využití

V této kapitole popisujeme architekturu, implementaci a možná využití prototypu systému pro analýzu a validaci obrazu v reálném čase. Systém využívá kombinaci moderních technologií a pokročilých metod strojového učení k zajištění efektivního a přesného zpracování obrazových dat z webových kamer. Hlavní komponenty systému zahrnují webové rozhraní založené na FastAPI<sup>1</sup>, NVIDIA Triton Inference Server<sup>2</sup> pro deployment modelů, soubor specializovaných neuronových sítí pro detekci a klasifikaci obrazu a Nginx pro hostování klientské části aplikace. Toto rozhraní poskytuje přístup k nahrávání a zobrazení výsledků analýzy a validace, přičemž Nginx<sup>3</sup> zajišťuje efektivní distribuci statických souborů a optimalizaci výkonu klienta. Systém je strukturován do několika fází, které na sebe logicky navazují a společně tvoří koherentní proces zpracování:

1. **Controller:** Přijímání obrazových dat a koordinace zpracování.
2. **Předzpracování a validátory obrazu:** Ověření kvality a relevance obrazu.
3. **Detekce objektů spolu s klasifikací počasí:** Identifikace objektů a jejich analýza, která končí klasifikací počasí.
4. **Validace na základě objektů:** Kontrola obsahu obrazu na základě detekovaných objektů.
5. **Klasifikace scenérie (kategorizace):** Rozpoznání a kategorizace scény.
6. **Agregace výsledků:** Agregace výsledků předchozích kroků workflow systému a vracení celkové výsledné analýzy.

## 6.1 Architektura

Systém je strukturován do několika hlavních modulů, jejichž koordinace zajišťuje komplexní zpracování a analýzu obrazu (viz Obrázek 6.1). V této části je každá komponenta a její role v systému popsána s důrazem na technické detaily.

### 6.1.1 Controller

Controller představuje vstupní bod systému. Skrze webové rozhraní *FastAPI* poskytující *RESTful API*<sup>4</sup> je zodpovědný za přijímání obrazových dat od uživatelů

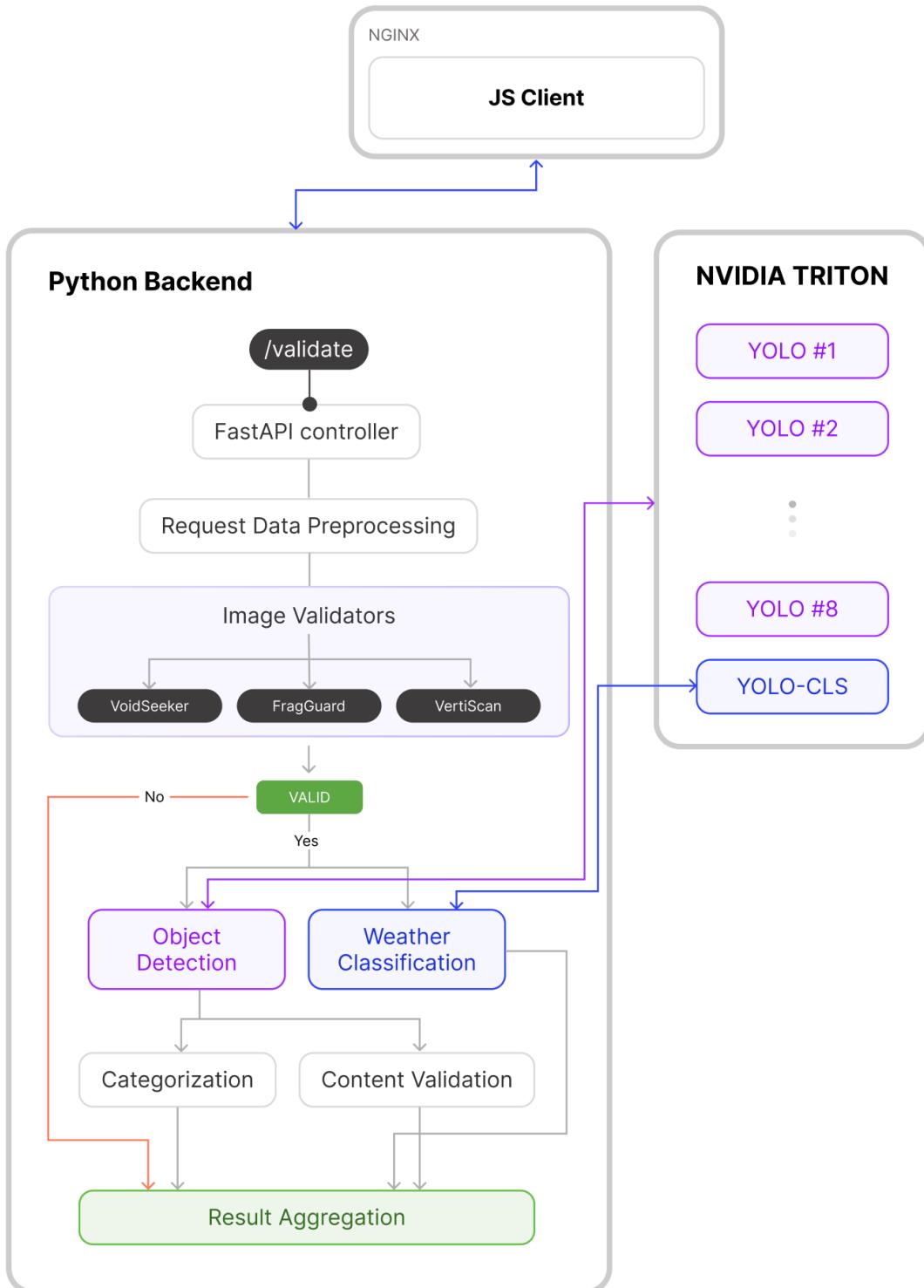
---

<sup>1</sup>FastAPI oficiální stránky: <https://fastapi.tiangolo.com/>

<sup>2</sup>NVIDIA Triton Inference Server oficiální stránky: <https://developer.nvidia.com/triton-inference-server>

<sup>3</sup>Nginx oficiální stránky: <https://nginx.org/>

<sup>4</sup>RESTful API je architektura webových služeb založená na principu Representational State Transfer, která využívá standardní metody protokolu HTTP (GET, POST, PUT, DELETE) pro komunikaci. Tato architektura je bezstavová, podporuje systémové vrstvení a umožňuje ukládání dat do dočasné paměti pro zvýšení efektivity, rozšířitelnosti a spolehlivosti v distribuovaných systémech.



**Obrázek 6.1** Architektura navrhovaného prototypu systému

pomocí koncových bodů rozhraní. FastAPI je zvoleno pro jeho asynchronní zpracování požadavků, což zlepšuje schopnost systému paralelně zpracovávat vysoký objem požadavků. V rámci API je definován koncový bod pro analýzu a validaci obrazu a koncový bod pro statické HTML. Analyzační bod rozhraní umožňuje uživateli nahrát obraz, který je poté systémem analyzován a validován. Druhý

zmíněný bod poskytuje uživatelské rozhraní pro interakci s API, včetně nahrávání obrazů a zobrazení výsledků.

### 6.1.2 Předzpracování CV validátory

Po přijetí obrazu dochází k jeho validaci prostřednictvím sady paralelně spuštěných validátorů, které zahrnují následující kroky:

- Detekce obrazů bez relevantního obsahu
- Detekce částečně stažených obrázků
- Detekce vertikálně poškozených obrázků

Tento krok je důležitý pro zajištění, že data postupující do dalších fází analýzy jsou validní a vhodná pro zpracování.

### 6.1.3 Detekce objektů a klasifikace

Pro detekci objektů a klasifikaci meteorologických podmínek jsou využívány specializované YOLO modely spolu s YOLO-World modelem hostované na NVIDIA Triton Inference Serveru. Využití Tritonu umožňuje flexibilní scaling a efektivní využití hardwarových zdrojů pro paralelní inferenci.

### 6.1.4 Validace na základě objektů

V této fázi systém ověřuje, zda obraz neobsahuje nevhodný nebo nežádoucí obsah, na základě předem definovaných kritérií a detekovaných objektů. Tento krok je klíčový pro eliminaci nevhodného materiálu a zajistí, že výstupní data jsou vhodná pro další použití.

### 6.1.5 Klasifikace scenérie

Proces průchodu modulem klasifikace scenérie probíhá paralelně s předchozím zmíněným modulem „validace na základě objektů“. Zde dochází k aplikaci strojových učících modelů pro klasifikaci scény. Modely jsou navrženy tak, aby rozpoznaly a kategorizovaly scénu na základě objektů, jejich rozložení a dalších vizuálních charakteristik.

### 6.1.6 Agregace výsledků

V poslední fázi jsou všechny informace a výsledky analýzy agregovány do jednoho strukturovaného JSON objektu. Tento objekt je pak odeslán zpět uživateli jako odpověď na původní požadavek a poskytuje ucelený přehled o výsledcích analýzy a validace obrazu.

## 6.2 Použité technologie

Pro realizaci našeho prototypu byla vybrána řada technologií a knihoven, které dosahují efektivního a přesného zpracování obrazových dat v reálném čase. Celý systém je psán v *Pythonu*, což je moderní a flexibilní programovací jazyk. Python byl zvolen nejen pro svou flexibilitu a širokou podporu knihoven, ale také pro jeho schopnost snadno se integrovat s jinými systémy a technologiemi, což je důležité pro vývoj rozsáhlých a modulárních aplikací.

Níže uvádíme výčet hlavních použitých knihoven a technologií.

- **OpenCV (Open Source Computer Vision Library)**<sup>5</sup> je knihovna s otevřeným zdrojovým kódem určená pro počítačové vidění a strojové učení. OpenCV je zásadní pro zpracování a analýzu obrazových dat v našem systému poskytující bohatý výběr funkcí pro různorodé úlohy zpracování obrazu. Díky její výkonnosti a efektivitě v reálném čase umožňuje OpenCV našemu systému rychle a efektivně zpracovávat obrazová data.
- **PyTorch**<sup>6</sup> je hlavní framework pro hluboké neuronové sítě použitý v tomto systému, který umožňuje vývoj a trénování komplexních modelů strojového učení s vysokou úrovní efektivity a flexibility. Díky jeho dynamickému grafu a podpoře automatické diferenciace poskytuje ideální prostředí pro rychlý vývoj a experimentování s modely.
- **Numpy**<sup>7</sup> a **Pillow**<sup>8</sup> jsou knihovny, které slouží pro zpracování obrazových dat. Numpy je knihovna pro numerické výpočty, která je ideální svou efektivitou na manipulaci s multidimenzionálními poli, což je základ pro práci s obrazovými daty. Pillow knihovna pro zpracování obrazů pak poskytuje nástroje pro čtení, zápis a manipulaci s obrazy.
- **FastAPI**<sup>9</sup> je moderní, vysoce výkonný webový framework pro stavbu API s Pythonem 3.7+, který je základem webového rozhraní systému.
- **Vanilla JS klient** je jednoduchý klient použitý jako demo aplikace, který umožňuje nahrávání obrazů a zobrazování výsledků analýzy a validace. Tento minimalistický přístup zajišťuje, že aplikace je přístupná a snadno použitelná i bez pokročilých znalostí webových technologií.
- **Docker kontejner**<sup>10</sup> je lehký, přenosný balíček s kódem a vším potřebným pro jeho spuštění. Kontejnery jsou izolované a běží samostatně, čímž zajišťují konzistenci a efektivitu aplikací bez ohledu na prostředí.
- **Dockerizace** je proces balení softwaru do kontejnerů, což umožňuje aplikacím běžet izolovaně od ostatních procesů a zjednoduší jejich nasazení a škálování na jakémkoliv hardwaru, zatímco zachovává konzistenci prostředí.

<sup>5</sup>OpenCV oficiální stránky: <https://opencv.org/>

<sup>6</sup>Pytorch oficiální stránky: <https://pytorch.org/>

<sup>7</sup>Numpy oficiální stránky: <https://numpy.org/>

<sup>8</sup>Pillow oficiální stránky: <https://python-pillow.org/>

<sup>9</sup>FastAPI oficiální stránky: <https://fastapi.tiangolo.com/>

<sup>10</sup>Docker oficiální stránky: <https://www.docker.com/>

- **Docker Compose** je nástroj, který umožňuje definovat a spustit více Docker kontejnerů jako jeden nebo více služeb pomocí jednoduchého YAML souboru. Tento přístup zjednodušuje konfiguraci a správu aplikací složených z více kontejnerů, což usnadňuje jejich spouštění na lokálním vývojovém stroji nebo v produkčním prostředí.
- **NVIDIA Triton Inference Server**<sup>11</sup> poskytuje vysokou výkonnou a flexibilní infrastrukturu pro nasazování a škálování modelů umělé inteligence a strojového učení s podporou pro grafické karty i procesory. Tato platforma umožňuje integraci modelů z nejpopulárnějších vývojových prostředí, jako jsou TensorFlow<sup>12</sup>, PyTorch a ONNX<sup>13</sup>, přičemž zajišťuje efektivní správu zdrojů a distribuci inferenčních úloh.
- **Nginx**<sup>14</sup> je vysoce výkonný HTTP a reverzní proxy server, stejně jako mail proxy server. V kontextu našeho systému je používán pro hostování klientské části aplikace zajišťující rychlý přístup a distribuci statických souborů. Nginx se vyznačuje svou vysokou efektivitou a schopností zvládat tisíce současných spojení, což je zásadní pro udržení vysoké dostupnosti a rychlé odezvy aplikace. Jeho konfigurovatelnost a modulárnost umožňují snadné nasazení a optimalizaci podle specifických potřeb našeho projektu.

V našem systému jsou hlavní komponenty, *Python backend* a *NVIDIA Triton Inference Server* zabaleny v *Docker kontejnerech* a propojeny pomocí *Docker Compose*. Tato architektura značně usnadňuje nasazování do infrastruktury a zajišťuje konzistenci prostředí napříč různými platformami. Dockerizace tak umožňuje našemu systému běžet izolovaně od ostatních aplikací a Docker Compose zjednoduší konfiguraci a správu více kontejnerů, což tvoří společně fungující celistvý systém.

Tato kombinace technologií a knihoven poskytuje robustní a flexibilní základ pro vývoj a nasazení systému pro analýzu a validaci obrazu z webových kamer.

## 6.3 Analýza časů zpracování

V této části poskytujeme podrobnou analýzu časů zpracování pro různé kombinace modelů a hardwaru, což nám umožňuje lépe porozumět omezením a výkonnosti našeho systému. Tato analýza nám také poskytuje cenné informace pro další optimalizaci a výběr vhodné hardwarové infrastruktury pro specifické aplikace a scénáře použití.

### 6.3.1 Použité hardwarové konfigurace

Zkoumali jsme výkonnost našeho systému na třech různých hardwarových konfiguracích, které zahrnují různé typy CPU a GPU. Tyto konfigurace jsou:

---

<sup>11</sup>NVIDIA Triton Inference Server oficiální stránky: <https://developer.nvidia.com/triton-inference-server>

<sup>12</sup>TensorFlow oficiální stránky: <https://www.tensorflow.org/>

<sup>13</sup>ONNX oficiální stránky: <https://onnx.ai/>

<sup>14</sup>Nginx oficiální stránky: <https://nginx.org/>

- **ONNX modely na Apple Silicon M1 (8 core CPU)**: Tato konfigurace využívá 8jádrový CPU Apple M1 spolu s 16GB unifikované paměti
- **TensorRT modely na NVIDIA Tesla T4 (16GB GPU) + AMD EPYC 7V12 (4 core CPU)**: Tato konfigurace kombinuje výkonnou GPU NVIDIA Tesla T4 s 4jádrovým CPU AMD EPYC ve virtuálním stroji, což umožňuje extrémně rychlé zpracování při použití TensorRT optimalizovaných modelů.
- **ONNX modely na AMD Ryzen Threadripper 2970WX (24 core CPU)**: S 24 jádry CPU poskytuje AMD Ryzen Threadripper vynikající výpočetní výkon pro zpracování ONNX modelů.

### 6.3.2 Výsledky měření

Analýza časů zpracování pro různé kombinace modelů a hardwaru odhaluje významné rozdíly ve výkonnosti (viz Tabulka 6.1).

Modely	HW	Čas zpracování
ONNX	M1 (8 core CPU)	4.759s
ONNX	AMD (24 core CPU)	1.523s
TensorRT	T4 (16GB GPU) + AMD (4 core CPU)	0.148s

**Tabulka 6.1** Průměrné časy zpracování pro různé konfigurace

Z těchto výsledků je zřejmé, že použití specializovaného hardwaru, jako je NVIDIA Tesla T4 GPU ve spojení s TensorRT modely, výrazně snižuje časy zpracování, což umožňuje realtime odezvu systému. Naopak, i přes vysokou efektivitu a optimalizaci ONNX modelů pro Apple Silicon M1, časy zpracování na této platformě jsou signifikantně delší. AMD Ryzen Threadripper konfigurace s ONNX modely představuje díky vysokému počtu jader procesoru vyváženou volbu, která nabízí dobrý kompromis mezi výkonností a dostupností hardwaru.

## 6.4 Modely v prototypu systému

V rámci našeho prototypu systému je implementována podpora pouze pro ONNX modely na CPU. Toto rozhodnutí bylo učiněno s ohledem na zajištění multiplatformní kompatibility a snadného nasazení systému na různých hardwarových konfiguracích bez potřeby specifického grafického hardwaru.

Použití TensorRT modelů, i když nabízí výrazně lepší výkonnost na podporovaných NVIDIA grafických kartách, přináší dodatečné komplexity a omezení. Přechod od ONNX modelů k TensorRT vyžaduje kompliaci modelů specificky pro cílové grafické karty, což je proces, který může trvat netriviální množství času a vyžaduje přístup k příslušnému hardwaru.

V kontextu našeho prototypu systému bylo proto rozhodnuto prioritizovat univerzálnost a snadné nasazení na úkor maximální možné výkonnosti. Tato volba reflektuje záměr poskytnout robustní a přístupné řešení pro analýzu a validaci obrazu, které je dostupné pro co nejširší skupinu uživatelů a aplikací.

## 6.5 Diskuse o implementaci

Architektura systému je navržena s ohledem na vysokou propustnost, efektivitu a přesnost analýzy obrazových dat. Integrace s NVIDIA Triton Inference Serverem a použití pokročilých modelů strojového učení umožňuje systému dosahovat vynikajících výsledků v realtime aplikacích. Systém je navržen tak, aby umožňoval snadnou integraci nových modelů a rozšiřování funkcionalit podle potřeb uživatelů.

Výsledky měření rychlosti systému ukazují, že pro dosažení nejvyšší možné efektivity a minimálního času zpracování je důležité vhodně kombinovat modely strojového učení s optimalizovaným hardwarem. Využití dedikovaných GPU s TensorRT modely může nabídnout významné zlepšení výkonu, zatímco ONNX modely na výkonných CPU mohou představovat vhodnější řešení pro scénáře s omezenými zdroji.

## 6.6 Využití a adaptabilita systému

Tato část se zaměřuje na možnosti uplatnění systému vyvinutého v rámci bakalářské práce. Systém je navržen tak, aby mohl být snadno modifikován a adaptován na široký rozsah aplikací od monitoringu zemědělských oblastí až po pomoc v meteorologii.

### 6.6.1 Zemědělství a monitorování vegetace

Systém lze modifikovat za účelem sledování vegetace v zemědělském sektoru, což by umožnilo detekci změn ve vývoji a zdraví plodin. Jeho funkčnost by se opírala o nynější schopnost rozpoznávat různé objekty na snímcích od trávy po stromy, které jsou potřeba k porozumění scény a možnosti její klasifikace. Tato funkcionalita by byla určena pro zemědělce, kterým systém může umožnit efektivněji spravovat obdělávané plochy a aktivně řešit vzniklé problémy. Systém lze přizpůsobit pro detekci konkrétních problémů zemědělství, jako jsou škůdci nebo sucho, což přispívá k lepšímu využívání zdrojů a podpoře udržitelného hospodaření.

### 6.6.2 Real-time monitorování meteorologických situací

Systém může přispět k poskytování okamžitých náhledů na aktuální meteorologické podmínky prostřednictvím analýzy snímků z webových kamer. Tato schopnost je cenná především pro meteorologické služby a aplikace pro předpověď počasí, které mohou integrovat živé záběry do svých platform. Služby a aplikace mohou využívat systém pro korekci a úpravu předpovědí počasí a taktéž ho lze rozšířit o funkce pro automatickou detekci specifických meteorologických jevů a uživatelům tím poskytnout přístup k aktuálním informacím o počasí.

### 6.6.3 Monitorování dopravní situace

Systém lze upravit pro analýzu snímků z dopravních kamer, a tak poskytovat real-time informace o stavu dopravy díky automatickým detekcím objektů, jako jsou silnice, lidé nebo auta. Systém lze snadno rozšířit o specializované detekce zácp,

nehod a změn v provozu, které tvoří zásadní informace pro optimalizaci dopravy, zvyšování bezpečnosti a efektivní plánování tras jak pro správce infrastruktury, tak pro běžné uživatele.

#### 6.6.4 Shrnutí využití

Každá z uvedených aplikací demonstруje flexibilitu a široké možnosti využití systému, který může být adaptován a rozšířen o další funkce pro specifické účely. Díky své schopnosti rychlé a přesné analýzy obrazových dat může systém významně přispět k lepšímu rozhodování a efektivnější reakci na různé dynamické podmínky v reálném světě.

### 6.7 Shrnutí

V této kapitole jsme popsali architekturu, implementaci a možná využití prototypu systému pro analýzu a validaci obrazu v reálném čase. Vyvinutý systém kombinuje moderní technologie a pokročilé metody strojového učení, což zajišťuje efektivní a přesné zpracování obrazových dat z webových kamer. Mezi klíčové komponenty patří webové rozhraní založené na FastAPI, NVIDIA Triton Inference Server pro hostování modelů strojového učení, soubor specializovaných neuronových sítí pro detekci a klasifikaci obrazu a Nginx pro hostování klientské části aplikace. Systém je strukturován do několika fází, které na sebe logicky navazují a společně tvoří koherentní proces zpracování zahrnující přijímání dat, předzpracování, detekci objektů, validaci a agregaci výsledků. Tento systém lze využít v různých aplikacích od zemědělství po monitorování dopravních situací.

# Závěr

V této práci jsme vyvinuli a implementovali prototyp systému pro analýzu a validaci obrazu z webových kamer, který poskytuje názornou demonstraci toho, jak moderní modely a algoritmy strojového učení fungují na uživatelských datech z reálného prostředí. Naše práce spojuje teoretické základy s praktickými implementacemi, čímž umožňuje identifikaci a řešení klíčových problémů v oblasti detekce poškozených obrázků, detekce objektů, klasifikace scén a meteorologických podmínek.

V rámci analýzy a experimentů jsme se nejdříve zaměřili na detekci nežádoucích objektů a poškozených obrázků. Námi vyvinuté algoritmy VoidSeeker a FragGuard dosáhly přesnosti detekce irrelevantních a částečně stažených obrázků 94.5% a 96%. VertiScan implementovaný s SVM s lineárním jádrem dosáhl úspěšnosti 83%.

Experimentální fáze dále ukázala, že aplikace osmi specializovaných modelů YOLOv8 přináší značné zlepšení v detekci objektů s optimální výkonností a úspěšností. Námi vyvinutý multihead model YOLOv8s, i přes nižší přesnost, předvedl vyšší inferenční rychlosť. Proti tomu stojí samostatný model YOLOv8s trénovaný na agregovaném datasetu, který se ukázal méně efektivní, stejně jako zero-shot model YOLO-World, který nesplnil očekávání ve srovnání se specializovanými modely, zejména v kontextu specifických detekcí jako texty či nahota.

V oblasti klasifikace scén se ukázalo, že využití rozšířeného modelu MLP - RMPL, založeného na extrahovaných příznacích z obrazu pomocí modelů YOLOv8s, dosahuje nejvyšší úrovně přesnosti a nabízí krátkou dobu inference. V klasifikaci počasí exceloval model YOLOv8-CLS, který překonal modely z rodiny EfficientNet a dosáhl přesnosti 72.7% a F1 skóre 60.8%.

## Závěrečné hodnocení

Úspěšně realizovaná integrace metod strojového učení a počítačového vidění v této práci dokládá, že je možné vytvořit nástroj pro automatickou analýzu a validaci obrazových dat v reálném čase, čímž byl splněn hlavní cíl práce. Experimentální fáze potvrdila efektivitu vybraných metod, přičemž aplikace osmi specializovaných modelů YOLOv8 a vyvinutý multihead model YOLOv8 vynikly svou výkonností. To demonstruje, že pečlivý výběr a optimalizace modelů strojového učení mají klíčový význam pro úspěšnost detekce a klasifikace v real-time aplikacích.

Přestože byly dosaženy významné úspěchy v oblasti detekce a klasifikace, z důvodu komplexnosti celé problematiky a nutnosti soustředit se na ostatní aspekty systému nebylo možné v rámci této práce implementovat detekci směru kamery podle polohy slunce. Ačkoliv byla vyvinuta funkční a efektivní detekce slunce, detekce směru kamery zůstává otevřeným polem pro budoucí výzkum. Tento fakt poukazuje na výzvy, které přináší integrace a optimalizace různorodých funkcí v komplexních systémech real-time analýzy.

Vývoj navrženého demonstračního prototypu a jeho následné hodnocení a validace na rozmanitých datasetech ukazuje robustnost a spolehlivost. Tyto výsledky poskytují pevný základ pro další rozvoj a optimalizaci systému, a zároveň otevírají možnosti pro jeho praktické využití v širších aplikačních doménách.

## Budoucí práce

V rámci budoucích směrů výzkumu a vývoje představovaného systému pro analýzu a validaci obrazových dat se nabízí několik klíčových oblastí, na které se lze zaměřit. Tyto oblasti reflektují potenciál pro rozšíření a zlepšení stávajícího systému.

Jedním z prvních kroků v rámci budoucích prací by mělo být přetrenování YOLO modelu na větší počet epoch a s větší velikostí batche. Z časových omezení během vývoje systému nebylo možné plně využít potenciál modelu YOLO pro detekci objektů. Dlouhodobější trénink by mohl modelu umožnit lepší generalizaci a dosažení vyšší přesnosti v detekci objektů, což by vedlo ke zlepšení celkové účinnosti systému.

Dalším významným krokem pro rozvoj systému je implementace a integrace YOLO multihead modelu, který by zahrnoval nejen detekční hlavy pro rozpoznávání objektů, ale také klasifikační hlavu. Tento přístup by umožnil modelu zpracovávat a analyzovat obrazová data v komplexnější úrovni, zahrnující jak detekci objektů, tak klasifikaci scén nebo meteorologických podmínek. Klíčovým aspektem by zde byla možnost celkového tréninku modelu, což by umožnilo lepší adaptaci na specifické charakteristiky zpracovávaných dat.

V kontextu rozšířování funkcionalit systému představuje významnou oblast výzkumu detekce směru kamery na základě pozice slunce na fotografii. Tato funkcionality vyžaduje pokročilé algoritmy pro analýzu obrazu a výpočet geometrických vztahů, což přináší nové možnosti pro interpretaci a validaci obrazových dat. Přestože pozice slunce lze na fotografii určit, komplexnost a rozsah původní práce si vyžadovaly vynechání této části.

Tato práce představuje přínos v oblasti počítačového vidění a strojového učení, zdůrazňující potenciál těchto technologií pro reálné aplikace. Předložené výsledky a zjištění otevírají cestu pro další výzkum, zejména v kontextu rozšíření funkčnosti systému a jeho aplikace v nových oblastech. Tímto se potvrzuje, že kombinace teoretických poznatků a praktických aplikací může vést k inovacím, které mají reálný dopad na společnost a přispívají k lepšímu porozumění a interakci s naším okolím.

# Literatura

1. CANNY, John. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*. 1986, č. 6, s. 679–698.
2. JOCHER, Glenn; CHAURASIA, Ayush; QIU, Jing. *Ultralytics YOLO* [<https://github.com/ultralytics/ultralytics>]. 2023. Ver. 8.0.0.
3. VIOLA, Paul; JONES, Michael. Robust Real-time Object Detection. *International Journal of Computer Vision*. 2001.
4. LOWE, David G. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*. 2004, roč. 60, č. 2, s. 91–110.
5. DALAL, Navneet; TRIGGS, Bill. Histograms of Oriented Gradients for Human Detection. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2005, s. 886–893.
6. GIRSHICK, Ross; DONAHUE, Jeff; DARRELL, Trevor; MALIK, Jitendra. *Rich feature hierarchies for accurate object detection and semantic segmentation*. 2013.
7. REDMON, Joseph; DIVVALA, Santosh; GIRSHICK, Ross; FARHADI, Ali. You Only Look Once: Unified, Real-Time Object Detection. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, s. 779–788. Dostupné z DOI: [10.1109/CVPR.2016.91](https://doi.org/10.1109/CVPR.2016.91).
8. LIU, Wei; ANGUELOV, Dragomir; ERHAN, Dumitru; SZEGEDY, Christian; REED, Scott; FU, Cheng-Yang; BERG, Alexander C. SSD: Single Shot Multi-Box Detector. In: LEIBE, Bastian; MATAS, Jiri; SEBE, Nicu; WELLING, Max (ed.). *Computer Vision – ECCV 2016*. Springer International Publishing, 2016, s. 21–37. ISBN 978-3-319-46448-0. Dostupné z DOI: [10.1007/978-3-319-46448-0\\_2](https://doi.org/10.1007/978-3-319-46448-0_2).
9. REDMON, Joseph; FARHADI, Ali. *YOLO9000: Better, Faster, Stronger*. 2016. Dostupné z eprint: [arXiv:1612.08242](https://arxiv.org/abs/1612.08242).
10. UIJLINGS, Jasper R. R.; SANDE, Koen E. A. van de; GEVERS, Theo; SMEULDERS, Arnold W. M. Selective Search for Object Recognition. *International Journal of Computer Vision*. 2013, roč. 104, s. 154–171. Dostupné z DOI: [10.1007/s11263-013-0620-5](https://doi.org/10.1007/s11263-013-0620-5).
11. CORTES, Corinna; VAPNIK, Vladimir. Support-vector networks. *Machine learning*. 1995, roč. 20, č. 3, s. 273–297.
12. GIRSHICK, Ross. *Fast R-CNN*. 2015. Dostupné z eprint: [arXiv:1504.08083](https://arxiv.org/abs/1504.08083).
13. REN, Shaoqing; HE, Kaiming; GIRSHICK, Ross; SUN, Jian. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*. 2015, roč. 28.
14. REDMON, Joseph; FARHADI, Ali. *YOLOv3: An Incremental Improvement*. 2018. Dostupné z eprint: [arXiv:1804.02767](https://arxiv.org/abs/1804.02767).
15. WANG, Chien-Yao; BOCHKOVSKIY, Alexey; LIAO, Hong-Yuan Mark. *YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*. 2022. Dostupné z eprint: [arXiv:2207.02696](https://arxiv.org/abs/2207.02696).

16. CHENG, Tianheng; SONG, Lin; GE, Yixiao; LIU, Wenyu; WANG, Xinggang; SHAN, Ying. YOLO-World: Real-Time Open-Vocabulary Object Detection. *arXiv preprint arXiv:2401.17270*. 2024.
17. RADFORD, Alec; KIM, Jong Wook; HALLACY, Chris; RAMESH, Aditya; GOH, Gabriel; AGARWAL, Sandhini; SASTRY, Girish; ASKELL, Amanda; MISHKIN, Pamela; CLARK, Jack; KRUEGER, Gretchen; SUTSKEVER, Ilya. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. Dostupné z eprint: [arXiv:2103.00020](https://arxiv.org/abs/2103.00020).
18. SANDLER, Mark; HOWARD, Andrew; ZHU, Menglong; ZHMOGINOV, Andrey; CHEN, Liang-Chieh. MobileNetV2: Inverted Residuals and Linear Bottlenecks [The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 4510-4520]. 2018. Dostupné z eprint: [arXiv:1801.04381](https://arxiv.org/abs/1801.04381).
19. TAN, Mingxing; PANG, Ruoming; LE, Quoc V. EfficientDet: Scalable and Efficient Object Detection [Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020)]. 2019. Dostupné z eprint: [arXiv:1911.09070](https://arxiv.org/abs/1911.09070).
20. LIN, Tsung-Yi; MAIRE, Michael; BELONGIE, Serge; BOURDEV, Lubomir; GIRSHICK, Ross; HAYS, James; PERONA, Pietro; RAMANAN, Deva; ZITNICK, C. Lawrence; DOLLÁR, Piotr. *Microsoft COCO: Common Objects in Context*. 2014. Dostupné z eprint: [arXiv:1405.0312](https://arxiv.org/abs/1405.0312).
21. KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*. 2017, roč. 60, č. 6, s. 84–90. ISSN 1557-7317. Dostupné z DOI: [10.1145/3065386](https://doi.org/10.1145/3065386).
22. SIMONYAN, Karen; ZISSERMAN, Andrew. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2014. Dostupné z eprint: [arXiv:1409.1556](https://arxiv.org/abs/1409.1556).
23. HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. *Deep Residual Learning for Image Recognition*. 2015. Dostupné z eprint: [arXiv:1512.03385](https://arxiv.org/abs/1512.03385).
24. TAN, Mingxing; LE, Quoc V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks [International Conference on Machine Learning, 2019]. 2019. Dostupné z eprint: [arXiv:1905.11946](https://arxiv.org/abs/1905.11946).
25. DOSOVITSKIY, Alexey; BEYER, Lucas; KOLESNIKOV, Alexander; WEISSEN-BORN, Dirk; ZHAI, Xiaohua; UNTERTHINER, Thomas; DEHGHANI, Mostafa; MINDERER, Matthias; HEIGOLD, Georg; GELLY, Sylvain; USZKOREIT, Jakob; HOULSBY, Neil. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2020. Dostupné z eprint: [arXiv:2010.11929](https://arxiv.org/abs/2010.11929).
26. CHEN, Tianqi; GUESTRIN, Carlos. XGBoost: A Scalable Tree Boosting System. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Francisco, California, USA: ACM, 2016, s. 785–794. KDD ’16. ISBN 978-1-4503-4232-2. Dostupné z DOI: [10.1145/2939672.2939785](https://doi.org/10.1145/2939672.2939785).
27. SOKOVNIN, Nikita; ČUHEL, Jan; KOZLOVSKÝ, Martin; TERŠEK, Matija. *datadreamer*. 2024. Ver. v0.1.3. Dostupné také z: <https://github.com/luxonis/datadreamer>.

28. MINDERER, Matthias; GRITSENKO, Alexey; HOULSBY, Neil. *Scaling Open-Vocabulary Object Detection*. 2023. Dostupné z eprint: [arXiv:2306.09683](https://arxiv.org/abs/2306.09683).
29. WANG, Jiayuan; Wu, QM; ZHANG, Ning. You Only Look at Once for Real-time and Generic Multi-Task. *arXiv preprint arXiv:2310.01641*. 2023.

# Seznam obrázků

3.1	Výstup detekce objektů modelu YOLOv8 od Ultralytics . . . . .	19
3.2	Architektura R-CNN . . . . .	21
3.3	Architektura Fast R-CNN . . . . .	22
3.4	Anchor boxy . . . . .	23
3.5	Architektura Faster R-CNN . . . . .	23
3.6	Architektura původní první verze modelu YOLO . . . . .	24
3.7	Architektura YOLOv8 . . . . .	27
3.8	Architektura SSD . . . . .	29
3.9	Základní architektura konvoluční neuronové sítě . . . . .	32
3.10	Architektura EfficientNet B0 . . . . .	34
4.1	Distribuce datových množin datasetu na klasifikaci scenerie . . . . .	39
4.2	Distribuce datových množin datasetu na klasifikaci počasí . . . . .	40
5.1	Grafy ROC pro detekci obrázků bez relevantního obsahu. . . . .	48
5.2	Graf ROC pro určení prahů z testovací množiny na detekci částečně stažených obrázků. . . . .	49
5.3	Yolo multihead architektura (připojení na feature pyramid z bacoboru). . . . .	54
6.1	Architektura navrhovaného prototypu systému . . . . .	65

# Seznam tabulek

2.1	Příklady snímků, které neprojdou úvodním krokem kontroly . . . . .	12
3.1	Porovnání výkonnosti modelů na detekci objektů . . . . .	30
4.1	Počty obrázků a anotací v MSCOCO 2017 datasetu . . . . .	41
5.1	Evaluace specializovaných finetunovaných modelů YOLOv8 . . . . .	53
5.2	Evaluace specializovaných finetunovaných modelů YOLOv8s se zamraženým CNN backbonem . . . . .	53
5.3	Parametry modelu YOLOv8 s 1-8 specializovanými detekčními hlavami . . . . .	54
5.4	Evaluace času predikce YOLOv8 modelů s 1-8 specializovanými detekčními hlavami . . . . .	55
5.5	Evaluace jediného finetunovaného modelu YOLOv8 na agregovaném datasetu . . . . .	55
5.6	Evaluace jediného finetunovaného modelu YOLOv8 trénovaného na syntetickém datasetu obsahující všechny detekované kategorie jako výše zmíněné specializované modely. . . . .	56
5.7	Evaluace YOLO-World modelu na datasetech . . . . .	56
5.8	Výsledky modelů vícestítkové klasifikace scenérie . . . . .	60
5.9	Výsledky modelů pro klasifikaci počasí . . . . .	62
6.1	Průměrné časy zpracování pro různé konfigurace . . . . .	69

# A Přílohy

## A.1 Přehled elektronických příloh

Složka **meteval** obsahuje zdrojové kódy demonstrativního prototypu systému vyvinutého v rámci této práce včetně natrénovaných modelů a dokumentací.

- **meteval/docs/cs/technical\_doc.pdf**: Dokumentace popisující technické detaily systému
- **meteval/docs/cs/user\_doc.pdf**: Uživatelská dokumentace sloužící jako manuál pro koncové uživatele systému. Popisuje, jak systém ovládat, jaké funkce nabízí a jaké jsou požadavky pro jeho běžné použití
- **meteval/README.md**: README soubor obsahuje základní informace o projektu a návod k jeho spuštění
- **meteval/assets**: Složka obsahující dva testovací obrázky