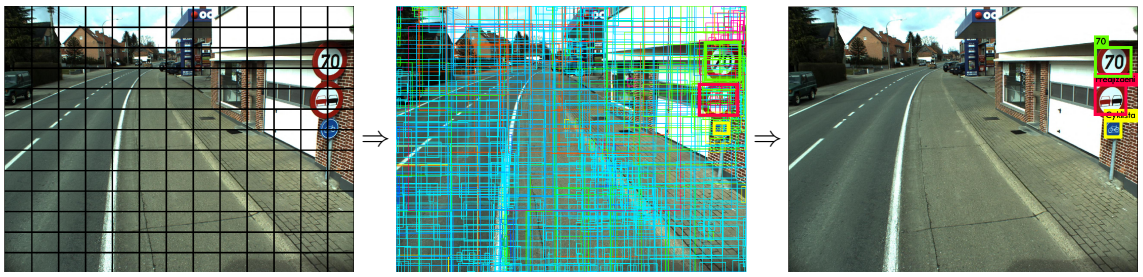
BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2019
http://excel.fit.vutbr.cz

# Detekce dopravních značek v obraze a videu pomocí konvolučních neuronových sítí

Filip Kočica*



**Abstrakt**

Tato práce řeší problematiku detekce dopravního značení za pomoci moderních technik zpracování obrazu. K řešení byla použita speciální architektura hluboké konvoluční neuronové sítě zvaná YOLO, tedy You Only Look Once, která řeší detekci i klasifikaci objektů v jednom kroce, což celý proces značně urychluje. Práce pojednává také o porovnání úspěšnosti modelů trénovaných na reálných a syntetických datových sadách. Podařilo se dosáhnout úspěšnosti 63.4% mAP při použití modelu trénovaného na reálných datech a úspěšnosti 82.3% mAP při použití modelu trénovaného na datech syntetických. Vyhodnocení jednoho snímku trvá na průměrně výkonném grafickém čipu $\sim$40.4*ms* a na nadprůměrně výkonném čipu $\sim$3.9*ms*. Přínosem této práce je skutečnost, že model neuronové sítě trénovaný na syntetických datech může za určitých podmínek dosahovat podobných či lepších výsledků, než model trénovaný na reálných datech. To může usnadnit proces tvorby detektoru o nutnost anotovat velké množství obrázků.

**Klíčová slova:** Konvoluční neuronová síť — YOLO — Detekce — Syntetická — Dopravní značka

**Přiložené materiály:** Demonstrační video — Generátor datových sad

*xkocic01@stud.fit.vutbr.cz, *Fakulta informačních technologií, Vysoké učení technické v Brně*

## 1. Úvod

Asistenční systémy pro řidiče a autonomní vozidla se v posledních letech dočkaly velké pozornosti. Stojí zejména na správné interpretaci okolního prostředí, kam mj. spadá problematika detekce dopravního značení, která má stále prostor pro zlepšení. S nárůstem výpočetního výkonu velmi vzrostlo použití hlubokých neuronových sítí pro různé úlohy, zejména pak pro zpracování obrazu, kde se konvoluční sítě staly standardem pro velkou část úloh. Proto je vhodné pokusit se řešit problém detekce značek pomocí těchto sítí.

Nevýhodou drtivé většiny metod používajících konvoluční sítě k detekci objektů a zároveň dosahujících dobrých výsledků, jsou vysoké nároky na výpočetní výkon (což znesnadňuje jejich použití v přenosných či vestavěných zařízeních) a dlouhá doba zpracování jednoho snímku – oba zmíněné systémy však musí dokázat pracovat v reálném čase v kombinaci s velmi dobrou úspěšností detekce. I když se detekce značek může zdát jako jednoduchá, existuje velké množství faktorů, které tuto úlohu znesnadňují (různé světelné podmínky, částečné překrytí, malá velikost, nízká kvalita, atd.). Proto je potřeba dostatečně velká datová sada, obsahující co největší množství i těchto negativních faktorů. Proces anotace velkého

počtu snímků je ovšem velmi náročný na čas (mj. může vzniknout chybná anotace). Ideálním řešením by tedy byl úspěšný systém, který by dokázal pracovat v reálném čase na zařízení s omezeným výpočetním výkonem a byl schopen se naučit dostatek informací i ze syntetických dat.

Za účelem dosažení zpracování v reálném čase na běžně dostupných grafických čipech se zachováním dobré úspěšnosti byl pro práci vybrán systém YOLOv3 verze tiny v kombinaci s neuronovou sítí *Darknet*, zvládající 1457 BFLOPS za sekundu [1, 2]. Pro trénování i vyhodnocení byla použita Belgická datová sada, obsahující 9006 plných[1] snímků se značkami rozdělenými do 108 tříd [3]. Generování datových sad pro trénování neuronových sítí je aktuálně velmi diskutovaným tématem a lze využít více způsobů v závislosti na typu generovaných dat. Protože je systém YOLO trénovaný na plných snímcích a dopravní značky nabývají poměrně jednoduchých tvarů, byl využit způsob umisťování dopravních značek do plných snímků z prostředí městské zástavby.

Nejlepší model pro detekci značek trénovaný na reálných datech dosáhl úspěšnosti **63.4**% mAP (*mean average precision*, viz 6). Ačkoli je tato úspěšnost téměř dvakrát vyšší, než se podařilo dosáhnout v práci [4] (pomocí systému Tiny YOLO první verze), tak je o více než 20 bodů mAP nižší než při použití SSD v práci [5]. Model trénovaný na syntetických datech dosáhl nejlepší úspěšnosti **82.3**% mAP. Rychlost vyhodnocení snímku **40.4***ms* (∼25 snímků za sekundu) je dostatečná pro zpracování v reálném čase.

## 2. Související práce

Porovnání výsledků jednotlivých prací není jednoduché, protože se často liší datovými sadami a použitými metrikami pro vyhodnocení.

Jak plyne z porovnání detekčních metod dopravních značek [6], dříve byla tato úloha řešena pomocí segmentace obrazu na základě barvy v kombinaci s detekcí geometrických útvarů či rohů [7]. V práci [8] se autoři zabývali rychlou metodou detekce tvarů založenou na Houghově transformaci přímo pro detekci dopravních značek. Výsledkem byla 95% úspěšnost detekce a metoda je navíc invariantní vůči natočení značky a pracuje v reálném čase.

Metoda hojně využívaná pro detekci značek je extrakce příznaků z obrazu. Před příchodem konvolučních neuronových sítí byl přístup s použitím histogramu orientovaných gradientů (dále jen HOG) v kombinaci s SVM (*Support vector machine*) označován za tzv. *state of the art* detekce objektů [4]. V práci [9] využívající kombinaci HOG a SVM se podařilo získat výsledky detekce značek s hodnotami *precision* i *recall* blížícím se k jedné, ale vyhodnocení jednoho snímku trvalo déle než vteřinu. Další možností je použití Haarových příznaků [10], v kombinaci s konvolučními neuronovými sítěmi pro klasifikaci se povedlo dosáhnout výsledků taktéž s hodnotami *precision* i *recall* blížícím se k jedné a zároveň zpracování v reálném čase.

V posledních letech přitáhlo velkou pozornost hluboké učení a standardem pro detekci objektů se tak staly konvoluční sítě. Práce [11] prokázala, že jejich použití pro detekci objektů může vést ke dramatickému zvýšení úspěšnosti detekce. Výborných výsledků detekce objektů dosáhla metoda R-CNN, která kombinuje návrh kandidátních oblastí (selektivní hledání [12]) s konvolučními sítěmi a řadí se tak do třídy dvou-krokových metod. Upravená verze této metody, Faster R-CNN, byla použita v práci [13] pro detekci značek a dosáhla úspěšnosti 34.4% mAP.

Pro rychlejší, ale stále přesnou detekci vznikla třída tzv. *single-shot* detektorů, kam spadá YOLO a SSD (*Single shot detector*). Ty provádí detekci i klasifikaci pomocí jedné konvoluční sítě. Systém Tiny YOLOv1 byl pro detekci značek použit v práci [4], kde se autorům podařilo dosáhnout úspěšnosti 33.8% mAP na Belgické datové sadě dopravních značek s rychlostí 100 snímků za sekundu. SSD bylo použito v práci [5], kde se autoři snažili o přesný odhad okrajů dopravních značek a dosáhli úspěšnosti kolem 85% mAP při rychlosti 7 snímků za sekundu.

Z dosažených výsledků zmíněných prací vyplývá, že pro úlohu detekce dopravních značek stále dosahují lepší úspěšnosti systémy využívající extrakci příznaků v kombinaci s klasifikátorem, ale konvoluční sítě je pomalu dohánějí jak z hlediska úspěšnosti, tak rychlosti.

## 3. You only look once

Systém YOLO přistupuje k detekci objektů jako k jednotné regresi přímo od pixelů snímku k souřadnicím ohraničujících boxů a pravděpodobnostem tříd, a to za pomoci jediné konvoluční neuronové sítě. Základní verze YOLO pracuje s rychlostí 45 snímků za sekundu a odlehčená verze použitá v této práci na více než 150 snímcích za sekundu na grafickém čipu Titan X. YOLO dosahuje více než dvojnásobek mAP něž ostatní detektory, které dokáží pracovat v reálném čase. YOLO je speciální systém, který je trénovaný na plných snímcích, což mu umožňuje se naučit i kontext, ve kterém se dané objekty nachází a snižuje tak pravděpodobnost detekce pozadí [14, 15, 1].

[1] Neořezané snímky plného rozlišení.

### 3.1 Princip fungování systému

Systém rozděluje vstupní snímek na mřížku o velikosti $S \times S$ buněk. Pokud se střed objektu nachází v buňce, pak je tato buňka zodpovědná za detekci daného objektu. Každá z těchto buněk predikuje $B$ ohraničujících boxů a zároveň také jistotu, s jakou se v daném boxu nachází objekt (tzv. *objectness*). Dále predikuje $C$ podmíněných pravděpodobností (zda se v boxu nachází objekt) jednotlivých tříd. Při testování se násobí podmíněné pravděpodobnosti tříd s hodnotou *objectness* (1).

$$\mathrm{PR}(\mathrm{Class}_i \mid \mathrm{Object}) * \mathrm{Pr}(\mathrm{Object}) * \mathrm{IoU}^{\mathrm{truth}}_{\mathrm{pred}} = \mathrm{PR}(\mathrm{Class}_i) * \mathrm{IoU}^{\mathrm{truth}}_{\mathrm{pred}} \quad (1)$$

To udává pravděpodobnost výskytu jednotlivých tříd v jednotlivých boxech, a také jak dobře predikovaný box ohraničuje daný objekt [14].

### 3.2 Ztrátová funkce

Ztrátová funkce slouží k ověření, jak dobře systém odpovídá trénovací sadě a lze ji rozdělit na pět částí, penalizující systém za tři typy chyb [16, 14].

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \quad (2)$$

$$+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \quad (3)$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} (C_i - \hat{C}_i)^2 \quad (4)$$

$$+ \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{noobj} (C_i - \hat{C}_i)^2 \quad (5)$$

$$+ \sum_{i=0}^{S^2} 1_{i}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \quad (6)$$

kde $1_i^{obj} = 1$ pokud se v buňce $i$ nachází objekt, jinak 0. Dále pak $1_{ij}^{obj} = 1$ udává, že $j$-tý ohraničující box je *zodpovědný* za detekci v rámci buňky $i$.

- **Chyba klasifikace** – Pokud se v buňce nachází objekt, tato chyba je počítána jako *squared-error* podmíněné pravděpodobnosti třídy, a to pro všechny třídy pomocí (6). $\hat{p}_i(c)$ udává pravděpodobnost výskytu třídy $c$ v buňce $i$.
- **Chyba lokalizace** – Tato chyba udává, jak dobře byl objekt lokalizován vůči *ground-truth* pomocí (2) a (3). $\lambda_{coord}$ zvyšuje váhu chyby.
- **Chyba jistoty predikce** – Pokud je v boxu detekován objekt, tak se tato chyba počítá pomocí (4), jinak (5). Většina boxů neobsahuje objekt, což způsobuje nevyváženost, a proto $\lambda_{noobj}$ snižuje váhu chyby při detekci pozadí.

### 3.3 Architektura

Architektura základní verze systému YOLO se skládá ze 24 konvolučních vrstev následovaných dvěma plně-propojenými vrstvami. Vstupní vrstvy slouží k extrakci vlastností, plně-propojené vrstvy k predikci ohraničujících boxů a pravděpodobností tříd [14].

### 3.4 Odlehčená verze

Odlehčená verze, zvaná Tiny či Fast YOLO se skládá z méně konvolučních vrstev (9 oproti 24) a také méně filtrů v těchto vrstvách, než základní verze. Ostatní parametry jsou zachovány.

## 4. Generátor datových sad

Za účelem vytvoření dostatečně velké datové sady s plnými snímky, a také pro účely porovnání modelů trénovaných na reálných a syntetických datech byl vytvořen generátor datových sad. Systém YOLO je trénován na plných snímcích, proto byl pro generování zvolen postup umisťování značek (na náhodné pozice) do plných snímků, tzv. pozadí, které pochází z prostředí městské zástavby. V těchto snímcích se nesmí vyskytovat žádné jiné dopravní značky, jelikož by jim chyběla anotace a negativně by ovlivňovaly proces trénování. Nástroj pro generování byl implementován v programovacím jazyce `C++` za použití knihovny `OpenCV`. Pro generování byly využity dva následující způsoby.

### 4.1 Použití vyřezaných značek

Tento způsob je založen na použití reálných obrázků značek vyřezaných z plných snímků, určených pro trénování klasifikátorů. Těchto značek je velké množství, ale nevýhodou pro toto použití je, že po vložení do snímku pozadí kolem těchto značek vzniká výrazná přestupná hrana, jak lze vidět na obrázku 1 vlevo. Tato hrana negativně ovlivňuje proces trénování a značně snižuje úspěšnost. Řešením je použití tzv. *Poisson blending*, které kombinuje gradienty a intenzity barev obou obrázků [17]. Příklad aplikace techniky *Poisson blending* lze vidět na obrázku 1 vpravo.

### 4.2 Použití transparentních značek

V této metodě byly používány přesně ořezané dopravní značky malého počtu, obsahující alfa-kanál. Výhodou tohoto přístupu je, že kolem značky nevzniká přestupná hrana. Umisťování pouze tohoto malého počtu značek do pozadí by však způsobilo, že by

**Obrázek 1.** Porovnání vložení ořezané dopravní značky do pozadí. **Vlevo**: bez použitím techniky *Poisson blending*. **Vpravo**: Po aplikaci techniky.

**Obrázek 2.** Příklad vygenerovaných dat (s různými světelnými podmínkami) pomocí transparentních značek a aplikace efektů upravujících vzhled značky.

**Obrázek 3.** Aplikace všech efektů používaných generátorem datových sad na značce typu *stop*, vyřezané z reálného snímku. Originál vlevo nahoře.

se snímky začaly velmi brzy opakovat, a proto je před vložením do snímku provedena jejich syntetická úprava pomocí různých *efektů*. Při aplikaci efektů je důležité, aby zůstal zachován realistický vzhled značek. Jednotlivé efekty lze vidět na obrázku 3. Pravděpodobnost, s jakou se daný efekt aplikuje, i rozsah aplikovaných hodnot byly pečlivě vybírány a upravovány jak pomocí iterativně vyhodnocovaných experimentů (změny pravděpodobnosti aplikace a aplikovaných hodnot), tak inspirací vzhledu značek z reálného světa. Počet aplikovaných efektů na jednu značku není omezen. Výsledek lze vidět na obrázku 2.

# 5. Trénování detektoru značek

Před samotným trénováním je důležité provést nastavení neuronové sítě pomocí konfiguračního souboru. V tomto souboru lze nastavit následující hodnoty. Počet filtrů, udávající počet konvolučních jader v dané vrstvě (nastaveno na `filters = (classes + 5) * 3`). Aktivační funkce, v téměř všech konvolučních vrstvách byla použita *Leaky ReLu*, ve zbylých dvou (před *yolo* vrstvami) lineární. Pokud je hodnota `random` nastavena na 1 (použito vždy), mění daná vrstva náhodně velikost snímku. Důležitá je velikost vstupní vrstvy – čím větší, tím lepších výsledků detekce bylo dosaženo, ale také se prodloužila doba detekce, jak lze vidět v porovnání na obrázku 6. Další hodnoty jako např. velikost dávky, velikost kotev, masky, apod.

## 5.1 Datová sada

Pro trénování na syntetických datech bylo potřeba zjistit optimální počet generovaných značek na třídu. V tabulce 1 lze vidět závislost počtu syntetických snímků na výsledné úspěšnosti. Jak lze z tabulky vyčíst, mezi 200, 500 a 1000 snímky se vyskytují velké skoky v úspěšnosti, zatímco mezi 1000 a 2000 se úspěšnost již téměř nezvýšila, zato se razantně prodloužila doba trénování. Jako optimální počet pro budoucí trénování na syntetických datech tedy vyšlo 1000 značek na třídu. V případě trénování na reálných datech byla síť vždy trénována na $2/3$ počtu snímků z Belgické sady.

**Tabulka 1.** Závislost úspěšnosti YOLOv3-608 na množství syntetických snímků na třídu.

| Počet snímků | Úspěšnost v mAP |
|---|---|
| 200 | 63.2% |
| 500 | 77.9% |
| **1000** | 86.3% |
| 2000 | **87.6%** |

## 5.2 Experimenty

Poté co bylo dosaženo dobré úspěšnosti a nedařilo se ji zvýšit za pomoci standardních metod, nastal čas provést řadu experimentů, které by mohly zajistit lepší parametry výsledného modelu.

### Smíchání reálné se syntetickou datovou sadou

Při trénování pouze na syntetické datové sadě se může konvoluční síť naučit vlastnosti vyskytující se pouze u syntetických snímků a nedokázat poté správně generalizovat na reálných snímcích. Proto bylo provedeno několik pokusů s přimícháním malého počtu reálných snímku k syntetické datové sadě. Tato malá sada reálných snímků by měla "usměrňovat" trénování správným směrem a výsledný model by poté měl dosáhnout lepší úspěšnosti na reálných snímcích. Při přidání cca. 30 reálných snímků k 1000 syntetickým snímkům na třídu došlo ke zvýšení mAP o 2%.

### Hard negative mining

*Hard negative mining* je technika postupného doplňování trénovací datové sady snímky, ve kterých bylo chybně interpretováno pozadí jako některý z

detekovaných objektů. Po provedení několika testů se ukázalo, že ačkoli došlo ke snížení počtu falešně pozitivních detekcí, což bylo cílem, snížil se i počet správných detekcí a úspěšnost klesla o 3.5% mAP z důvodů přetrénování.

#### Binární model

Model XNOR-net pracuje jak se vstupem, tak s vahami ve formě binárních čísel. Tento model se používá pro zvýšení rychlosti na CPU a snížení výpočetních nároků, například pro použití v přenosných zařízeních s omezenými zdroji. Při testování s uvedeným modelem došlo k poklesu mAP o více než 20% oproti běžnému modelu pracujícími s reálnými čísly a nevýraznému zvýšení rychlosti detekce.

#### Přidání dalších vrstev do architektury

V modelu Tiny YOLOv3 se vyskytují dvě vrstvy `[yolo]` s kotvami. Protože značky mohou nabývat různých velikostí, pokusem bylo přidání třetí takovéto vrstvy s `mask = 6,7,8` (používající největší *kotvy*) do modelu Tiny. Výsledkem byla úspěšnost obdobná modelu se dvěma vrstvami. Důvodem pravděpodobně bylo, že se v Belgické datové sadě nevyskytuje mnoho velkých značek, které by tato vrstva detekovala.

#### Přepočítání kotev

*Recalculate anchors*, tedy přepočítání "kotev", slouží k výpočtu nových hodnot tak, aby co nejlépe odpovídaly velikosti vstupních snímků. Tato technika má zvýšit průměrné *IoU*, tzn. zajistit lepší úspěšnost detekce. Při přepočítání kotev na velikost vstupních snímků došlo ke zlepšení o 0.5% mAP.

#### Počáteční hodnoty modelu

Autoři systému poskytují několik před-trénovaných vah pro různé modely. Tyto váhy byly před-trénovány na datové sadě *ImageNet* a protože konvoluční síť musí detekci hran a tvarů umět, je lepší využít již před-trénované hodnoty. Při použití před-trénovaných vah dosáhl výsledný model o cca. 2% lepší mAP a při trénování potřeboval o 2000 iterací méně, než při trénování modelu od "nuly". Z tohoto důvodu byly při každém dalším trénování použity před-trénované váhy.

## 6. Vyhodnocení

Pro vyhodnocení úspěšnosti modelů byla použita metrika mAP (*mean average precision*) s $IoU > .5$. Ta udává střední hodnotu AP všech tříd, kde AP je definováno jako plocha pod *precision-recall* křivkou. *Precision* (udávající jak přesné predikce jsou) na ose $y$ a *recall* (udávající kolik ze všech pozitivních případů bylo správně predikováno) na ose $x$ [18].


**Obrázek 4.** Příklad často se vyskytujících nesprávných (falešně positivních) či chybějících (falešně negativních) detekcí.


**Obrázek 5.** Správné detekce i za zhoršených podmínek, jako například částečné překrytí značky, malá velikost, poškození a nevhodné světelné podmínky.

Při trénování modelu na reálná data byla Belgická datová sada rozdělena na dvě části. Větší částí sady bylo provedeno trénování modelu a na zbytku provedeno vyhodnocení. Trénování na syntetických datech umožnilo použít celou Belgickou datovou sadu pouze pro vyhodnocení úspěšnosti.

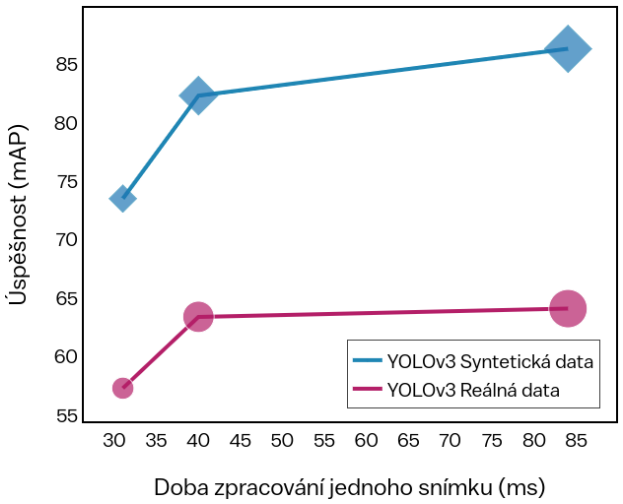
### 6.1 Kvalitativní vyhodnocení

Pokud je značka dobře viditelná a bez znatelného poškození, model ji ve většině případů správně detekuje s 90-100% jistotou. Problém modelu dělají značky velmi malé, pod velkým úhlem, poničené či vybledlé, a také občasná záměna pozadí za objekt (pokud má objekt podobný tvar a barvu), jak lze vidět na obrázku 4. Selhání potlačení nemaximálních hodnot, tzn. některá ze značek je detekována vícekrát, je nejméně často se vyskytující chybou, většinou pouze pokud je značka velmi blízko kameře. I přes to, že má YOLO problémy s dokonalou lokalizací objektů, predikované boxy ve většině případů ohraničují celou značku a to velmi přesně. Příklad několika detekcí za zhoršených podmínek lze vidět na obrázku 5.

### 6.2 Kvantitativní vyhodnocení

K vyhodnocení úspěšnosti jsem využil repozitář *mAP* [19], který jsem doplnil o vykreslení ROC (*Receiver operating characteristic*) křivky a analýzu chyb.

Výsledné modely měly poměrně vysoký počet falešně pozitivních predikcí, proto bylo vhodné zjistit, co dělá systému problém. Poté bylo možné navrhnout řešení a dosahovat tak lepších výsledků. Použil jsem metodiku analýzy chyb popsanou v práci [20]. Predikce je buď to správná, nebo je zařazena do jedné z následujících tříd [14].




**Obrázek 6.** Porovnání závislosti úspěšnosti na době zpracování jednoho snímku třech modelů (zleva) YOLOv3-320, YOLOv3-416 a YOLOv3-608 trénovaných na syntetických a reálných datech. Zvětšující se velikost bodů reprezentuje velikosti vstupní vrstvy modelů. Z grafu lze vidět, že čím větší vstupní vrstva se použije, tím lepších výsledků detekce je dosaženo, ale také se tím prodlouží doba detekce. Za nejvhodnější lze tedy považovat prostřední model YOLOv3-416, který poskytuje nejlepší kompromis (tzv. *tradeoff*) mezi úspěšností a časem. Výsledky byly získány na GeForce 840M.

- **Správná predikce** – správná třída, IoU $> 0.5$
- **Chyba lokalizace** – správná třída, IoU $\in (.1, .5)$
- **Podobné** – podobná třída, IoU $> .1$
- **Ostatní** – nesprávná třída, IoU $> .1$
- **Detekce pozadí** – jakákoli detekce s IoU $< .1$

Více-násobná detekce jednoho objektu se dá také považovat za chybu (selhání potlačení nemaximálních hodnot), a proto bylo vyhodnocení doplněno o třídu vícenásobné detekce. Naopak kvůli nedostatku času nejsou detekce nesprávných tříd rozděleny do kategorií *podobné* a *ostatní*. Vizualizovanou distribuci chyb obou modelů lze vidět na obrázku 7.

## 7. Shrnutí

Práce měla za úkol použít moderní architekturu konvoluční neuronové sítě pro detekci dopravních značek za účelem získání dobrých výsledků detekce v kombinaci s co možná nejkratší dobou zpracování. Dále bylo cílem vytvořit generátor syntetických datových sad a provést porovnání modelů trénovaných na reálných a syntetických datech a zjistit, zda jsou syntetická data pro podobné účely vhodná.

Nejlepším výsledkem získaným při trénování modelu na reálných datech bylo **63.4**% mAP. Použití syntetických dat přineslo podstatně lepší úspěšnost detekce **82.3**% mAP. Oběma modelům trvá vyhodnocení jednoho snímků v průměru **40.4***ms* na průměrně výkonném grafickém čipu Nvidia GeForce 840M. Při testování vyhodnocení na sdíleném clusteru s grafickým čipem Nvidia GTX 1080 Ti byla výsledná doba v průměru **3.9***ms* a na CPU Intel i5 bylo dosaženo průměrné rychlosti **1263***ms*.

Přínosem této práce je skutečnost, že ačkoli syntetická data nejsou pro trénování konvolučních sítí lepší než reálná, lze pomocí nich dosáhnout kvalitních výsledků bez nutnosti anotace tisíce snímků. Dále bylo potvrzeno, že Tiny YOLO třetí verze sice nedosahuje nejlepších výsledků detekce, ale zato velmi vysoké rychlosti zpracování snímků, což ze systému dělá zaslouženého konkurenta *state of the art* metod.

V budoucnu by bylo možné rozšířit generátor datových sad o 3D analýzu pozadí a dopravních značek, za účelem dosažení vyšší realističnosti syntetické datové sady a hlubší analýzu trénování na syntetická data obecně. Vzhledem k faktu, že YOLO má problém s detekcí malých objektů, bylo by vhodné provést podobné porovnání detekce značek i u ostatních metod (SSD, Faster RCNN, apod.).

## 8. Poděkování

Zde bych rád poděkoval svému vedoucímu práce prof. Ing. Adamu Heroutovi, Ph.D za cenné rady, podnětné připomínky a vstřícnost při konzultacích. Dále za přístup k výpočetním a úložným zařízením ve vlastnictví stran a projektů, přispívajících k Národní Infrastruktuře Metacentrum v rámci programu "Projects of Large Research, Development, and Innovations Infrastructures" (CESNET LM2015042).

## Literatura

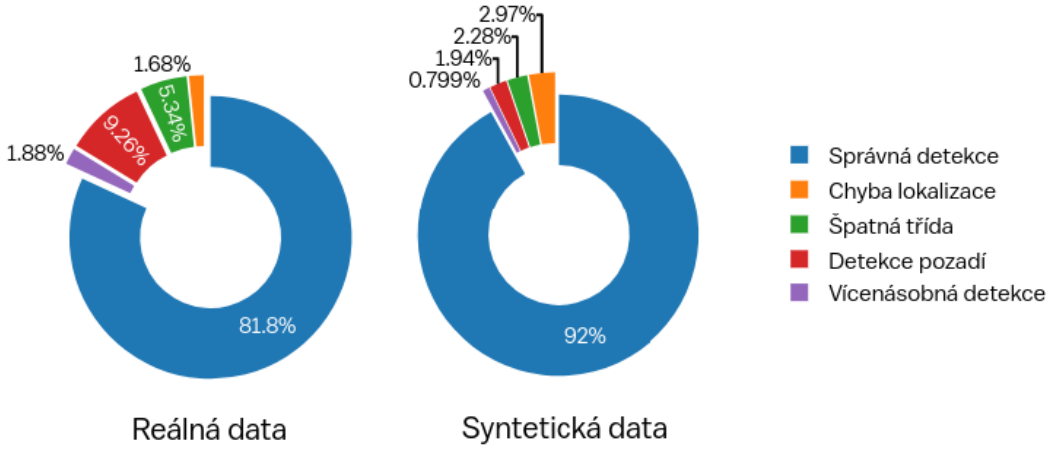
[1] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv*, 2018.

[2] Joseph Redmon. Darknet: Open source neural networks in c. http://pjreddie.com/darknet/, 2013–2016.

[3] Karel Zimmermann Radu Timofte and Luc van Gool. Multi-view traffic sign detection, recognition, and 3d localisation. In *Journal of Machine Vision and Applications (MVA 2011)*, Dec 2011.

[4] Royce Cheng-Yue Chung Yu Wang. Traffic sign detection using you only look once framework. *Convolutional Neural Networks for Visual Recognition CS 231n*, 2015.

[5] H. S. Lee and K. Kim. Simultaneous traffic sign detection and boundary estimation using convolutional neural network. *IEEE Transactions on*




**Obrázek 7.** Analýza chyb dvou modelů, trénovaných na reálných a syntetických datech. Model trénovaný na reálných datech (**vlevo**) má větší problém se záměnou pozadí s objekty a také nesprávné určování tříd značek. To je zapříčiněno nedostatečným množstvím informací o některých třídách (malou velikostí datové sady). Naopak má nižší chybovost lokalizace, což znamená že dokáže predikované boxy lépe zarovnat na detekovanou značku. Pokles v úspěšnosti mAP zapříčiňují také falešně negativní vzorky, které do grafu nejsou zahrnuty.

*Intelligent Transportation Systems*, 19(5):1652–1663, May 2018.

[6] Karla Brkic. An overview of traffic sign detection methods. *Department of Electronics, Microelectronics, Computer and Intelligent Systems Faculty of Electrical Engineering and Computing Unska*, 3:10000, 2010.

[7] A. de la Escalera, L. E. Moreno, M. A. Salichs, and J. M. Armingol. Road traffic sign detection and classification. *IEEE Transactions on Industrial Electronics*, 44(6):848–859, Dec 1997.

[8] G. Loy and N. Barnes. Fast shape-based road sign detection for a driver assistance system. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 1, pages 70–75 vol.1, Sep. 2004.

[9] G. Wang, G. Ren, Z. Wu, Y. Zhao, and L. Jiang. A robust, coarse-to-fine traffic sign detection method. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–5, Aug 2013.

[10] Lotfi Abdi and Aref Meddeb. Deep learning traffic sign detection, recognition and augmentation. In *Proceedings of the Symposium on Applied Computing*, pages 131–136, 2017.

[11] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013.

[12] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[13] Z. Zuo, K. Yu, Q. Zhou, X. Wang, and T. Li. Traffic signs detection based on faster r-cnn. In *2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pages 286–288, June 2017.

[14] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[15] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016.

[16] Jonathan Hui. Real-time Object Detection with YOLO, YOLOv2 and now YOLOv3. [online], Naposledy navštíveno 24. 3. 2019.

[17] Andrew Blake Patrick Pérez, Michel Gangnet. Poisson image editing. *ACM Transactions on Graphics (TOG)*, 22(3):313–318, July 2003.

[18] Jonathan Hui. mAP (mean Average Precision) for Object Detection. [online], Naposledy navštíveno 23. 2. 2019.

[19] João Cartucho. map. https://github.com/Cartucho/mAP, 2019.

[20] Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. Diagnosing error in object detectors. *Computer Vision–ECCV 2012*, pages 340–353, 2012.