GYMNÁZIUM JANA KEPLERA

Maturitní práce

Předmět: **Informatika**

Šárka Morávková

Detekce chyb v datasetu obrázků pro trénování neuronových sítí

Vedoucí práce: **Emil Miler**

3/2025

Čestné prohlášení

Odevzdáním této maturitní práce na téma Detekce chyb v datasetu obrázků pro trénování neuronových sítí potvrzuji, že jsem ji vypracovala pod vedením vedoucího práce samostatně za použití v práci uvedených pramenů a literatury. Prohlašuji, že jsem při její tvorbě nepoužila nástrojů umělé inteligence jiným způsobem, než jako nástroj pro korekturu drobných chyb v kódu.

V Praze dne 3. 3. 2025

Poděkování

Tímto bych chtěla poděkovat Mgr. Emilu Milerovi za vstřícnou a pohotovou komunikaci a investovaný čas, který mi při vedení této práce věnoval. Dále bych ráda poděkovala Ing. Kláře Janouškové za ochotu, cenné rady a taktéž za veškerý čas, který mi věnovala během psaní této práce. Také bych chtěla poděkovat profesoru Jiřímu Matasovi za poskytnutí tématu a zpětné vazby, Illiovi Volkovovi a Nikitovi Kiselovi za poskytnuté materiály a věcné rady a Kateřině Hanzelkové za opravená data.

Obsah

Slovník pojmů **4**

Úvod **5**

1 Rešerše **7**

2 Navržená metoda detekce chyb a metriky **9**

3 Implementace **13**

4 Instalace a spuštění **17**

5 Výsledky **19**

6 Závěr **22**

Seznam použité literatury **24**

Slovník pojmů

datová sada (dataset) množina obrázků s *popisky* roztříděnými do *tříd*

jistota (confidence) hodnota od 0 do 1, přiřazená k *predikci*, říkající s jakou pravděpodobností je *predikce* správná

model naučená neuronová síť, její kvalitu určuje algoritmus, pomocí kterého se na datech učila a správnost těchto dat

predikce (prediction) *modelem* navržený *popisek* s danou *jistotou* jeho správnosti

popisek (label) krátký text obsahující jedno či více slov/sousloví, synonym, jednoznačně vystihující, co obrázek, ke kterému je přiřazen, zachycuje - zároveň název *třídy*

třída množina obrázků se stejným *popiskem*

Úvod

Neuronové sítě dnes nacházejí uplatnění v širokém spektru aplikací. Od jednoduchých úloh (například doporučovací algoritmy), kde není vysoká přesnost klíčová, až po velmi sofistikované systémy (rozpoznání, zda je houba jedlá), kde i drobné chyby mohou mít zásadní dopad. Proto je velmi důležité spolehlivě měřit jejich chybovost a být schopni posoudit, které změny a přístupy vedou ke zlepšení výsledků. Pro takové vyhodnocení potřebujeme přesná testovací data, na kterých můžeme výsledky modelů měřit. Ze zkušenosti víme, že nějaké procento chyb je prakticky nevyhnutelné, příkladem mohou být úlohy v testech Cermat a Scio, které i po kontrole několika odborníky bývají nejednoznačné. V mnohem větší míře to lze pozorovat v poloautomaticky připravených testovacích datech.

V roce 2009 vyšel v té době největší soubor obrázků, ImageNet. Jedná se o datovou sadu, která obsahuje obrázky roztříděné do kategorií podle toho, co zobrazují. Tato datová sada ukázala potenciál neuronových sítí v oblasti počítačového vidění a následně měla velký vliv na vývoj celého strojového učení. Během let vznikaly nové, větší datové sady a tato je dnes z pohledu trénování neuronových sítí překonaná.

Přesto, že je již 15 let starý, je jeho menší část, ImageNet-1k, stále velmi často využívána k vyhodnocení kvality a porovnání existujících modelů. Jeho velká chybovost je přitom dobře známá a diskutovaná ve velkém množství článků. Dokonce se na něm testují modely s přesností blížící se jeho chybovosti, tudíž jejich přesnost nelze spolehlivě změřit. Pokud nepřejdeme k vytvoření nové přesné sady, a zůstaneme u původních popisků ImageNetu, nebudeme schopni nové modely přesně vyhodnocovat. Zpřesnění ImageNetu řeším ve své práci.

Cílem mé práce je navrhnout algoritmus pro automatickou detekci chyb v ImageNet-1k, což usnadní anotátorům snížit množství chyb v této datové sadě. K tomu jsem napsala program, který podle predikcí modelu, navrženého algoritmu a názvu třídy nalezne potenciálně chybné obrázky a seřadí je podle pravděpodobnosti chyby. Používám model OpenCLIP, který je poměrně nový a podle dostupných zdrojů zatím nevyužitý pro tento účel. Ten porovnávám se starším modelem EfficientNet. Kvalita navrženého algoritmu je vyhodnocena na malé množině dat opravené pečlivým anotátorem.

Výsledkem je zjištění, že tyto modely mohou pomoct anotátorovi při snižování množství chyb ve složitějších třídách ImageNetu tím, že zmenší množinu dat, které bude anotátor kontrolovat. Nová množina obrázků ke zkontrolování bude obsahovat přibližně 70 % chybných (většina obrázků, se kterými se anotátor potká, budou opravdu chybné) na rozdíl od původních 35 % (75 % chyba není). Přibližně 40 % chyb nebylo detekováno.

1 Rešerše

ImageNet

ImageNet je datová sada obrázků rozdělených do tříd. Tyto třídy byly vybrány ze skupin synonymních slov a sousloví, neboli synsetů, hierarchicky uspořádaných do stromů. Touto slovní databází je WordNet [7]. (příklad třídy konkrétního druhu zvířete: 1.1, příklad abstraktnější třídy: 1.2)

ImageNet-1k [6], který byl původně vytvořen pro ImageNet Large Scale Visual Recognition Challange [3], se následně pro svou velikost a rozmanitost stal základem metod pro měření schopností nových modelů. Byl vydán v roce 2012 s trénovací, validační a testovací sadou z nichž první dvě jsou veřejně dostupné a z poslední jsou veřejné pouze obrázky bez popisků. Datová sada obsahuje 1000 tříd zastoupených ve všech sadách. Soupis tříd můžete najít například na stránce https://deeplearning.cms.waikato.ac.nz/user-guide/class-maps/IMAGENET/. Trénovací sada obsahuje 1 281 167 obrázků, validační 50 000 a testovací 100 000. Nejčastěji se pracuje s validační sadou, ale některé články se zabývají i opravami trénovací sady. Validační sada je minimálně z 6 % chybná. [8]

Obrázek 1.1: 64, green mamba

Obrázek 1.2: 506, coil, spiral, volute, whorl, helix

Metody hledání chyb v datové sadě

Confident learning [8] je složitější přístup, který se zaměřuje na určování velikosti šumu v datech a jejich následné profiltrování, čímž získá podmnožinu těchto dat s menšími rozdíly mezi jednotlivými obrázky. Při učení odhaduje množství chyb pro každou třídu zvlášť. Je vhodný pro sady s různým množstvím obrázků v jednotlivých třídách. Sady ImageNetu-1k mají stejné množství obrázků v každé ze tříd.

Snazším přístupem je baseline approach [8], kde předpokládáme, že i model naučený na částečně chybných datech dokáže chyby v těchto datech odhalit. Vezmeme pro každý obrázek predikci modelu s nejvyšší pravděpodobností a považujeme ji za správnou. Pro obrázek označený jako chybný tedy platí, že ho model zařadil do jiné třídy, než do které původně patřil. Článek Understanding and Utilizing Deep Neural Networks Trained with Noisy Labels [1] tuto metodu popisuje detailněji, využívá ji k vylepšení trénovacích dat, ale nezjišťuje její přesnost.

2 Navržená metoda detekce chyb a metriky

Úkolem programu je vyhodnotit, které obrázky z původní datové sady jsou označeny špatně. Obrázky budou považovány za chybné, pokud se první predikce modelu nebude shodovat s původní třídou, do které byl obrázek zařazen.

Program bude postupovat takto: Ze souboru načte predikce zvoleného modelu, seřadí je podle jistoty první predikce, zpracuje navrženou metodou, najde zástupce chybných obrázků a zobrazí je. Výstupem je seznam chybně označených obrázků, jejich ilustrace a vyhodnocení správnosti výstupu, pokud pro něj existují anotátorem opravená data. Program je snadno modifikovatelný pro vyhodnocení dalších modelů a přidání dalších vyhodnocovacích metod.

Vstup

Program má čtyři parametry, všechny jsou volitelné:

- třídu (číslo), ve které bude program chyby hledat (pokud třídu nezvolí, bude vybrána náhodně)
- počet obrázků s největší pravděpodobností chyby, které program zobrazí (pokud počet není zadán, zobrazí se všechny obrázky označené jako chybné)
- model, jehož predikce se využije pro vyhodnocení (pokud model není zvolen, označení bude probíhat podle modelu OpenCLIP)
- zda zobrazit tabulku pro obrázky, které byly vyhodnoceny jako chybné, s daty poskytnutými modelem, na kterých je chybnost vyhodnocena (pokud není zodpovězeno, tabulka se nezobrazí)

Výběr modelu pro hledání chyb

Důležitým faktorem je také výběr modelu, pomocí kterého se snažíme chyby odhalit.

Učení „s učitelem“ a „bez učitele“

V angličtině supervised model (s učitelem) a unsupervised model (bez učitele) se liší typem dat, které model dostane pro učení.

Trénovací datová sada pro model trénovaný s učitelem bude obsahovat obrázky rozdělené do konečného počtu tříd. Mezi obrázky spadající pod stejnou třídu bude model hledat podobnosti, aby až v budoucnu dostane nezařazený obrázek, dokázal určit, ke které třídě má tento obrázek nejblíže. Model si tedy nespojuje názvy s obrázky, pouze řadí nové obrázky k předvytvořeným skupinám na jejichž zástupcích byl učen. Příkladem takového modelu je EfficientNet.

Jiným způsobem se učí modely bez učitele. Například pro modely typu CLIP platí [2], že jejich trénovací datová sada bude obsahovat dvojice obrázek - text. Dat tohoto typu je na internetu velké množství, proto se model může učit na opravdu velké množině, a díky takovému množství dat může zobecňovat, ke kterému textu bude pravděpodobně patřit který obrázek. Takový model při učení umisťuje do pomyslného prostoru data tak, aby si byly navzájem podobné obrázky a texty co nejblíže. Pro toto umisťování časem najde nějaká pravidla, která použije na jemu neznámé obrázky.

Modely typu CLIP je možné použít na různé typy úloh, na rozdíl od modelů s učitelem, které jsou trénovány na jedno konkrétní třídění dat.

Stáří modelu

Vývoj v oblasti neuronových sítí jde stále kupředu a lze předpokládat, že se ubírá směrem k lepším a lepším výsledkům. Výběr jednoho z novějších modelů by mohl zajistit kvalitní výsledky.

Vybrané modely

OpenCLIP je model učený formou bez učitele, který se učil na 400 milionech dvojic.

EfficientNet je model trénovaný na trénovací sadě ImageNetu (model trénovaný s učitelem). Tento model byl následně použit k vytvoření Noisy Student EfficientNet modelu. Jeho učení probíhalo tak, že původní model vytvořil nové popisky, stal se učitelem, na nichž byl v kombinaci s původními popisky natrénován další model (student). A ten opět vytvořil k obrázkům nové popisky, stal se učitelem, na kterých se pak učil další model. Tento postup byl několikrát opakován. Vzniklý Noisy Student EfficientNet dosáhl velmi dobrých výsledků, v porovnání s prvním EfficientNetem byl v některých ohledech až o 20 % lepší. [4]

Modely OpenCLIP, EfficientNet a EfficientNet Noisy Student jsou poměrně nové a zastupují širší škálu metod, jakými lze modely učit, proto je vhodné je použít k účelům této práce.

Vyhodnocení

K vyhodnocení přesnosti použiji tzv. matici záměn. Tato metoda vyhodnocení dat se používá zejména u klasifikačních modelů a v diagnostických testech. Základem je porovnání skutečných a predikovaných výsledků:

Správně pozitivní – Případy, kdy test správně identifikoval přítomnost určitého jevu (obrázek označený jako chybný je opravdu chybný).

Falešně pozitivní – Test chybně identifikoval přítomnost jevu (obrázek označený jako chybný není chybný).

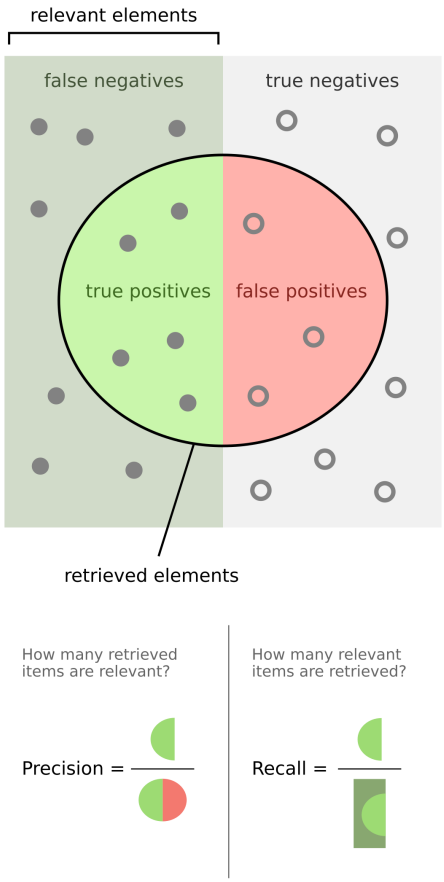
Falešně negativní – Test neodhalil přítomnost jevu, i když ve skutečnosti existuje (obrázek označený jako správný je chybný).

Správně negativní – Test správně označil absenci jevu (obrázek označený jako správný je opravdu správný). Z těchto hodnot se počítají klíčové metriky:

Senzitivita - podíl správně pozitivních případů ze všech skutečně pozitivních případů. Udává schopnost testu správně identifikovat pozitivní případy. Čím vyšší senzitivita, tím méně falešně negativních výsledků.

Specificita (recall) - podíl správně negativních případů ze všech skutečně negativních případů. Udává schopnost testu správně identifikovat negativní případy. Čím vyšší specificita, tím méně falešně pozitivních výsledků.

Přesnost (precision) - podíl správně pozitivních případů ze všech případů, které test



Obrázek 2.1: Ilustrace přesnosti a senzitivity

označil jako pozitivní. Udává, jak často je pozitivní výsledek testu skutečně správný.

Tato metoda pomáhá hodnotit, jak spolehlivý klasifikační model nebo test jsou. Vysoká senzitivita znamená nízký počet falešně negativních výsledků, vysoká specificita znamená nízký počet falešně pozitivních výsledků a vysoká přesnost znamená, že pozitivní výsledky jsou většinou správné.

3 Implementace

Kód je napsán v programovacím jazyce Python. Tento jazyk je snadno čitelný, pro budoucího uživatele bude tedy snazší si program přizpůsobit podle potřeby. Kód napsaný v tomto jazyce není tak efektivní ve využití strojového času jako například kód v jazyce C, ale tento program není příliš rozsáhlý, proto tím výsledná práce nebude znatelně ovlivněna.

Pro větší přehlednost je kód rozdělen do čtyř souborů:

1. **user_interface.py**

   Souboru opakovaně spouští celý program funkcí run, jejímž úkolem je získávat hodnoty vstupních proměnných od uživatele, ověřovat, zda jsou v mezích, které lze splnit, spustit funkci z následujícího souboru a předat jí všechny hodnoty od uživatele.

2. **image_displayer.py**

   Funkce show_images zajiťuje správný postup programu pro zobrazení obrázků, který je tím pro uživatele přehledně zaznamenán na jednom místě: načtení dat modelu pomocí funkce *load_model* z následujícího souboru, vyhodnocení správnosti dat funkcí z následujícího souboru, vyhodnocení počtu obrázků, který se zobrazí, ověření správnosti podle dat ověřených anotátorem, pokud existují, a výsledné zobrazení obrázků.

3. **evaluator.py**

   V tomto souboru se nacházejí všechny vyhodnocovací funkce pracující s daty obrázků. Hlavní z nich jsou find_first_method_results, která vrací pandas DataFrame s informacemi o obrázcích označených jako chybné, (podobnou funkci lze napsat pro další metody vyhodnocení,) evaluate_data srovná výsledky s daty opravenými anotátorem, pokud existují, a uloží je do složky results.

4. **confidence_files_reader.py**

Obsahuje dvě funkce z nichž load_table načítá soubor s predikcemi zvoleného modelu a get_dictionary_for_file přiřadí vybranému souboru předpřipravený „slovník“, pomocí kterého budou moct funkce později hledat proměnné v i DataFramu pro model, jehož soubor vyhodnocených dat obsahuje jinak pojmenované proměnné.

Vstup

Vstup program načítá příkazem input():

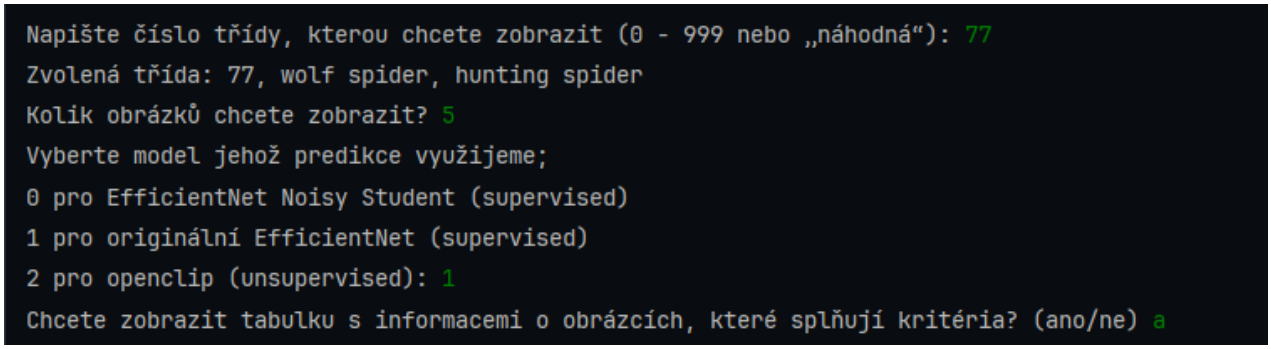
- číslo třídy: 0 - 999, nebo *Enter* -> náhodná třída
- počet obrázků: 0 - 50, nebo *Enter* -> 50 (= všechny označené jako chybné)
- model: 0 -> EfficientNet Noisy Student, 1 -> původní EfficientNet, 2 -> OpenCLIP, nebo *Enter* 2 (= OpenCLIP)
- zda zobrazit výsledky: ano, a -> tabulka se zobrazí, *Enter* nebo cokoliv jiného -> tabulka se nezobrazí

Překlad čísla třídy na název třídy

Soubory ve složce *visualization\loader* umožňují překlad z čísla třídy na její původní název. Zároveň jsou nezbytné pro zjištění správné složky, ve které se nachází obrázky dané třídy. Autoři tohoto kódu jsou zároveň autoři článku Flaws of ImageNet, Computer Vision's Favorite Dataset [5].

Načtení a zpracování dat

Modely vyhodnocená data jsou v souboru CSV, který je ideální pro ukládání tabulek. Knihovna Pandas, která je určena pro jednoduchou a rychlou analýzu dat, umožňuje uživateli s tímto formátem snadno pracovat.



Obrázek 3.1: Příklad spuštěného programu

Zobrazení

Knihovna matplotlib byla původně určena k zobrazování jednoho či více grafů a jejich popisu. Graf/y lze nahradit obrázkem. Hodí se pro snadné zobrazování obrázků s krátkým popisem.

Uložení vyhodnocených dat

Pokud jsou data vyhodnocována na opravené množině, uloží se zároveň do souboru s názvem ve tvaru ***název_modelu***_*results.csv* do složky *results*.

Pro rychlejší nalezení chyb ve zvolených třídách a vypočtení senzitivity, specificity a přesnosti je možné spustit soubor *evaluator.py* nacházející se ve složce *visualization* samostatně. Použijte pro to příkaz *python evaluator.py* (případně *python3 evaluator.py*). Výsledky se pak nachází v souboru *results\all_results.csv*. Před spuštěním je možné zvolit, které třídy budou spočteny a to editací posledních řádků tohoto programu, které následují řádek *if __name__ == "__main__":*. (Proveden bude příkaz na řádku, který nezačíná znakem „#“.)

Ukončení programu

Program ukončíte klávesovou zkratkou *Ctrl + C*.

Obrázek 3.2: Příklad zobrazených obrázků

```
Vyhodnocení pro třídu 77:

          | chybné | správné | celkem
-------------------------------------
označené  |        |         |
jako      | 4      | 1       | 5
chybné    |        |         |
-------------------------------------
označené  |        |         |
jako      | 14     | 31      | 45
správné   |        |         |
-------------------------------------
celkem    | 18     | 32      | 50


senzitivita = 22 %
specificita = 96 %
```

Obrázek 3.3: Příklad výstupních hodnot

4 Instalace a spuštění

Potřebný software

Pro správné fungování programu je potřeba nainstalovat do cílového zařízení Python (3.12.9) a skrze pip (pip3) doinstalovat následující knihovny:

- pandas 2.2.3
- matplotlib 3.10.0
- numpy 2.2.3
- pillow 11.1.0

Stažení projektu z githubu

Ze stránky https://github.com/Salaaat/dataset-error-detection stáněte celou složku dataset-error-detection například takto: Zvolte *Code*, v zobrazeném okně *Download Zip* a stažený soubor rozbalte.

Stažení obrázků validační sady ImageNetu-1k

Z google disku https://drive.google.com/drive/folders/1Zll6_-iJDCWS-qSfm9jF-Z18rHFoQeOC?usp=drive_link stáhněte val.zip a rozbalte ji ve složce *imagenet-1k*. Stažení je možné také z oficiálních stránek https://www.kaggle.com/datasets/sautkin/imagenet1kvalid, ale je poté třeba složky i obrázky přejmenovat podle původní ImageNet notace.

Spuštění

Zkontrolujte, že obsah složky dataset-error-detection nebo složky dataset-error-detection-main\dataset-error-detection-main se shoduje se složkou na githubu: https://github.com/Salaaat/dataset-error-detection a navíc se ve složce imagenet-1k\val nachází neprázdné složky s názvem ve tvaru: **n***číslo*. (celá cesta k jednomu z obrázků by měla vypadat např. takto: *...\dataset-error-detection\imagenet-1k\val\n01770393\ILSVRC2012_val_00001007.JPEG*)

Otevřete příkazovou řádku a pomocí příkazu cd se přesuňte do složky „cesta k repozitáři“ *\visualization*. V této složce spusťte příkaz *python user_interface.py* (případně *python3 user_interface.py*.

5 Výsledky

Anotátorem opravená data

Některé třídy ImageNetu obsahují minimum správných obrázků. Jsou to většinou třídy zvířat, které stěží rozezná a správně určí běžný člověk. Například třída „tchoř černonohý“ obsahuje pouze jeden obrázek, který do této třídy patří, zato většinu třídy tvoří obrázky fretky domácí. K těmto těžkým třídám patří např. 4 třídy lasicovitých a 6 tříd pavouků (a jedna třída „pavučina“, která se objevuje v mnoha obrázcích ze tříd pavouků), jejichž opravené verze mi poskytl pečlivý anotátor. Pro anotátora byla třída „tchoř černonohý“ přejmenována na „fretku domácí“, tudíž pozitivní výsledky pro tuto třídu zachycují fretku domácí.

Výsledky těžkých tříd

EfficientNet je učen na ImageNetu, kde se pravděpodobně v těchto třídách vyskytuje stejně malé množství správných zástupců, jako ve třídách ImageNetu-1k. Očekávané výsledky by měly tedy výrazně ukazovat ve prospěch OpenCLIP modelu, který se učil na mnohem větším množství dat.

Jak je ale vidět v tabulce 5.1 s celkovými výsledky všech těžkých tříd, senzitivita všech modelů se pohybuje mezi 50 - 60 %, specifita kolem 90 % a přesnost kolem 70 %. Za zmínku stojí vysoká specificita 98 % modelu OpenCLIP při rozeznávání tříd pavouků. Tento úspěch by bylo potřeba ověřit na větším množství dat. Výrazně nižší specificita 75 % a přesnost 68 % OpenCLIPu na třídách lasicovitých by mohla naznačovat větší vhodnost EfficientNetu pro tyto třídy, ale pro nedostatek dat to nelze s jistotou říci.

Třídy v tabulce 5.2 lze podle výsledků rozdělit do několika skupin. V některých třídách (72, 73, 75, 76, 356, 357) je výsledek až na malé odchylky stejný pro všechny modely. Tyto výsledky pravděpodobně dělí obrázky na jednoduše (TP, kde se shodne více modelů) a těžce (FN, kde se shodne více modelů) rozpoznatelné. Třída „slíďák mokřadní“ je příkladem těžce

Tabulka 5.1: Celkové výsledky

	efficientnet noisy student	efficientnet původní	openclip	
	97	115	111	správně pozitivní
	31	53	44	falešně pozitivní
	95	77	81	falešně negativní
	327	305	314	správně negativní
	550	550	550	celkem
	50	**59**	**57**	senzitivita [%]
	91	**85**	**87**	specificita [%]
	75	**68**	**71**	přesnost [%]
pavouci	48	56	49	senzitivita [%]
	94	87	**98**	specificita [%]
	80	68	92	přesnost [%]
lasicovití	52	62	65	senzitivita [%]
	91	87	**75**	specificita [%]
	82	79	68	přesnost [%]

rozpoznatelné třídy, což můžeme říct podle podobně vysokých hodnot TP, FP a FN. Zato některé třídy (72, 74, 76, 356) jsou pro modely snáze rozpoznatelné.

Z výsledků je vidět, že jejich kvalita se velmi liší na základě třídy, ne modelu. Mezi modely nelze na těchto specifických datech pozorovat skoro žádné rozdíly. Pro širší zobecnění nemáme dostatečně velkou vyhodnocovací množinu.

Tabulka 5.2: Porovnání jednotlivých modelů

Class Name	Model	TP	FP	FN	TN	Total
žlutý a černý zahradní pavouk (72)	OpenCLIP	2	0	0	48	50
	EffNet NS	2	0	0	48	50
	Orig. EffNet	2	7	0	41	50
Araneus cavaticus (73)	OpenCLIP	15	0	21	14	50
	EffNet NS	15	0	21	14	50
	Orig. EffNet	15	0	21	14	50
křižák obecný (74)	OpenCLIP	20	4	3	23	50
	EffNet NS	15	0	8	27	50
	Orig. EffNet	19	5	4	22	50
snovačka jedovatá (75)	OpenCLIP	4	0	15	31	50
	EffNet NS	4	1	15	30	50
	Orig. EffNet	6	0	13	31	50
sklípkan (76)	OpenCLIP	1	0	0	49	50
	EffNet NS	1	0	0	49	50
	Orig. EffNet	1	1	0	48	50
slíďák mokřadní (77)	OpenCLIP	7	0	11	32	50
	EffNet NS	11	11	7	21	50
	Orig. EffNet	13	13	5	19	50
lasice (356)	OpenCLIP	26	1	2	21	50
	EffNet NS	24	0	4	22	50
	Orig. EffNet	26	0	2	22	50
norek (357)	OpenCLIP	2	1	14	33	50
	EffNet NS	2	0	14	34	50
	Orig. EffNet	4	0	12	34	50
tchoř tmavý (358)	OpenCLIP	22	2	12	14	50
	EffNet NS	14	3	20	13	50
	Orig. EffNet	17	3	17	13	50
tchoř černonohý (359)	OpenCLIP	8	23	2	17	50
	EffNet NS	6	7	4	33	50
	Orig. EffNet	8	11	2	29	50
pavučina (815)	OpenCLIP	4	13	1	32	50
	EffNet NS	3	9	2	36	50
	Orig. EffNet	4	13	1	32	50
pavouci	OpenCLIP	49	4	50	197	300
	EffNet NS	48	12	51	189	300
	Orig. EffNet	56	26	43	175	300
lasicovití	OpenCLIP	58	27	30	85	200
	EffNet NS	46	10	42	102	200
	Orig. EffNet	55	14	33	98	200

Závěr

Výsledky

Metoda ohodnocení obrázku podle k němu modelem přidělené první predikce dosahuje na složitějších třídách pavouků a lasicovitých použitelných výsledků. Senzitivita všech modelů se pohybuje mezi 50 - 60 %, specifita kolem 90 % a přesnost kolem 70 %. Rozdíly mezi modely nejsou v tomto případě zvlášť výrazné. Pozorovatelné je, že hledání chyb tímto způsobem je různě účinné v závislosti na třídě. Pro zobecnění nemáme dostatek dat.

Možné zlepšení

Kód nekontroluje existenci některých souborů, proto kdyby byl program nainstalován špatně nebo by některé soubory chyběly, program by se při setkání s chybou vypnul bez toho, aby oznámil uživateli, co je špatně (zobrazí se pouze výchozí chybové oznámení).

Volba informací, které bude popisek obrázku při jeho zobrazení obsahovat, by měla být přístupná uživateli jiným způsobem, než změnou kódu.

Popisky obrázků by se neměly překrývat, jako se to v některých případech děje.

Funkce evaluate_multiple by mohla kontrolovat, pro které třídy bylo už vyhodnocení provedeno, a nepočítat jej znovu. Také by mohla pojmenovávat soubory all_results podle tříd, kterých výsledky obsahují.

Hodnota senzitivity, specificity a přesnosti se spočítá velmi podobným způsobem. Jedna funkce by mohla zkrátit zápis tohoto výpočtu v kódu.

Volba třídy by měla být možná i podle názvu třídy, nejen podle čísla třídy.

Další možný postup

Další metoda pro vyhodnocení by mohla vypadat například takto: stanovila by se hranice v procentech např. 30 %, pokud by predikce modelu, která se shoduje s původní třídou, měla větší jistotu než je stanovená hranice, byl by obrázek považován za správný. Tato metoda by umožnila obrázky obsahující více tříd označit za správné pro každou z těchto tříd.

Pro možnost komplexnějších výstupů vycházejících z výsledků programu by bylo vhodné zobrazovat navíc predikce různých modelů pro konkrétní obrázek.

Pohodlnější zacházení s programem by mohl umožnit uživatelem konfigurovatelný soubor, ve kterém by byly uloženy hodnoty proměnných, které uživatel při jednotlivých vstupech nemění.

Seznam použité literatury

[1] Pengfei Chen et al. "Understanding and Utilizing Deep Neural Networks Trained with Noisy Labels". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. Kamalika Chaudhuri a Ruslan Salakhutdinov. Sv. 97. Proceedings of Machine Learning Research. PMLR, zář. 2019, s. 1062–1070. URL: https://proceedings.mlr.press/v97/chen19g.html.

[2] Alec Radford a další. "Learning Transferable Visual Models From Natural Language Supervision". In: *ArXiv* (2021). URL: https://arxiv.org/abs/2103.00020.

[3] Olga Russakovsky a další. "ImageNet Large Scale Visual Recognition Challenge". In: *Arxiv* (2010). URL: https://arxiv.org/abs/1409.0575.

[4] Qizhe Xie a další. "Self-training with Noisy Student improves ImageNet classification". In: *ArXiv* (2019). URL: https://arxiv.org/abs/1911.04252.

[5] Nikita Kisel et al. "Flaws of ImageNet, Computer Vision's Favourite Dataset". In: *arXiv preprint arXiv:2412.00076* (2024).

[6] Standford Vision Lab. "ImageNet Large Scale Visual Recognition Challenge". In: *Arxiv* (2020). URL: https://www.image-net.org/.

[7] George A. Miller. "WordNet: Lexical Database for English". In: *Communications of the ACM Vol. 38, No. 11: 39-41* (1995).

[8] Curtis G. Northcutt, Anish Athalye a Jonas Mueller. "Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks". In: *Proceedings of the 35th Conference on Neural Information Processing Systems Track on Datasets and Benchmarks*. Pros. 2021.