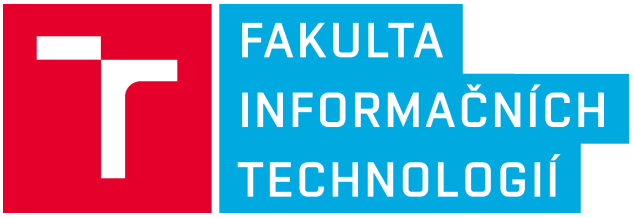



Vysoké učení technické v Brně

ZPO

Zpracování obrazu

---

# Technická zpráva

**Pokus o posouzení realističnosti a kvality rastrových obrázků**

---

*Autor:*
Petr Buchal

*Login:*
xbucha02

21. května 2019

# 1 Úvod

Jako zadání projektu jsem si vybral "Pokus o posouzení realističnosti a kvality rastrových obrázků". Mým cílem je zaměřit se na detekci manipulace s obrázky. To je v dnešní době kvůli umělé inteligenci schopné generovat zcela syntetické obrázky relevantnější než kdy dříve. Projekt jsem se rozhodl vypracovat sám.

# 2 Zadání

Cílem projektu je ověřit, zda snímek vznikl fotografováním nebo jiným procesem, zda případně snímek nebyl "retušován"a zda snímek má dobrou kvalitu, například že je dost ostrý a není v saturaci. Pro realizaci projektu je třeba použít například některý z níže uvedených algoritmů:

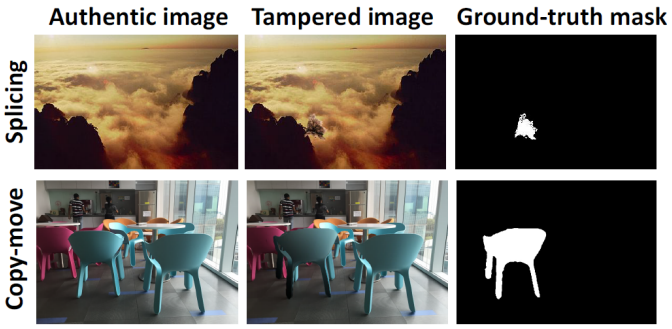
- Zjištění frekvenčního spektra různých "kousků"obrazu a ověření, zda může jít o fotografii (šum, apod.), lze ukázat na příkladu fotografie/nefotografie
- Zjištění, zda obrázek nemá lokálně odlišné charakteristiky (v některém místě rozmazán, někde ostrý, což by svědčilo o nízké kvalitě nebo o manipulaci)
- Ověření, zda v některých oblastech obrazu není saturace nebo "černo". pozor, saturace/černo nemusí být přímo hodnoty 255 nebo 0, ale mohou to být vysoké/nízké konstantní hodnoty

Na základě průzkumu metod, které se tímto tématem zabývají, jsem si vybral

## 2.1 Zpřesnění zadání

V projektu jsem se rozhodl rozpracovat část zadání, která se zabývá ověřováním, zdali byl zkoumaný obrázek retušován. Původně jsem chtěl implementovat metodu, která provádí analýzu obrázku neuronovou sítí z článku "Learning Rich Features for Image Manipulation Detection"[14]. Na konzultaci mi ale bylo doporučeno implementovat klasičtější metodu a tak jsem implementoval "Error Level Analysis"[10]. ELA poskytuje mapu, která označuje pravděpodobná místa výskytu manipulací v obrázku. K ní jsem dále potřeboval implementovat metodu, která z dané mapy bude schopná získat bounding box. Vyzkoušel jsem několik různých přístupů, problémem bylo, že metody většinou našly bounding box i v obrázcích, kde žádná manipulace nebyla. Proto jsem se rozhodl natrénovat klasifikátor k odfiltrování obrázků, kde manipulace není tak, aby se bounding box hledal pouze tam, kde to je relevantní.

Konkrétně jsem natrénoval binární klasifikátor SVM, kterému na vstup přichází feature vektory, které jsou výstupem předtrénované sítě Inception do které jde mapa z ELA, viz článek [12].



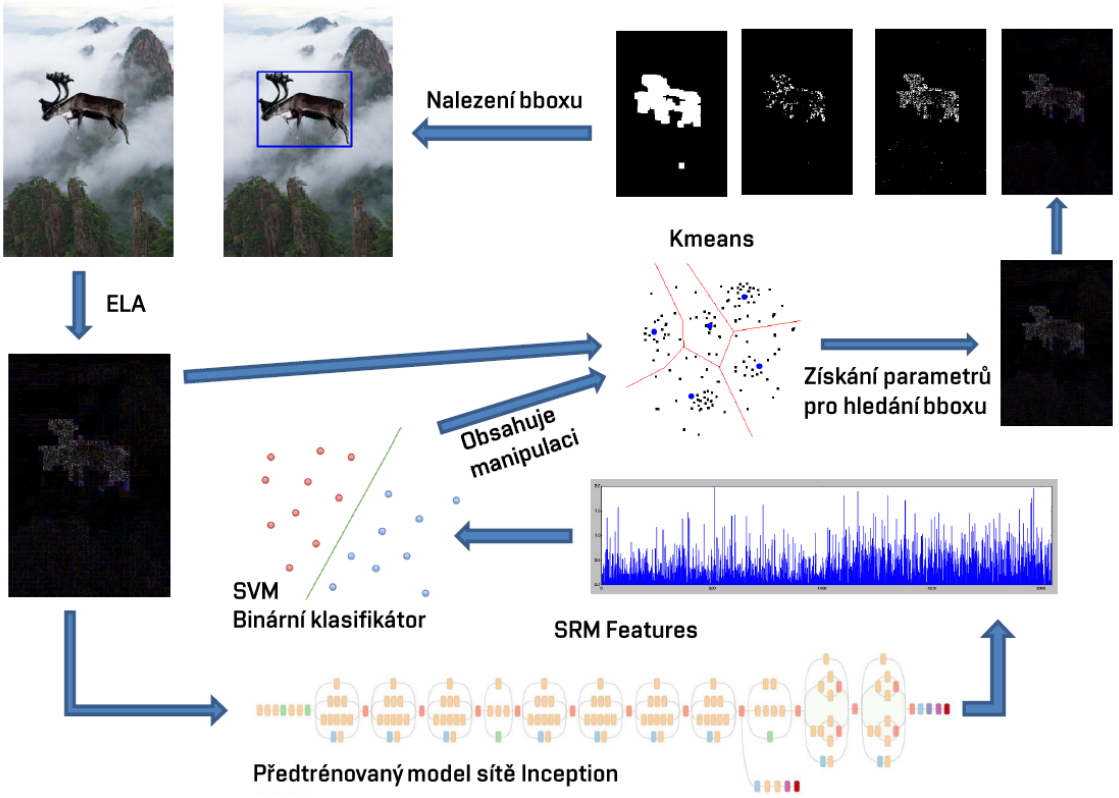

Obrázek 1: Ukázky manipulací obrázků které se snaží projekt rozpoznat, převzato z [14].

# 3 Aplikace a implementace

Projekt jsem se rozhodl implementovat v jazyce Python. Aplikace je spustitelná z příkazového řádku. Pipeline implementované metody pro detekci manipulace je vidět na obrázku 2.

## 3.1 Error Level Analysis

Error Level Analisis je metoda, která se zabývá detekcí manipulovaných míst v obrázcích na základě ztrátové komprese [10]. Metoda zvýrazňuje oblasti obrázku, které mají odlišné úrovně komprese od jeho zbytku. U obrázku JPG by se měly úrovně komprese pohybovat na přibližně stejné úrovni, pokud má tedy obraz v určitém místě odlišnou úroveň komprese, může to indikovat, že s ním bylo manipulováno [2]. Metoda zvýrazňuje oblasti s různou kompresí porovnáním originálního obrázku s jeho komprimovanou verzí (např. 80 % kvality). Metoda ELA má nicméně i nedostatky, úrovně komprese dosahují vyšších hodnot v částech obrazu, kde se vyskytují celistvé jednobarevné plochy (např. obloha), kdežto u ploch s velkou spoustou detailů komprese takových úrovní nedosahuje. To má za následek zvýraznění částí, kde se žádné manipulace nacházet nemusejí [1].




Obrázek 2: Pipeline implementované detekce manipulace v obrazu.

## 3.2 Waveletový soft-thresholding

Na potlačení zvýrazňování částí obrazu, kde manipulace ve skutečnosti není, jsem použil metodu z článku [4] tzv. waveletový soft-thresholding. Metoda se obecně používá k odstranění šumu při čemž zachovává vysoké frekvence a nerozmazává obraz. Problémem je poté nastavení prahu, v prezentovaném článku je volen poloautomaticky, tedy pro jeho výpočet se pro každý obrázek ručně nastaví určitý koeficient a další koeficienty se nastaví na základě charakteristiky obrázku. To ale pro moji potřebu není vhodné a tak jsem se rozhodl zvolit práh experimentálně, více viz 3.3. V článku [4] se dále uvádí, že na základě provedených experimentů bylo zjištěno, že na úspěšnost thresholdingu nemá vliv vybraný typ vlnky. Při soft-thresholdingu se

hodnoty vlnkové koeficienty upravují následovně

$$\rho_{\lambda}^{soft} = \begin{cases} x - \lambda, & x \geq \lambda \\ x + \lambda, & x \leq \lambda \\ 0, & |x| < \lambda, \end{cases}$$

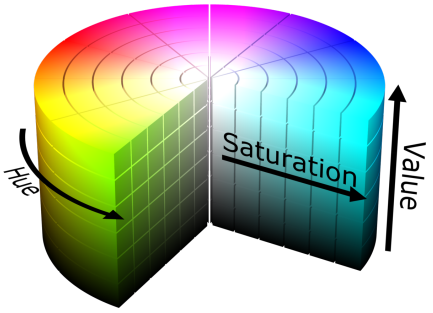
kde $\lambda$ je hodnota prahu [3].

## 3.3 Hledání bounding boxu

Nalezení bounding boxu se skládá z těchto po sobě jdoucích operací. První se provede waveletový soft-thresholding na odstranění šumu vzniklého ELA, dále se provede thresholding, který z ELA mapy vytáhne pouze barevné pixely, protože ty primárně detekují manipulaci, poté se použije mediánové filtr pro odstranění šumu, který v obrázku ještě zůstal, následně se provede morfologická operace otevírání, která zvýrazní podezřelé oblasti, přičemž se následně vybere ta největší a se prohlásí za manipulaci v obraze. První experimenty na hledání bounding boxu měly pevně nastavené koeficienty pro jednotlivé operace, které se na ELA mapě prováděly. To vedlo k dobrým výsledkům na určitých typech obrázků, obecně se ale tyto metody ukázaly jako neúčinné, úspěšnost na datasetu CASIA1 byla kolem 23 %. Proto jsem se rozhodl obrázky klasifikovat do tříd a jednotlivým třídám experimentálně vybrat nejlepší parametry pro prováděné operace. Jako první jsem se pokusil obrázky klasifikovat a přiřadit jim optimální parametry na základě množství šumu, které obsahují, viz článek [5]. To vedlo ke zlepšení úspěšnosti na 32 %. Následně jsem použil shlukovací metodu k-means [7], která zlepšila úspěšnost na 37 %. To ale bylo pravděpodobně způsobené náhodou, jelikož by tento přístup neměl fungovat kvůli prokletí dimenzionality [15], o které jsem se dozvěděl na konzultaci.

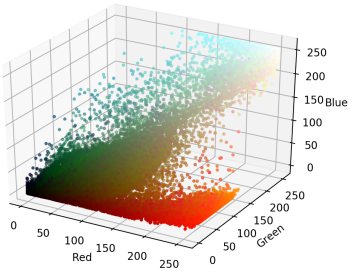
### 3.3.1 Threshold pixelů s určitým barevným rozmezím

Obraz může být reprezentován pomocí několika barevných modelů. Nejčastější model reprezentace je RGB (red, green, blue), OpenCV poté pracuje s prohozenými barevnými složkami a modelem BGR. Ani jeden z těchto modelů ovšem není ideální pro selekci určitého barevného rozmezí barev, viz obrázek 5, kde každá osa reprezentuje jeden z barevných kanálů. Z obrázku vyplývá, že segmentace jednotlivých barev by byla kvůli jejich neseparabilitě komplikovaná. V barevném modelu HSV (obrázek 3) se barevná segmentace provádí daleko lépe, viz obrázek 6, kde osy reprezentují odstín, sytost a jas. Barvy jsou zde více lokalizované a vizuálně separovatelné [9]. V projektu provádím threshold pixelů s určitým barevným rozmezím právě v modelu HSV.

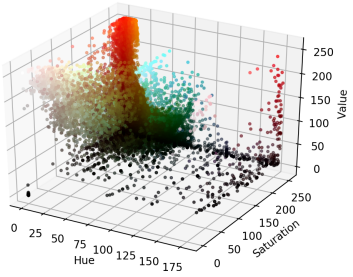



Obrázek 3: Barevný model HSV, kde hue je odstín, saturation je sytost a value je jas, převzato z [11].

Obrázek 4: Originální obrázek, převzato z [9].




Obrázek 5: Reprezentace obrázku pomocí RBG modelu, převzato z [9].




Obrázek 6: Reprezentace obrázku pomocí HSV modelu, převzato z [9].

### 3.3.2 Mediánový filtr

Mediánový filtr se řadí mezi nelineární filtry, protože nesplňuje podmínku

$$f(x_1 + x_2) = f(x_1) + f(x_2),$$

konkrétněji pak jde o tzv. rank-order filtr. Tyto filtry jsou založené na myšlence, že nová hodnota pixelu se vhodně vybere z okolí pro daný pixel. Vyberou se tedy hodnoty z okna o určité velikosti (např. 3x3, 5x5 pixelů atd.), ty se seřadí a ze seřazených hodnot se vybere "vhodná". V případě mediánového filtru se vybere medián ze seřazených hodnot jako nová hodnota pixelu [13].

### 3.3.3 Morfologická operace otevření

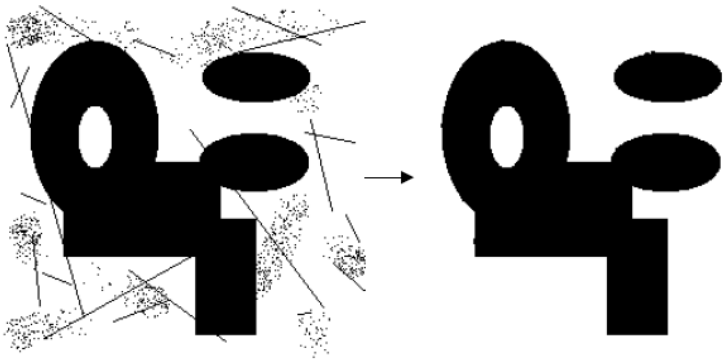
Obrázky lze modelovat pomocí tzv. bodových množin. Morfologickou transformací rozumíme relaci mezi obrazem a typicky menší bodovou množinou tzv. strukturním elementem, který je vztažen k lokálnímu počátku. Samotná morfologická operace poté odpovídá systematickému posunu strukturního elementu po obraze, při čemž výsledek transformace v každé poloze odpovídá relaci [6]. Binární otevření je poté operace, která se skládá ze dvou dílčích operací v konkrétním pořadí - z eroze a dilatace. Erozi lze vyjádřit jako průnik všech posunů obrazu $X$ o vektory $-b$

$$X \ominus B = \bigcap_{b \in B} X_{-b},$$

dilataci následně jako průnik všech posunů obrazu $X$ o vektory $b$

$$X \oplus B = \bigcap_{b \in B} X_{b}.$$

Operace má za cíl odstranit prvky šumu erozí a následně opravit původní tvary významných objektů dilatací, viz obrázek 7.



Obrázek 7: Ilustrace výsledku morfologické operace otevírání, převzato z [6].

### 3.3.4 Určení množství šumu v obraze

Nejprve se provede filtrace Laplaciánovým filtrem

Ten patří k operátorům druhé derivace, které obecně detekují průchody nulou, což je snazší než hledání extrému. Nevýhodou těchto operátorů je příliš velké vyhlazení obrazu, což má za následek ztrátu ostrých rohů. Následně se ze vzorce

| | | |
|---|---|---|
| 1 | -2 | 1 |
| -2 | 4 | -2 |
| 1 | -2 | 1 |

$$noise = \frac{\sqrt{\frac{1}{2} * \pi} * \sum_{y=0}^{height-1} \sum_{x=0}^{width-1} |pixel_{x,y}|}{(6 * (width - 2) * (height - 2))}$$

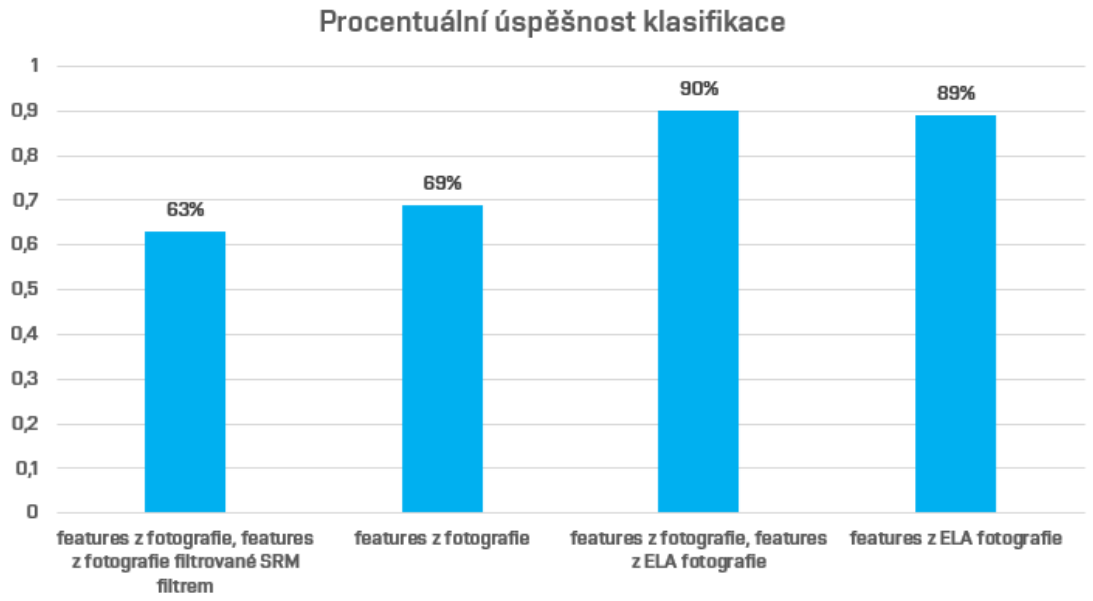
spočítá odhad šumu v obraze [5].

## 3.4 Klasifikátor

Při vytváření klasifikátoru jsem vycházel z článku [12]. V článku se vytváří klasifikátor pomocí předtrénované neuronové sítě Inception a SVM. Neuronová síť byla předtrénovaná na ImageNet datasetu. Na vstup sítě Inception jde obrázek, na výstupu sítě vyleze vektor 2048 features, který se dále pošle na vstup SVM, který už klasifikuje, zdali je na obrázku manipulace nebo nikoliv. Vektor 2048 features je takzvaný bottleneck feature. Jedná se o poslední mapu aktivací před plně propojenými vrstvami v neuronové síti. Vyzkoušel jsem několik přístupů k tomu, co půjde na vstup neuronové sítě Inception, potažmo SVM. V článku [14] se k detekci manipulací používá neuronová síť na jejíž vstup se kromě obrázku posílá ještě jeden obrázek filtrovaný SRM filtrem, který zvýrazňuje šumové artefakty pro další evidenci manipulace. Zkusil jsem tedy na vstup SVM posílat vektor 4096 features, který se skládal z features samotného obrázku a features obrázku filtrovaného SRM filtrem. Dále jsem zkusil na vstup SVM posílat features ze samotného obrázku, obrázku a mapy ELA a ze samotné mapy ELA. Výsledky experimentů jsou vidět v grafu na obrázku 8. Nakonec jsem použil klasifikátor SVM, kterému jde na vstup vektor features vytvořených pouze z mapy ELA. Rozhodl jsem se tak, kvůli srovnatelnému výsledku klasifikátoru s vektorem features z obrázku a mapy ELA na vstupu, ale vyšší rychlosti výpočtu způsobené polovičním množstvím features. Pokud spojím trénovací a testovací data z datasetu CASIA1, dostane se klasifikátor nad 95 % úspěšnost.

## 3.5 Datasety

Data se kterými jsem během projektu pracoval pocházela z datasetu CASIA1 [8], který obsahuje 800 originálních snímků a cca 900 manipulovaných. Manipulované




Obrázek 8: Výsledky experimentu se vstupy jednotlivých klasifikátorů. Zobrazeno na testovacích datech.

obrázky se poté skládají z vytvořených pomocí spojování různých obrázků (splicing) a kopírování a přesouvání částí stejného obrázku (copy-move), viz obrázek 1.

# 4 Závěr

Binární klasifikátor dosahuje úspěšnosti 90 % na testovací sadě, s čímž jsem spokojený. ELA společně s metodou na hledání bounding boxů poté dosahuje úspěšnosti kolem 37 %. Je zde tedy místo ke zlepšení, pravděpodobně vybráním sofistikovanější metody.

# Reference


[1] Image forensics: Error level analysis. https://bit.ly/2VnmBHE.

[2] Tutorial: Error level analysis. https://bit.ly/2VyHtR3.

[3] David Bařina. Integrální transformace obrazu, 2013. https://bit.ly/2LX2MHF.

[4] Daniel C Jeronymo. Semi-automatic wavelet soft-thresholding applied to digital image error level analysis. 01 2016.

[5] John Immerkær. Fast noise variance estimation. *Computer Vision and Image Understanding*, 64(2):300 – 302, 1996.

[6] Ph.D. Ing. Michal Španěl. Matematická morfologie. https://bit.ly/2Ju3073.

[7] Ph.D. Ing. Michal Španěl. Klasifikace a rozpoznávání, 2009. https://bit.ly/2VC8Rsx.

[8] Phat Sovathana. Casia dataset. https://bit.ly/2GcFtDG, Oct 2018.

[9] Rebecca Stone. Image segmentation using color spaces in opencv python – real python, Oct 2018. https://bit.ly/2LZ2v79.

[10] Wikipedia. Error level analysis — Wikipedia, the free encyclopedia. https://bit.ly/2Y5gwBs, 2019. [Online; accessed 02-May-2019].

[11] Wikipedia. HSV — Wikipedia, the free encyclopedia. http://cs.wikipedia.org/w/index.php?title=HSV&oldid=16493801, 2019. [Online; accessed 21-May-2019].

[12] May Yeung. Using tensorflow and support vector machine to create an image classifications engine. https://bit.ly/2GcFtDG, October 2016.

[13] Prof. Dr. Ing. Pavel Zemčík. Nelineární filtry, extrakce příznaků. https://bit.ly/2VCycCD.

[14] Peng Zhou, Xintong Han, Vlad I. Morariu, and Larry S. Davis. Learning rich features for image manipulation detection. *CoRR*, abs/1805.04953, 2018.

[15] Miroslav Čepek. Vytěžování dat: Přednáška 13 – redukce dimenzionality, 2016. https://bit.ly/2JqDCyU.