VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

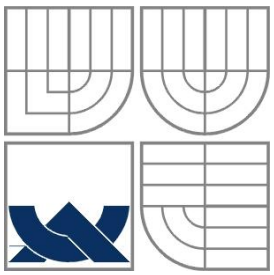
AUTOMATICKÝ ODHAD NADMOŘSKÉ VÝŠKY Z OBRAZU

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

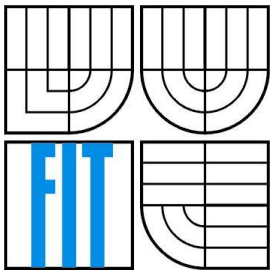
AUTOR PRÁCE
AUTHOR

Bc. JAN VAŠÍČEK

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AUTOMATICKÝ ODHAD NADMOŘSKÉ VÝŠKY Z OBRAZU

ALTITUDE ESTIMATION FROM AN IMAGE

DIPLOMOVÁ PRÁCE
MASTER‘S THESIS

AUTOR PRÁCE
AUTHOR

Bc. JAN VAŠÍČEK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MARTIN ČADÍK, Ph.D.

BRNO 2015

Abstrakt

Tato práce se zabývá automatickým odhadem nadmořské výšky kamery z obrazu. Úlohu jsem řešil pomocí konvolučních neuronových sítí, u nichž využívám schopnost učit se nové příznaky na základě trénovacích dat. Trénovací sada obrazů (dataset), která by obsahovala údaje o nadmořské výšce kamery, nebyla k dispozici a proto bylo nutné vytvořit dataset nový. Schopnosti člověka v dané úloze také nebyly dříve testovány, proto jsem provedl uživatelský experiment s cílem změřit průměrnou kvalitu lidského odhadu nadmořské výšky kamery. Výsledky experimentu, kterého se zúčastnilo 100 lidí ukazují, že člověk je schopen odhadnout nadmořskou výšku z obrazu s průměrnou chybou 879 m. Automatický systém založený na konvoluční neuronové síti dosahuje lepších výsledků než člověk. Průměrná chyba odhadu systému je 712 m. Navržený systém může kromě samotného odhadu nadmořské výšky z obrazových dat nalézt uplatnění také ve složitějších úlohách, jako je vizuální geolokalizace kamery.

Abstract

This thesis is concerned with the automatic altitude estimation from a single landscape photograph. I solved this task using convolutional neural networks. There was no suitable training dataset available having information about image altitude, thus I had to create a new one. To estimate human performance in altitude estimation task, an experiment was conducted counting 100 subjects. The goal of this experiment was to measure the accuracy of the human estimate of camera altitude from an image. The measured average estimation error of subjects was 879 m. An automatic system based on convolutional neural networks outperforms humans with an average elevation error 712 m. The proposed system can be used in more complex scenario like the visual camera geo-localization.

Klíčová slova

Rozpoznávání obrazu, odhad nadmořské výšky, konvoluční neuronové sítě

Keywords

Automatic image recognition, altitude estimation, convolutional neural networks

Citace

Vašíček Jan: Automatický odhad nadmořské výšky z obrazu, diplomová práce, Brno, FIT VUT v Brně, 2015.

AUTOMATICKÝ ODHAD NADMOŘSKÉ VÝŠKY Z OBRAZU

Prohlášení

Prohlašuji, že jsem tento diplomový projekt vypracoval samostatně pod vedením Ing. Martina Čadíka, Ph.D. Další informace mi poskytl Ing. Michal Hradiš. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jan Vašíček
27.5.2015

Poděkování

Chtěl bych poděkovat vedoucímu mé diplomové práce Ing. Martinu Čadíkovi, Ph.D. za pomoc a rady při zpracování této práce. Dále bych rád poděkoval Ing. Michalu Hradišovi za odborné rady během návrhu konvolučních sítí. Výzkum vedoucí k těmto výsledkům vznikl v rámci projektu LOCATE 4SGA8694, který je financován z programu SoMoPro II, spolufinancovaného Evropskou unií a Jihomoravským krajem.

Obsah

1 Úvod

Nadmořská výška je jednou z nejdůležitějších informací v horském prostředí. Označuje svislou vzdálenost mezi konkrétním místem a hladinou moře. Měření nadmořské výšky má mnohaletou historii. V současnosti se údaje o nadmořské výšce používají v mnoha vědních oborech. Mezi ně například patří geologie, meteorologie, výzkum globálních změn klimatu, hydrologie, navigace bezpilotních letounů apod. Standardně se nadmořská výška měří přesnou nivelací k střední hladině nejbližšího moře. Dalším způsobem jak změřit nadmořskou výšku je porovnání polohy daného bodu s matematicky vypočteným elipsoidem WGS 84 [1].

Krajina kolem nás je každodenně zachycena formou fotografií a videozáznamů. Množství pořízených fotografií a videozáznamů neustále roste. Lidé mají přístup k online službám, které dovolují procházet několika miliónové kolekce fotografií a efektivně v nich vyhledávat podle specifických dotazů. Bohužel téměř všechny veřejně dostupné fotografie a videozáznamy neobsahují informaci o nadmořské výšce. Kromě toho, většině z nich chybí zároveň informace o místě jejich pořízení (GPS souřadnice).

Tato práce se zaměřuje na automatický odhad nadmořské výšky z obrazu. Automatická anotace fotografií, která ke každému snímku přesně odhadne nadmořskou výšku, může být využita v řadě aplikací. Mezi ně patří volnočasové venkovní aktivity, turistické aplikace, vzdělávací aplikace, geografické hry nebo klasifikace velkých kolekcí fotografií. V dostupných materiálech se mi nepodařilo najít informace o předchozím výzkumu této problematiky. Z tohoto důvodu je práce vedena jako prvotní výzkum úlohy odhadu nadmořské výšky z obrazu, které jsou pořízený z pohledu člověka. Úlohu řeším pomocí metod počítačového vidění. K tomuto účelu jsem vytvořil unikátní datovou sadu z kolekce fotografií, kterou jsem dostal k dispozici. Dále jsem provedl uživatelský test, abych zjistil, jak si navržené experimenty vedou v porovnání s člověkem.

Populární technikou jak řešit úlohy rozpoznávání obrazu jsou konvoluční neuronové sítě. Tato technika je známá již od roku 1980 [2]. K praktickému využití se však používá až v posledních několika letech, kdy je k dispozici dostatečný výpočetní výkon pro vytvoření kvalitních a robustních sítí, které dosahují lepších výsledků než tradiční techniky používané k řešení obecných úloh rozpoznávání obrazu. Konvoluční sítě bylo možné použít k efektivnímu odhadu nadmořské výšky, který překonává schopnosti člověka na této úloze.

V 2. kapitole této práce jsou vysvětleny dva hlavní způsoby jak přistupovat k úlohám rozpoznávání obrazu. V 3. kapitole jsou popsány podobné úlohy, ze kterých jsem při návrhu systému vycházel. Kapitola 4 popisuje uživatelský test a jeho výsledky. V kapitole 5 následuje návrh a implementace konvolučních sítí pro odhad nadmořské výšky kamery z obrazu. Kapitola 6 se věnuje vyhodnocení dosažených výsledků experimentů.

2 Automatický odhad nadmořské výšky kamery z obrazu

Tato kapitola popisuje způsoby jak řešit úlohu automatického odhadu nadmořské výšky kamery z obrazu. Na začátku kapitoly jsou stručně vysvětleny základní pojmy týkající se počítačového vidění a rozpoznávání obrazu. V další části kapitoly jsou popsány různé způsoby rozpoznávání obrazu, které se dají použít při řešení úlohy. Na konci kapitoly je popsán problém přetrénování a přeučení metod strojového učení, který může vést ke zhoršení výsledků.

2.1 Strojové učení

Strojové učení je jednou z oblastí umělé inteligence zabývající se algoritmy určenými k učení počítačových systémů. Učení je možné definovat jako zlepšení výkonu systému v určitém prostředí díky znalostem, které jsou získány ze zkušeností v tomto prostředí [3]. Cílem algoritmu učení je předpovídat výstupní hodnotu systému pro platný vstup poté, co zpracuje všechna trénovací data [4]. Algoritmus musí být schopný zobecnit trénovací data, aby dokázal správně předpovídat výstupy i pro nové vstupy. Algoritmy strojového učení lze rozdělit do dvou hlavních kategorií:

- **Učení s učitelem** - algoritmy sloužící k učení systému z trénovacích dat, které kromě vstupu obsahují i správný výstup systému. Výstup systému může být spojitá hodnota (regrese) či označení třídy vstupních dat (klasifikace). Během procesu učení se v každém kroku porovnává aktuální výstup se správným výstupem a podle výsledku se upraví vnitřní stav systému.

- **Učení bez učitele** - algoritmy pracují s trénovacími daty, které neobsahují správný výstup systému. Podstatou činnosti učení bez učitele je hledání společných vlastností vstupních dat.

2.2 Úvod do rozpoznávání obrazu

Cílem rozpoznávání obrazu je řadit objekty v obraze do jedné z tříd podle jejich společných vlastností tak, že objekty vzájemně si podobné zařazujeme do stejné třídy. Lze uvažovat například klasifikaci obrazu do podrobnějších tříd, tedy třídy město, hora, poušť atd. Další úlohou může být nalezení hledané spojité hodnoty na základě vstupního obrazu (určení GPS souřadnic z obrazu pomocí regrese atd.). V této kapitole budou nejdříve představeny technické problémy, které úlohy rozpoznávání obrazu provázejí. Dále pak budou popsány dva hlavní přístupy jak řešit rozpoznávání obrazu. Prvním z nich je tradiční způsob založený na metodách pro extrakci příznaků, které byly manuálně vytvořeny odborníky v řešené úloze. Druhým způsobem je deep learning, který se dostává v posledních letech do popředí. V textu bude používán termín deep learning, protože v českém jazyce pro tuto techniku zatím neexistuje ustálený výraz.

2.2.1 Technické problémy

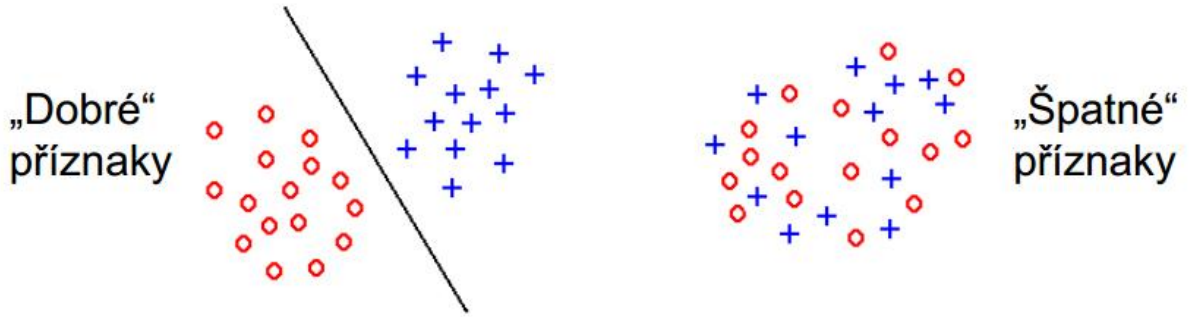
Během vývoje systémů pro rozpoznávání obrazu je potřeba řešit řadu problémů spjatých např. s nedokonalostmi senzorů, či převodem 3D objektů do 2D reprezentace [5]. Mezi hlavní problémy patří:

- **Šum** - šum lze definovat jako náhodné a nežádoucí signály, které zakrývají požadovanou informaci. Zašuměný obraz vypadá jako by byl pokryt nevzhlednými barevnými body. Šum je způsoben nedokonalostmi senzoru nebo nevhodným snímáním obrazu [6].
- **Rozmazání objektu** - během pořizování snímku může dojít k rozmazání některých jeho částí. K tomu může dojít nesprávným nastavením fotoaparátu pro typ snímané scény (dynamická vs. statická scéna) nebo vadou objektivu. Problém při rozpoznávání obrazu to způsobuje pokud dojde k rozmazání důležité části fotografie (charakteristický objekt scény atd.).
- **Osvětlení objektu** - rozpoznávaný objekt by měl být na všech obrazech osvětlen stejným způsobem. Ideální stav je, když je objekt osvětlen tak, že jsou na něm vidět všechny jeho defekty a charakteristické rysy. Tohoto stavu je jen málokdy dosaženo na všech snímcích objektu kvůli vrhaným stínům či změně intenzity slunečního záření v průběhu dne [7].
- **Pozice kamery** - různé pozice kamery během snímání objektu mohou vést k výrazným rozdílům vzhledu objektu. To může vést ke změně velikosti, tvaru či rotaci objektu. Tento problém lze korigovat výběrem vhodného algoritmu pro detekci klíčových bodů [8].

2.2.2 Extrakce příznaků

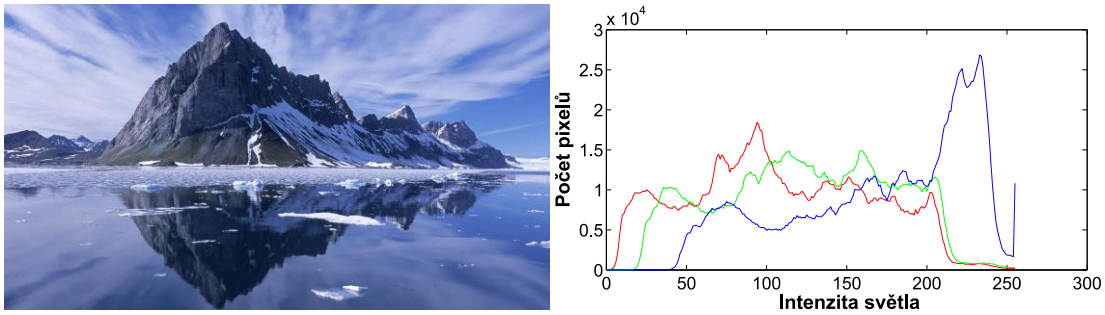
Příznaky se používají pro popis charakteristických rysů obrazu. Obraz obsahuje velké množství nepotřebných dat, například pozadí. Dobré příznaky také musí vhodně rozlišovat cílové třídy/hodnoty uvnitř příznakového prostoru. Proto se příznaky musí vybírat podle charakteru řešeného problému. Na obrázku 2.2 je zobrazen rozdíl mezi vhodnými a nevhodnými příznaky pro klasifikaci ve 2D příznakovém prostoru [9].

Příznaky mohou být buďto globální, tedy počítané pro celý obraz nebo lokální, které se počítají pro lokální regiony obrazu. Jako příklad globálního příznaku lze uvést histogram barev obrazu. Globální příznaky jsou citlivější na šum, překrytí objektu či změnu pozice kamery. Lokální příznaky se typicky počítají ve dvou krocích. Nejprve se v obraze vyhledají klíčové body (rohy, hrany atd.), z jejichž okolí se poté vypočítají deskriptory. V následující části jsou uvedeny příklady metod pro výpočet globálních i lokálních příznaků:



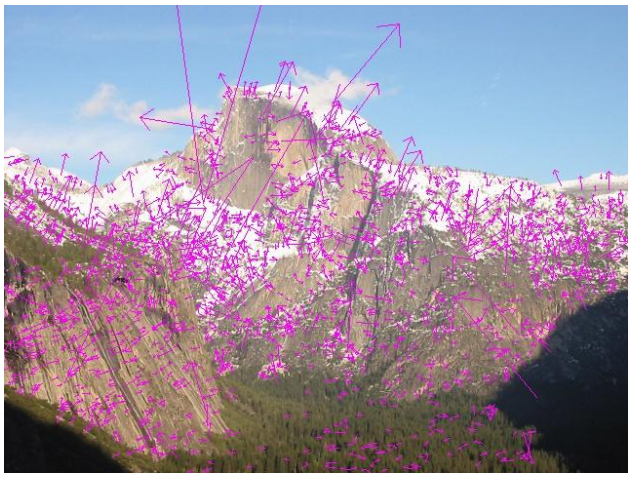
Obrázek 2.1: Porovnání dobrých a špatných příznaků v 2D příznakovém prostoru [4]

- **Histogram barev**
  Jedná se o globální příznak reprezentující distribuci barev v obrazu. Pro digitální fotografie tedy reprezentuje rozložení počtu pixelů (obrazových bodů) v každé úrovni intenzity barvy. Na obrázku 2.2 je zobrazen RGB obraz a jeho histogram barev pro každý barevný kanál.



Obrázek 2.2: Histogram barev obrazu

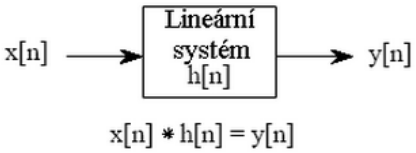
- **SIFT**
  SIFT (Scale-invariant feature transform) je algoritmus určený k detekci klíčových bodů v obraze a vypočítání jejich deskriptorů (viz obrázek 2.3). Tyto deskriptory mohou být použity jako jedinečná reprezentace objektu a umožňují jeho úspěšné rozpoznání. Hlavní výhodou tohoto detektoru je nezávislost na měřítku, odolnost vůči šumu a změně osvětlení. Algoritmus se hodí na porovnávání obrazu pořízených z různých vzdáleností a obsahujících problémy popsané v kapitole 2.2.1 [10].



Obrázek 2.3: SIFT příznaky s vyznačeným směrem gradientů

- **GIST**
  Algoritmus pro výpočet deskriptoru GIST je primárně určen k použití u obrazů scén. GIST je globální deskriptor, při jehož výpočtu vůbec nedochází k segmentaci obrazu (rozlišení objektů v obraze) a zpracování jednotlivých regionů/ objektů v obraze. Cílem je vytvořit nízkoúrovňovou reprezentaci obrazu, která popisuje dominantní prostorové struktury ve scéně [11].

2.2.3 Diskrétní 2D konvoluce



Obrázek 2.4: Použití konvoluce lineárním systémem [12]

Konvoluce je matematický operátor, který kombinuje dvě funkce a vytváří funkci novou. Používá se při analýze odezvy systémů na libovolné signály x[n], známe-li impulsní odezvu systému h[n] (viz obrázek 2.4)[12]. Impulsní odezva (také nazývaná jádro filtru či kernel) je výstupní signál systému, na který jsme přivedli jednotkový signál (viz obrázek 2.5). Výstup systému y[n] v čase n pro libovolný diskrétní signál je dán pomocí sumy:

$$y[n] = \sum_{k=-\infty}^{+\infty} x[k]h[n-k]$$

Rastrový obraz lze chápat jako dvourozměrný diskrétní signál. Konvoluce se proto často používá v souvislosti se zpracováním obrazu. Pomocí konvolučních jader lze definovat obrazové filtry provádějící například:

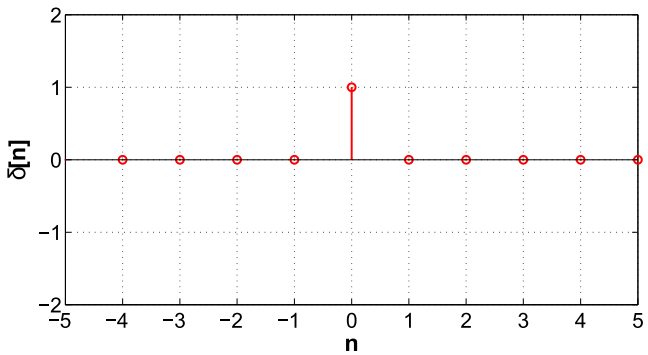
- vyhlazování (filtr dolní propust)
- doostřování (filtr horní propust)
- detekce hran (gradientní operátory)

Takové obrazové filtry lze použít pro redukci technických problémů popsaných v kapitole 2.2.1 nebo extrahování příznaků [13]. Operaci konvoluce lze pro dvojrozměrný obraz vyjádřit následujícím vztahem, v němž symbol g(x,y) značí výstupní obraz, f(x,y) vstupní obraz a h(x,y) konvoluční jádro o rozměrech R x S.

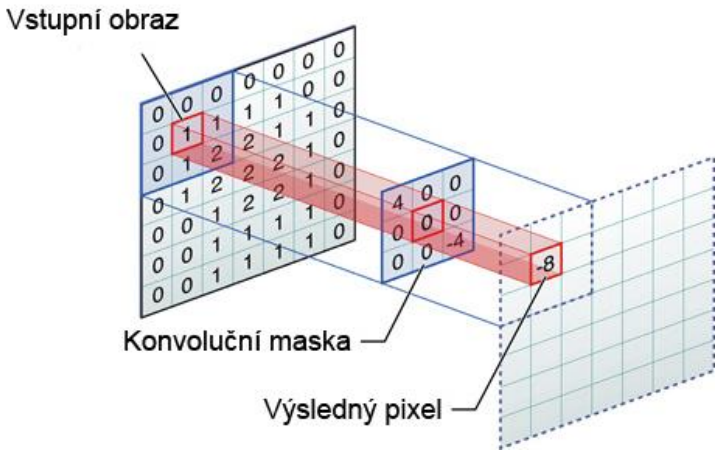
$$g(x,y) = f(x,y) * h(x,y) = \sum_{i=-S/2}^{S/2} \sum_{j=-R/2}^{R/2} f(x-i, y-j)\, h(i,j)$$

Graficky lze konvoluci dvou signálů (obrazu a konvolučního jádra) popsat jako postupné posouvání převrácené masky (konvolučního jádra) po obrazu pro stanovení odezvy. Výstupní hodnota signálu (obrazu) se pro každý posun masky vypočítá jako součet pixelů vážených příslušnými koeficienty masky (viz obrázek 2.6).

.



Obrázek 2.5: Grafické znázornění jednotkového impulsu



Obrázek 2.6: Grafické znázornění 2D diskrétní konvoluce [14]

2.2.4 Klasifikace

Klasifikaci lze definovat jako zobrazení z množiny objektů do množiny předem definovaných tříd. Klasifikátorem je potom algoritmus, který provádí klasifikaci. Tyto algoritmy mohou být jednoduché - třídit na základě jednoduchých třídících pravidel (jednoduchý klasifikátor navržený odborníkem v dané oblasti) až po složité funkce, které využívají mechanismů strojového učení – např. neuronové sítě či SVM, které je potřeba nejdříve natrénovat na trénovacích datech. Výkonnost klasifikátorů se zpravidla posuzuje dle klasifikační úspěšnosti. Použití klasifikátoru v úlohách rozpoznávání obrazu je ilustrováno na obrázku 2.7.

2.2.5 Regrese

Metoda strojového učení, která slouží k aproximaci reálné funkce. Můžeme říct, že se jedná o formu data-miningu (dolování z dat), která modeluje a analyzuje korelaci mezi několika proměnnými [15]. Principem je minimalizování chyby mezi odhadem modelu a správnou hodnotou. Jako příklad běžně používaného typu regresní metody lze uvést lineární regresy, která předpokládá, že výstupní proměnnou Y lze přibližně modelovat lineární kombinací vstupů X tj.

$$Y \approx \beta_0 + \beta_1 X$$

2.2.6 Tradiční způsob

Tradiční způsob rozpoznávání obrazu založený na extrakci příznaků lze rozdělit do dvou kroků (viz obrázek 2.7). Nejdříve je potřeba z předzpracovaného obrazu (odstranění šumu, normalizace atd.) získat charakteristické znaky ve formě příznaků. Tyto příznaky by měly co nejpřesněji popisovat obraz co nejmenším množstvím dat. V obraze se typicky hledá několik typů příznaku, které tvoří vektor příznaků. Druhým krokem je klasifikace/regrese, která vektoru příznaků přiřadí jednu z výstupních tříd, respektive odpovídající výstupní spojitou hodnotu v případě regrese [16]. Na obrázku 2.8 je zobrazeno podrobnější schéma tradičního způsobu klasifikace, které již představuje příklad v praxi používaného přístupu.

Obrázek 2.7: Schéma tradičního způsobu klasifikace obrazu [16]

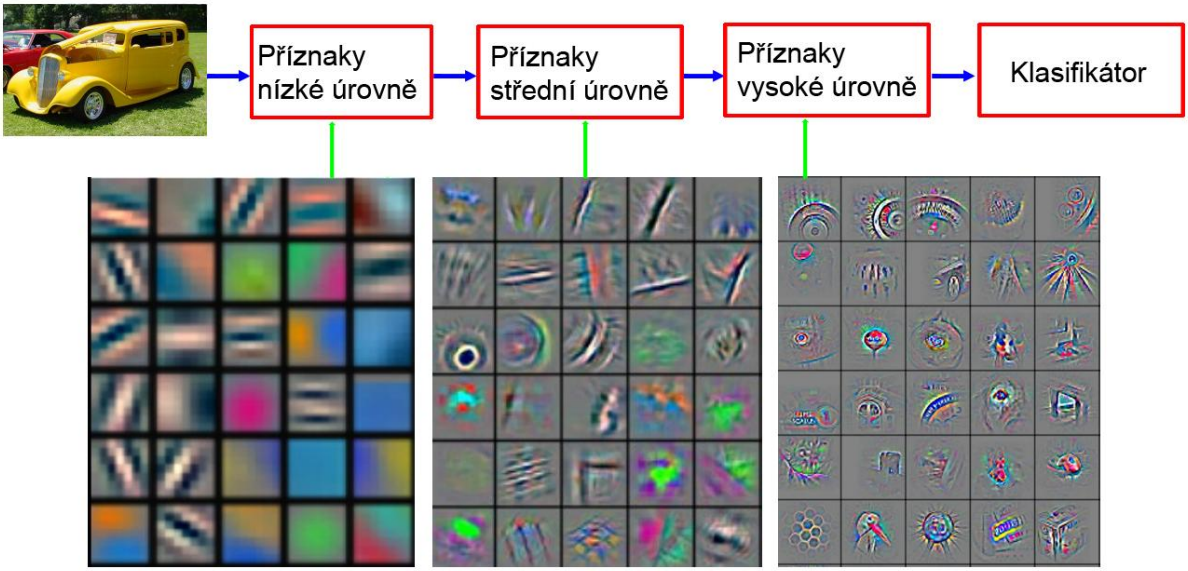
Obrázek 2.8: Podrobné schéma tradičního způsobu klasifikace obrazu [16]

2.3 Deep learning

Deep learning je množina algoritmů strojového učení, které se snaží modelovat vysoko úrovňové abstrakce dat. K tomu používají modely architektur složené z několika nelineárních transformací. Obraz může být reprezentován mnoha způsoby (matice s intenzitami jednotlivých pixelů, vektor příznaků atd.). Pro různé klasifikační/regresní úlohy se hodí různé reprezentace (abstrakce) obrazů. Deep learning je určen k zjištění vhodnosti reprezentace dat v jednotlivých úlohách a k vytvoření modelů, které jsou schopné učit se tyto reprezentace [17].

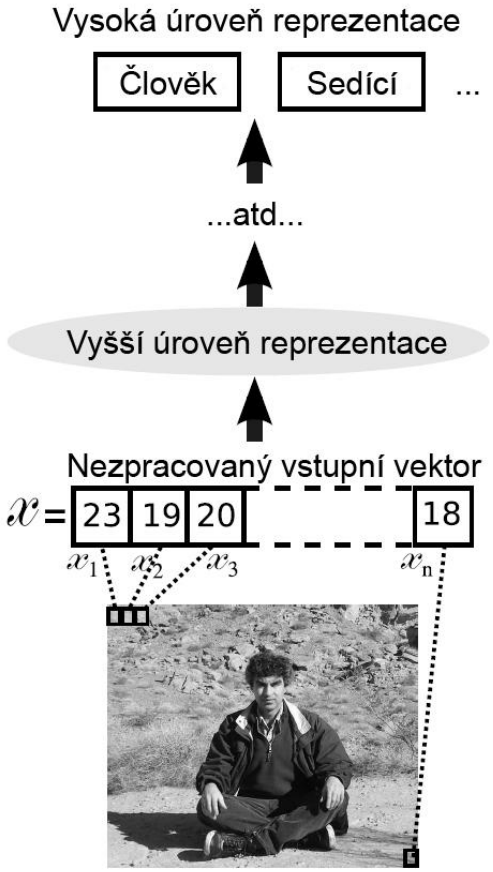
Když má člověk řešit úlohu z oboru umělé inteligence, tak často využívá intuici k rozložení hlavního problému do menších podproblémů a několika úrovní reprezentace. Jako příklad lze uvést moderní verzi tradičního způsobu klasifikace obrazu se schématem na obr. 2.8. Na schématu jsou vidět moduly, které postupně transformují vstupní obraz na reprezentaci vhodnou pro klasifikaci [18]. Hlavním problémem takového přístupu je potřeba vyvinout funkci pro získání vhodných příznaků, které budou obraz transformovat do vyšších úrovní abstrakce. V současnosti neexistuje obecná metoda pro vytvoření funkce, která transformuje obraz na vhodnou abstrakci [16].

Cílem deep learningu je automaticky objevit funkce určené pro tvorbu takových abstrakcí. A to od nízko úrovňových příznaků až po vysoko úrovňové vzory (viz obrázek 2.9). Ideální představou výzkumníků jsou učící se algoritmy, které slouží k takovému účelu s minimální snahou člověka (např. manuálně určit všechny potřebné abstrakce) a bez nutnosti algoritmu poskytnout velké množství relevantních dat s manuálně vytvořenými popisky.



Obrázek 2.9: Schéma zobrazující několik úrovní příznaků [16]

Jako příklad lze uvést obrázek 2.10, na kterém je sedící muž. Jednou z vysoko úrovňových abstrakcí může být označení *Člověk* (toto označení musí být uváděno vždy v souvislosti s řešenou úlohou, jelikož existuje velké množství obrazů, které by tomuto označení vyhovovalo). Tato abstrakce je postupně vytvářena pomocí nižších úrovní abstrakce.



Obrázek 2.10: Hierarchie příznaků [12]

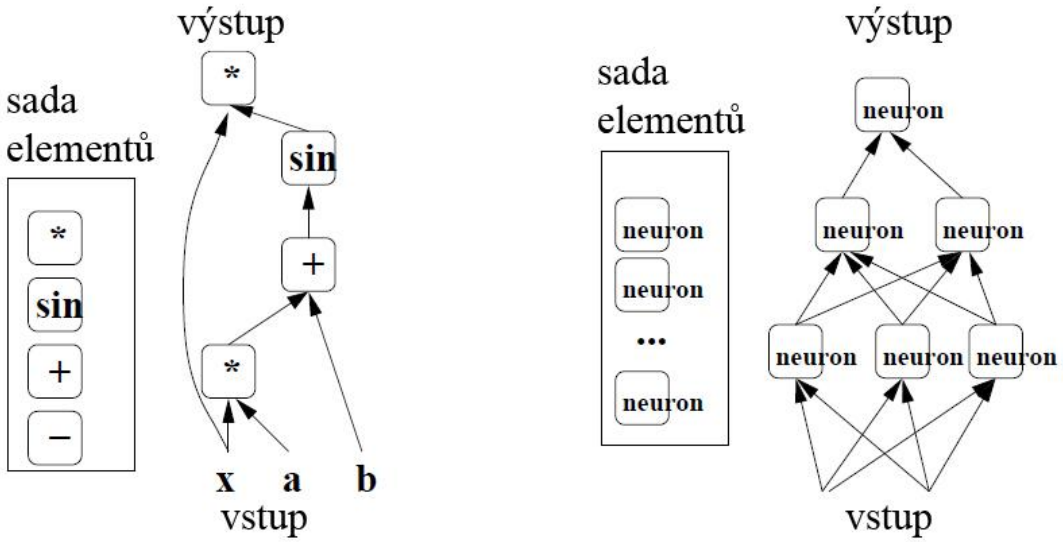
2.3.1 Učení deep architektur

Deep learning metody se zaměřují na učení hierarchií příznaků. Jak bylo uvedeno dříve, vysoko úrovňové abstrakce jsou skládány pomocí nízko úrovňových abstrakcí. Automatické učení příznaků na několika úrovních abstrakce dovoluje systému "učit se" komplexní funkce pro mapování vstupních dat na výstup bez potřeby příznaků vytvořených člověkem (handcraft příznaky). To je důležitá vlastnost především pro vysoko úrovňové abstrakce, u kterých člověk přesně neví, jak je popsat. Schopnost automatického učení kvalitních příznaků se stává čím dál tím důležitější s ohledem na vzrůstající množství dat a potřebu zpracovat je různými způsoby.

2.3.2 Hloubka architektury

Hloubka architektury označuje počet úrovní kompozice nelineárních operací ve funkci, kterou se učí [18]. Architekturu můžeme označit za hlubokou, pokud obsahuje více než dvě takové úrovně. Hloubka tradičních neuronových sítí odpovídá počtu jejich vrstev.

Pojem hloubky můžeme ukázat na flow grafu funkce $f(x) = xsin(ax + b)$, který je na obrázku 2.11. Ten má tři vstupy *a*, *b*, *x*. Dále pak uzly pro výpočet jednotlivých operací, ze kterých se funkce skládá (součin, součet, sinus). Vstupy grafu jsou postupně transformovány uzly až na výslednou reprezentaci odpovídající výstupu funkce.



Obrázek 2.11: Flow graf funkce $f(x) = xsin(ax + b)$ [18]

Pokud se zvolí nedostatečná hloubka architektury, může dojít k velkému nárůstu potřebných uzlů. Hloubka 2 je dostatečná pro mnoho úloh (logická hradla atd.). Pro složitější funkce však může dojít k exponenciálnímu nárůstu potřebných uzlů při odstranění jedné z vrstev [16].

V následujících kapitolách budou vysvětleny pojmy neuronová síť a konvoluční neuronová síť. Neuronová síť je zde vysvětlena proto, že z ní konvoluční neuronová síť do velké míry vychází.

2.4 Neuronové sítě

Neuronové sítě jsou modelem biologických struktur neuronů v živých organismech. Jsou často užívané v umělé inteligenci díky své schopnosti "učit se". Každá neuronová síť je složena z neuronů, které jsou vzájemně propojeny tak, že výstup jednoho neuronu je vstupem do dalších neuronů [19].

2.4.1 Model umělého neuronu

Neuron je základní stavební jednotkou neuronové sítě a lze jej matematicky popsat pomocí funkce:

$$y = f(w_0 + w_1 x_1 + w_2 x_2 + \ ... + \ w_n x_n)$$

kde

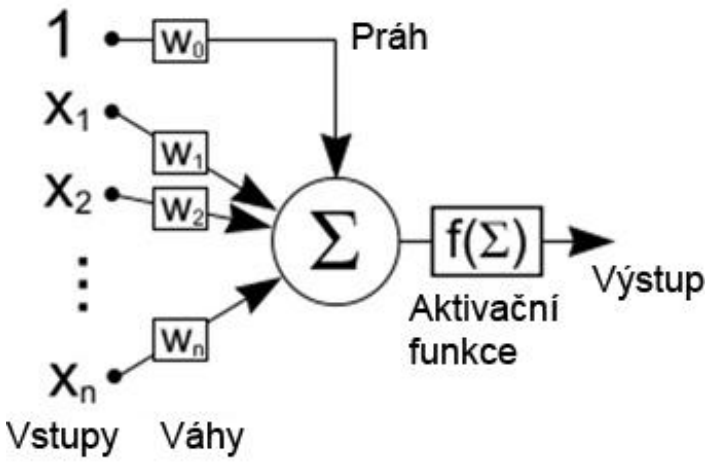
- w_0 je práh
- $w_1,..,w_n$ - váhy vstupů
- $x_1,..,x_n$ - vstupy neuronu
- f(x) - aktivační funkce, která má obvykle tvar funkce *sigmoid* nebo *tanh*.

Neuron se skládá z několika vstupů a jednoho výstupu. Každý vstup je modifikován váhou, která násobí daný vstup. Následně jsou všechny vážené vstupy sečteny a v závislosti na hodnotě prahu a typu aktivační funkce je určen výstup neuronu.

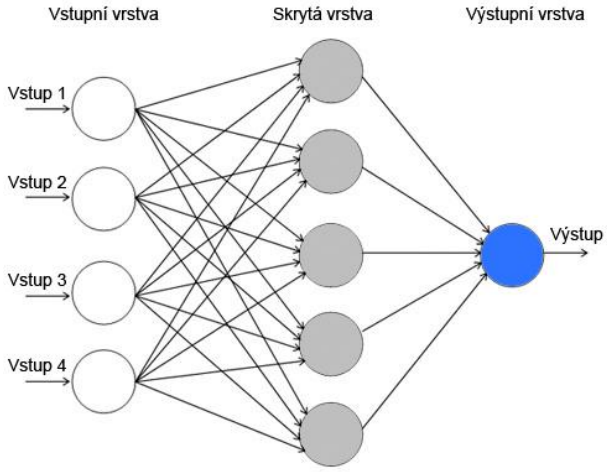
Činnost neuronu, který má N vstupů, lze chápat tak, že rozděluje N-dimenzionální prostor na dva podprostory. Výstup neuronu pak určuje, kterému z těchto dvou podprostorů náleží vstup, který si lze představit jako bod v tomto N-dimenzionálním prostoru.

2.4.2 Architektura neuronové sítě

Počet neuronů a jejich vzájemné propojení v síti určuje architekturu neuronové sítě. Jedním z nejpoužívanějších zapojení neuronů je tzv. *feed-forward* zapojení – několik vrstev neuronů je vzájemně propojených každý s každým [19]. Z hlediska využití rozlišujeme v síti vstupní, vnitřní (skryté) a výstupní vrstvy neuronů (viz obrázek 2.13). Vstupní vrstva pouze přeposílá signál na všechny neurony, neprovádí se žádný výpočet. V ostatních vrstvách pak neurony pracují podobně, jak bylo popsáno v kapitole 2.4.1.



Obrázek 2.12: Model neuronu [20]



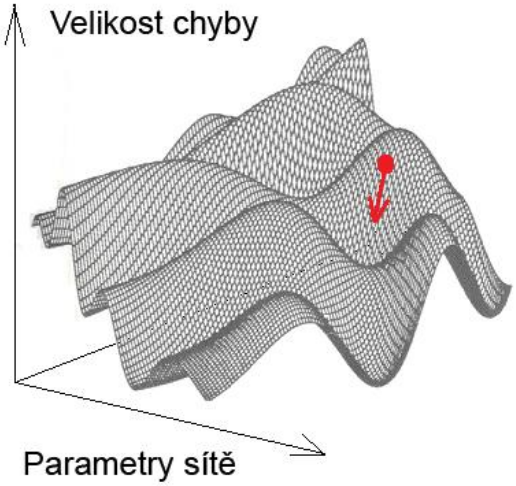
Obrázek 2.13: Dopředné (*feed-forward*) zapojení neuronů

Neuronová síť kombinuje funkce jednotlivých neuronů, které obsahuje. Celá síť je pak schopna rozpoznávat nejenom dvě třídy, ale libovolný počet tříd a je schopna řešit složitější problémy, ve kterých nestačí prostor rozdělit na dva podprostory.

2.4.3 Učení neuronové sítě

Nejčastěji používaným algoritmem k učení neuronových sítí je algoritmus *Backpropagation*. Jelikož tématem práce není implementace neuronové sítě, tak zde bude algoritmus *Backpropagation* stručně vysvětlen bez přesného matematického vyjádření.

Samotný algoritmus obsahuje tři etapy: dopředné (*feed-forward*) šíření vstupního signálu trénovacího vzoru, zpětné šíření chyby a aktualizace váhových hodnot na spojeních [19]. Na začátku učení se zvolí náhodné nastavení parametrů. Takovou náhodnou neuronovou síť lze zanést jako bod do grafu na obrázku 2.14. Algoritmem lze zjistit směr, ve kterém chyba z tohoto počátečního bodu klesá nejrychleji. Směr reprezentuje vektor, který nazýváme gradient. Postupně ve směru gradientu upravujeme váhy neuronové sítě (posouváme bod na grafu), dokud se nedostaneme do místa s nejnižší chybou (lokální minimum) [20].



Obrázek. 2.14: Ukázka závislost velikosti chyby na nastavení parametrů neuronové sítě [20]

2.5 Konvoluční neuronové sítě

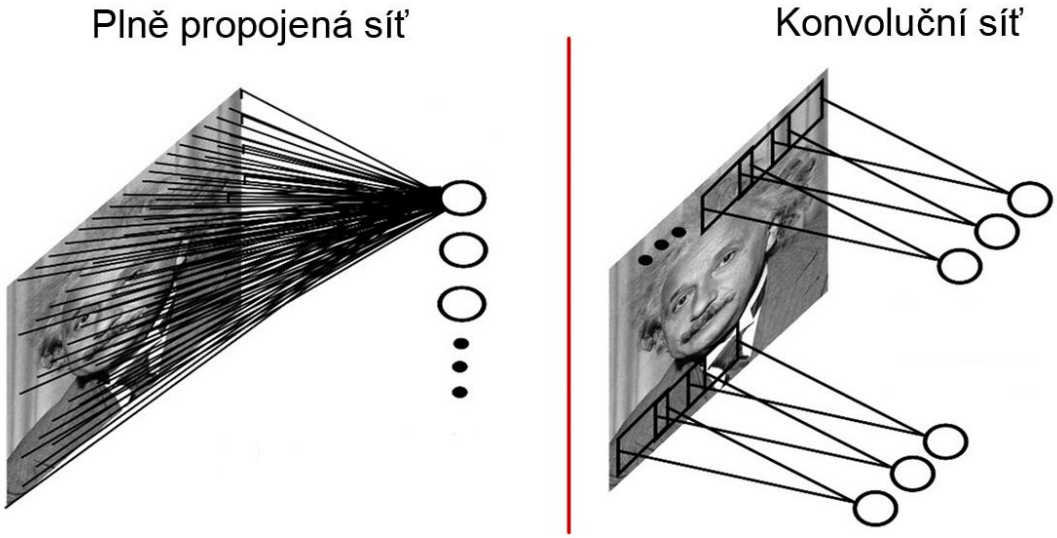
Konvoluční neuronové sítě jsou speciálním druhem neuronových sítí. Byly primárně navrženy pro rozpoznávání dvojrozměrných obrazových vzorů přímo z pixelů minimálně předzpracovaných obrazů. Vznik konvolučních neuronových sítí lze datovat do roku 1980 kdy Kunihiko Fukushima vytvořil síť *Neocognitron* [2]. Jejich vývoj probíhal pomalu a narážel na řadu problémů [21]. Zlom nastal v roce 2012, kdy Alex Krizhevsky s kolegy vytvořil konvoluční nerunovou síť, která dominovala v soutěži ILSVRC na datasetu *ImageNet* [22]. V posledních dvou letech pak probíhá rychlý vývoj v této oblasti [23]. V následujících kapitolách bude vysvětlena jejich odlišnost od standardních neuronových sítí a dále budou představeny techniky použité v síti *AlexNet* [22].

2.5.1 Architektura konvolučních neuronových sítí

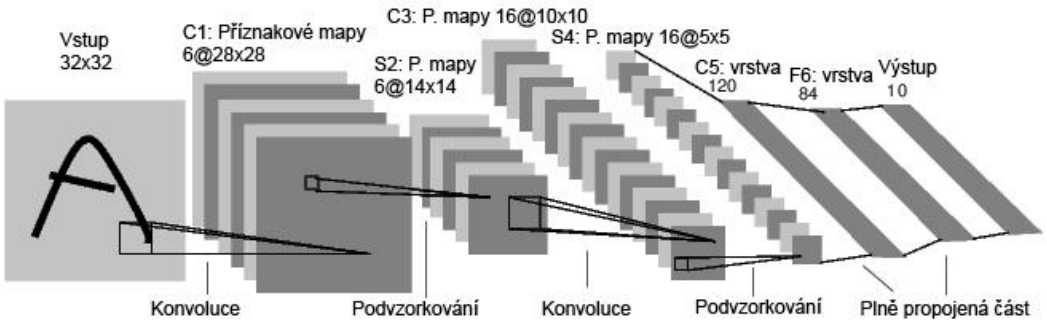
Konvoluční neuronové sítě lze popsat jako dopředné neuronové sítě, které jsou obvykle hluboké. Na rozdíl od standardních dopředných neuronových sítí obsahují speciální vrstvy neuronů, u nichž se vynucuje struktura vah [23].

2.5.2 Konvoluční vrstva

Myšlenkou je získat lokální charakteristiky obrazu pomocí konvoluce, která funguje jako filtr obrazu [24]. Konvoluce je v síti implementována pomocí sdílení vah mezi neurony a specifickým způsobem jejich propojení. Neurony nejsou plně propojené (neuron není propojen se všemi neurony sousedních vrstev), ale jsou spojeny pouze s malým počtem neuronů předchozí vrstvy, které reprezentují lokální oblast obrazu či jeho jiné reprezentace (viz obrázek 2.15). Výsledek konvoluce je reprezentován ve formě příznakových map (tzv. kanály). Při rozpoznávání obrazů je vhodné rozpoznávat více příznaků. Proto těchto příznakových map bývá většinou použito více.



Obrázek 2.15: Způsob zapojení neuronů v konvolučních vrstvách [11]



Obrázek 2.16: Ukázka architektury sítě *LeNet* [20]

2.5.3 Vrstva podvzorkování

Tato vrstva představuje formu nelineárního podvzorkování signálu. Eugenio Culurciello [21] uvádí, že je tato nelinearita velmi důležitá. Pokud by byly v síti pouze lineární operace, tak by šlo několik vrstev spojit do vrstvy jedné. Vrstva podvzorkování počítá lokální statistické údaje pomocí funkce *max*/*mean*. Tím zvětšuje robustnost sítě vůči malým translacím obrazu [23]. Dalším přínosem podvzorkování prostorové informace je možnost hledat větší příznaky v následujících konvolučních vrstvách při použití podobných rozměrů konvolučních jader [21].

2.5.4 Princip činnosti

Konvoluční neuronové sítě hledají kvalitní příznaky pro rozpoznávání vstupního obrazu. Toho dosahují postupnou změnou reprezentace obrazu. Vstupem sítě je typicky obraz, který je v každé konvoluční vrstvě transformován na reprezentaci, která vhodněji popisuje informace o obsahu obrazu (viz obrázek 2.16). To vede k redukci prostorové informace. Na konci procesu extrakce příznaků je vektor čísel, který se následně předá do klasifikátoru/regresoru (typicky dopředné neuronové sítě).

2.5.5 Techniky použité v síti AlexNet

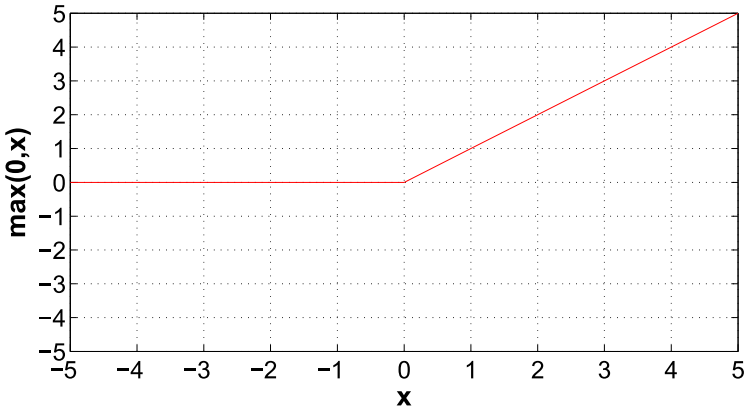
Autoři v práci [22] navrhli konvoluční neuronovou síť, která dosáhla velmi kvalitních výsledků na datasetu *ImageNet*. Použili přitom techniky, které se později staly součástí mnoha dalších sítí. Mezi nejdůležitější patří tyto techniky:

- **ReLU (Rectified Linear Unit)**

  Model neuronu, jehož aktivační funkce má tvar:

  $$f(x) = \max(0, x)$$

  Představuje zdroj nelinearity v konvolučních sítích. Pro kladné hodnoty součtu váhovaných vstupů neuoronu se jedná o lineární průběh. Záporné hodnoty se mapují na nulovou hodnotu (viz obrázek 2.17). Konvoluční neuronové sítě používající ReLU jednotky se trénují několikrát rychleji než jejich ekvivalenty s *tanh* jednotkami. ReLU jednotky tak dovolují pracovat s většími sítěmi při stejném výpočetním čase.



Obrázek 2.17: Průběh funkce $f(x) = \max(0, x)$

- **Local Response Normalization**
  ReLU jednotky mají zajímavou vlastnost a to, že nepotřebují normalizaci vstupu k zabránění saturace. Učení neuronu probíhá v případě, pokud je mu přiveden kladný vstup od alespoň jednoho trénovacího vzoru. V síti *AlexNet* je i přesto použita normalizace, která vede ke snížení chybovosti sítě o 1,4% [19]. Normalizace slouží například k invarianci sítě vůči změnám osvětlení a může být prováděna prostorově nebo přes všechny kanály [23].

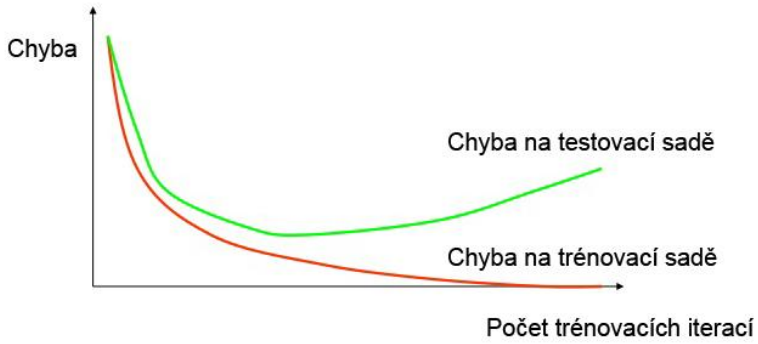
- **Data augmentation**

  Jednou z možností, jak zamezit přetrénování (viz kapitola 2.6) na velikých sítích, je umělé navýšení velikosti datasetu. Toho bylo v síti *AlexNet* dosáhnuto výpočetně nenáročnými transformacemi obrazů. Ze vstupních obrazů o velikosti 256 x 256 byly náhodně extrahovány oblasti o velikosti 224 x 224. Tyto oblasti a jejich horizontálně převrácené ekvivalenty pak vytvořily nový trénovací dataset.

- **Dropout**
  Kombinace výsledků několika modelů je efektivní způsob jak zmenšit chybu sítě. Problém je výpočetní náročnost tohoto způsobu, jelikož trénování jedné sítě trvá několik dní. Jednou z možností, jak tento problém obejít, je použít techniku *Dropout*. *Dropout* spočívá v nastavení nulového výstupu skrytých neuronů s pravděpodobností 0,5. Neurony, jejichž výstup je nula, se nepodílejí na učení sítě. To ve výsledku znamená, že neuronová síť pro každý vstup vytváří odlišnou architekturu sítě a znalosti tak musí distribuovat po celé síti.

2.6 Přetrénování a přeučení metod strojového učení

Na závěr kapitoly věnované popisu různých způsobů, jak přistupovat k odhadu nadmořské výšky kamery z obrazu pomocí metod strojového učení je nutné uvést, že dokonalé trénování na trénovacích datech nemusí být pro výsledné vlastnosti systému dobré, protože systém může přijít o schopnost generalizace (zobecňování). To znamená, že systém není schopen dosáhnout zobecnění vztahu mezi vstupy a výstupy. Generalizace umožňuje řešit jiné, pro systém nové a neznámé případy na základě podobnosti, čehož lze využít k predikci nebo klasifikaci dat [25].



Obrázek 2.18: Velikost chyby na trénovací a testovací sadě během trénování

Z tohoto důvodu se množina dat, která reprezentuje řešený problém, rozděluje na trénovací a testovací sadu. Pomocí trénovací sady se adaptují parametry zvoleného klasifikátoru/regresoru a testovací sada slouží k ověření kvality trénování, a to včetně její generalizační schopnosti.

V ideálním případě konverguje celková chyba klasifikátoru/regresoru na trénovací sadě k nule (systém je schopný dokonale predikovat výstup pro data z trénovací sady). Chyba klasifikátoru/regresoru ovšem na testovací sadě nejdříve klesá a následně roste (viz obrázek 2.18). Stav minima chyby na testovací množině je vhodným okamžikem pro ukončení procesu učení, protože od této chvíle dál klasifikátor/regresor ztrácí schopnost generalizace a je příliš orientován na trénovací sadu [26]. Tento jev se nazývá přetrénování (*overtraining*).

Příliš velká fixace na trénovací sadu způsobuje také tzv. přeučení (*overfitting*). Dochází k němu použitím příliš složité architektury zvoleného klasifikátoru/regresoru nebo příliš bohatého způsobu popisu dat, kdy během trénování máme k dispozici malé množství trénovacích dat. Přeučení lze řešit zvětšením trénovací sady, snížením složitosti architektury systému nebo použitím různých regularizačních technik, které do procesu učení zavádějí penalizační proměnnou [25].

Cílem učení tedy není maximalizace výkonu systému na trénovacích datech, ale rozumný kompromis mezi výkonem na trénovací sadě a schopností zobecnit znalosti na nových datech.

3 Předchozí práce

V současnosti bohužel neexistují dostupné materiály o výzkumech týkajících se odhadu nadmořské výšky z obrazu z pohledu člověka. V této kapitole jsou proto uvedeny příklady projektů, které se snaží získávat podobné informace z obrazu. Na těchto projektech můžeme vidět použití tradičního způsobu rozpoznávání obrazu i použití konvolučních neuronových sítí.

3.1 Yahoo Weather

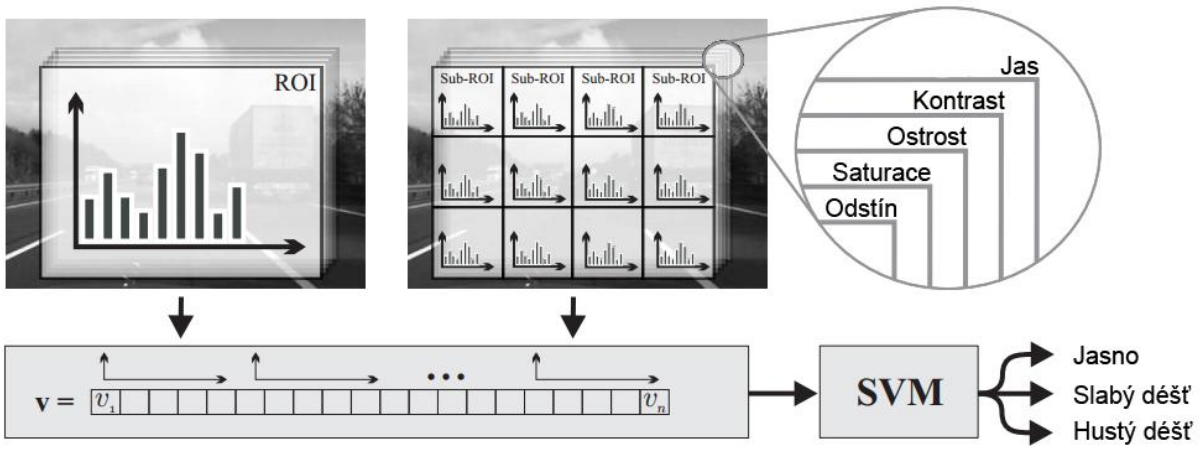
V *Yahoo Labs* vytvořili systém [27], který dokáže odhadnout povětrnostní podmínky a denní dobu z obrazu. Projekt začal vznikat v roce 2013. Na začátku byl vytvořen ekosystém, jehož cílem bylo vytvořit kolekci čítající 1 milion kvalitních fotek s motivem počasí. Podmínkou bylo, aby kolekce obsahovala fotografie reprezentující typické příklady počasí z celého světa. Fotografie byly získávány od uživatelů *Flickru*[1], kteří posílali zajímavé fotografie s motivem počasí do skupiny s názvem *Flickr Project Weather Group*. Editoři skupiny kontrolovali vhodnost fotek na základě jejich velikosti, obsahu a kvality. Následně byly z kolekce odstraněny fotografie obsahující nevhodné objekty (obličej atd.) nebo fotografie, kterým chyběly GPS souřadnice. Z přispěvatelů do skupiny *Flickr Project Weather Group* byla vytvořena komunita, na kterou byly aplikovány metody zkoumající společná témata lidí používajících sociální sítě. Pomocí těchto metod byly vytvořeny klíčové kategorie systému (zataženo, déšť, bouřka atd.). Na konci pak byla použita konvoluční neuronová síť pro vygenerování příznaků, které byly použity pro trénování klasifikátoru. Příklady výstupu klasifikátoru jsou zobrazeny na obrázku 3.1.

3.2 Klasifikace počasí pro asistenční systémy ve vozidlech

Cílem výzkumu bylo vytvořit systém, který by z obrazu dokázal určit aktuální počasí. Výstup systému je jedna ze tří kategorií: jasno, slabý déšť, hustý déšť. Výzkum byl veden tradičním způsobem klasifikace obrazu. Hlavní těžiště práce tedy spočívalo v nalezení vhodných příznaků pro tuto úlohu. V práci je představena metoda, která kromě samotného obrazu používá 12 oblastí zájmu (ROI), které rovnoměrně pokrývají celý obraz. Všechny oblasti zájmu se dále dělí na bloky o velikosti 10x10 pixelů.

Obrázek. 3.1: Ukázka výstupu systému *Yahoo Weather* [13]

[1] www.flickr.com/



Obrázek 3.2: Princip činnosti systému pro určení aktuálního počasí [28]

Pro každý blok se počítá těchto 5 příznaků:

- lokální kontrast
- nejnižší jas
- ostrost
- odstín
- saturace

Výsledky z jednotlivých bloků jsou poté zaneseny do histogramu pro každou oblast zájmu. Ve výsledku je tedy vytvořeno 5 histogramů v každé z 13 oblastí (obraz + 12 oblastí zájmu). Metoda pro každý obraz vytvoří vektor příznaků, který obsahuje 650 hodnot (13 oblastí · 5 příznaků · 10 tříd histogramu). Tento vektor je následně předán do SVM klasifikátoru, který vrací jednu z cílových kategorií (viz obrázek 3.2).

Ukázku výstupu systému je možné vidět na obrázku 3.3. Systém dosahuje téměř 11% chybovosti. Autoři článku uvádějí, že je to způsobeno plynulými přechody rozdílů mezi obrazy v jednotlivých kategoriích. Navrhují proto postup, kde jsou obrazy snímány v sekvencích, ze kterých se odstraní extrémy a poté určí výsledná kategorie [28].

Obrázek 3.3: Ukázka výstupu systému pro určení aktuálního stavu počasí [28]

3.3 IM2GPS

V této práci je představen algoritmus, jak provádět geolokalizaci obrazu. Výsledná lokace obrazu je reprezentována jako rozdělení pravděpodobnosti výskytu na zemském povrchu. Pokud je na obrazu známá kulturní památka, pak bude systém pravděpodobně vracet správné GPS souřadnice. Pokud je na obrazu spíše obecná scéna jako poušť, tak systém bude vracet vysoké hodnoty pro suchá, písčitá místa. Pro vytvoření datasetu použili tvůrci systému fotografie z *Flickru*, které kromě GPS souřadnic zároveň obsahovaly v hash tagu zeměpisné názvy (Washington, Asie, Kanada atd.). Z této kolekce byly odstraněny fotografie obsahující zavádějící hash tagy jako narozeniny, koncert, svatba. Tímto byl vytvořen dataset obsahující 6 472 304 obrazů. I přesto dataset pokrýval pouze malou část zeměkoule (průměrně 0,0435 obrazu na km^2). Dále autoři představují tyto příznaky pro klasifikaci:

- Náhled s malým rozlišením
- Histogram barev
- Histogram textur
- Délky a úhly přímek
- GIST deskriptor
- Hledání geometrických útvarů

Proces geolokalizace pak probíhá tak, že se pro vstupní obraz nejdříve vypočítají všechny příznaky a poté se spočítá vzdálenost v příznakovém prostoru ke všem obrazům v datasetu. Autoři používají několik způsobů jak tuto vzdálenost počítat (1-NN, 120-NN, mean-shift clustering). Výsledky systému jsou však pro praxi nepoužitelné. Pouze ve dvou třetinách případů dosahují lepších výsledků než je náhoda [29].

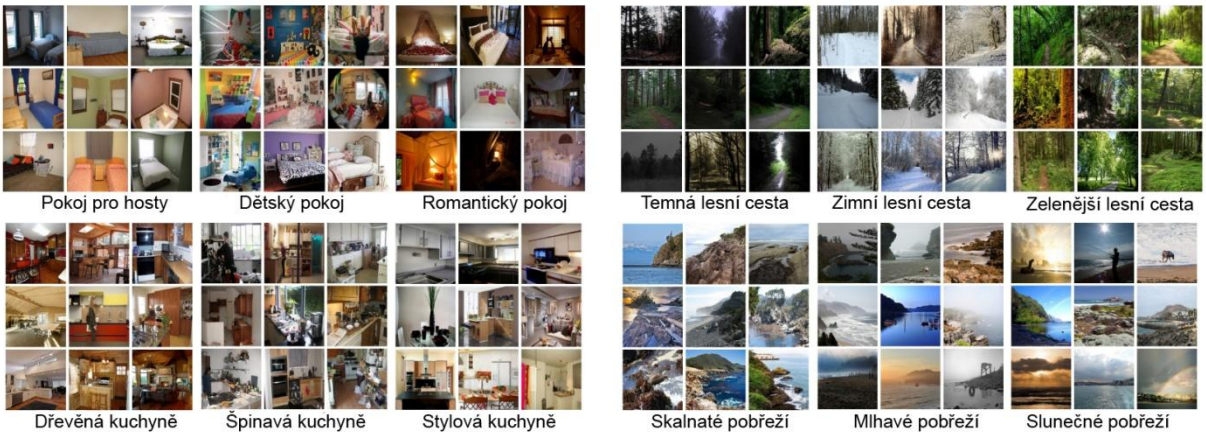
3.4 Places database

Places database [30] je datová sada sloužící k trénování klasifikátorů scén. Hlavní motivací autorů práce *Places database* bylo vytvoření první dostatečně velké datové sady scén, která by se dala použít k trénování konvolučních sítí. Konvoluční sítě natrénované na datových sadách objektů dosahovaly v úloze klasifikace scén podobných výsledků jako klasifikace scén založená na použití hand-crafted příznaků. Důvodem je odlišnost vysoko-úrovňových příznaků pro obě úlohy. Existující datové sady [31] [32] byly pro trénování konvoluční sítě příliš malé a proto bylo nutné vytvořit novou unikátní datovou sadu scén.

Seznam kategorií scén v *Places database* je stejný jako v datové sadě *Sun database*. Tento seznam byl použit k vytváření dotazů pro stahování obrazů. Pro každou kategorii v seznamu bylo vytvořeno velké množství dotazů, které byly generovány tak, že název kategorie byl kombinován s populárními přídavnými jmény (slunečný, pustý, špinavý atd.). Použitím tohoto způsobu bylo staženo velké množství obrazů (viz obrázek 3.4). Kolekce obrazů byla následně manuálně anotována pomocí služby *Amazon Mechanical Turk*[2] ve dvou krocích. Anotace spočívala v rozhodnutí zda obraz opravdu patří do předpokládané kategorie (například: "Jedná se obývací pokoj?"). Oba kroky se lišily v nastavené defaultní odpovědi. Defaultní hodnota odpovědi v prvním kroku byla "ano". V druhém kroku naopak "ne". Po dvou krocích anotace bylo do *Places database* zařazeno 7 076 580 obrazů z 476 kategorií scén. Subset datové sady *Places database* byl použit pro natrénování konvoluční sítě

[2] www.mturk.com/mturk

Places-CNN, která v současnosti dosahuje nejlepších výsledků v úloze klasifikace vnitřních/venkovních scén (viz kapitola 5.4.2).



Obrázek 3.4: Ukázka stažených obrazů, které jsou seřazeny podle použitých dotazů [30]

3.5 Odhad nadmořské výšky bezpilotního letounu pomocí kamery

Práce [33] se zabývá odhadem nadmořské výšky bezpilotního letounu vybaveného kamerou. Kamera je umístěna na spodní části letounu a snímá prostor pod letounem. Motivací autorů byla optimalizace a miniaturizace senzorů umístěných na bezpilotních letounech. K určení nadmořské výšky, která je důležitá např. během přistávání, leteckých manévrech i stabilním letu, je potřeba samostatný senzor. Tento senzor není nutné používat pokud by bylo možné nadmořskou výšku odhadnout z obrazové informace pořízené kamerou, která je nepostradatelnou součástí všech bezpilotních letounů

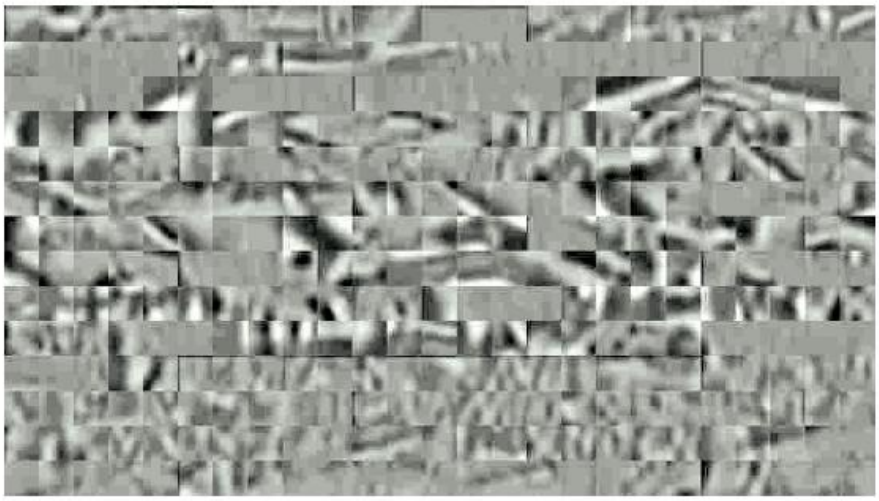
Celý algoritmus je založen na mapování textur z obrazu na nadmořskou výšku. Autoři definují tyto předpoklady:

- Terén pod bezpilotním letadlem je plochý a proto stačí udělat jediný odhad pro celý snímek.
- Bezpilotní letoun neprovádí náhlé změny nadmořské výšky. Změna nadmořské výšky dvou po sobě následujících snímků tedy není skoková.

Snímky pořízené z kamery jsou příliš zašumělé, rozmazané pohybem nebo obsahují příliš velké rozdíly v osvětlení. Konvenční filtry (fourierova transformace, vlnková transformace atd.) navíc nedokážou zachytit variace textury vzhledem k variacím nadmořské výšky.

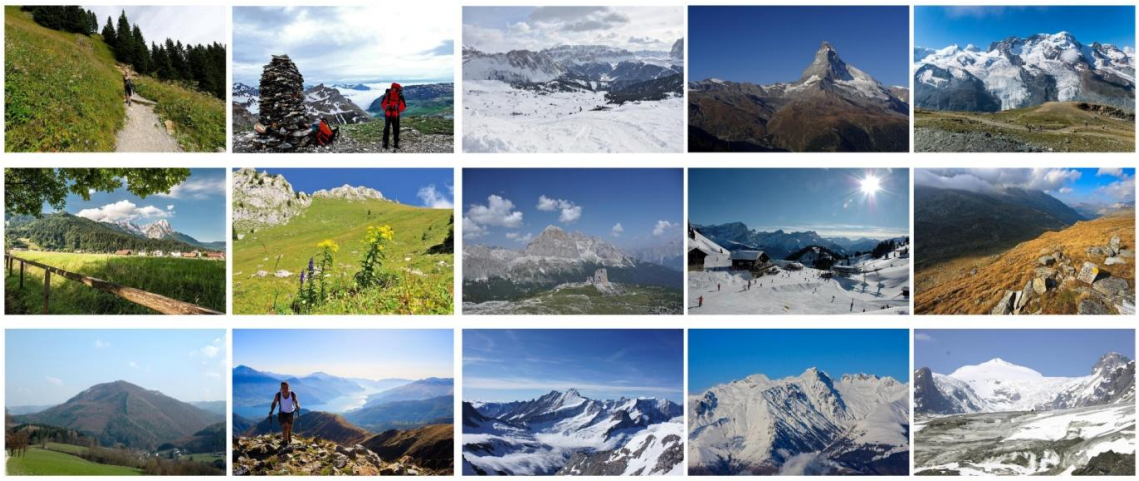
Pro eliminaci těchto problémů je v práci použita metoda řídkého kódování (sparse coding). Každý obraz je možné reprezentovat jako k-dimenzionální vektor. Princip metody spočívá ve vytvoření řídké báze z obrazů různých terénů (viz obrázek 3.5). Libovolný obraz je pak možné modelovat řídkou, lineární kombinací bázových vektorů. Metoda je robustní vůči šumu a do jisté míry i vůči změnám osvětlení. Pro aproximaci vztahu získaných vektorů a nadmořské výšky je použita regrese. Na konci je použit časoprostorový MRF (*Gaussian Markov Random Field*) pro upravení odhadu na základě definovaných předpokladů/omezení (terén je plochý, bezpilotní letoun neprovádí náhle změny nadmořské výšky).

Navržený přístup dosahuje přibližných odhadů v malých nadmořských výškách, kdy bezpilotní letoun letí malou rychlostí. Zcela selhává ve vyšších nadmořských výškách. Důvodem je chybějící informace o textuře v takových fotografiích. Jako částečné řešení autoři navrhují používat kvalitnější kameru, či informace z ostatních senzorů na bezpilotním letounu.



Obrázek 3.5: 350 bázových vektorů o velikosti 16x16, které byly vytvořeny z 50 000 náhodně vybraných leteckých obrazů z internetu. [33]

4 Odhad nadmořské výšky člověkem



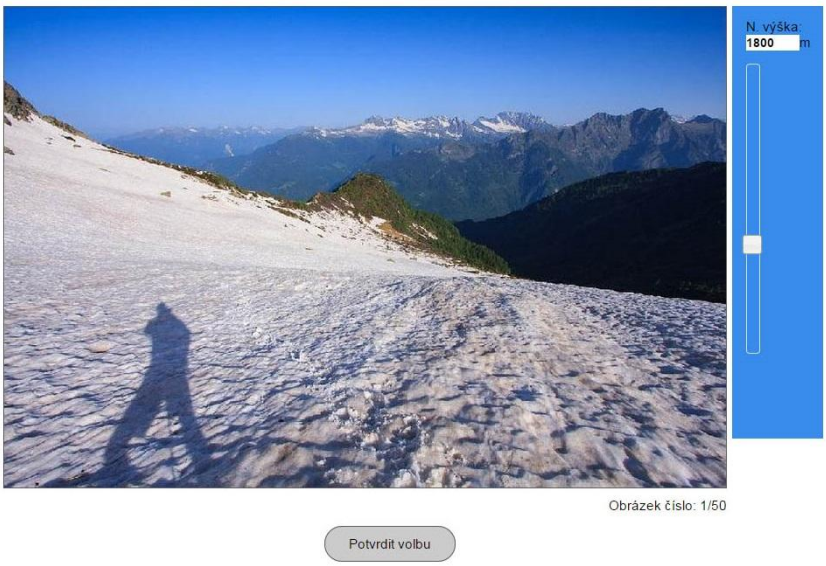
Obrázek 4.1: Ukázka obrazů použitých v uživatelském testu

Tato kapitola prezentuje uživatelský test, jehož cílem bylo kvantifikovat schopnost lidí určovat nadmořskou výšku na vytvořeném datasetu *Alps100k* (viz kapitola 5.1). Kromě toho, že získané výsledky poskytují nové a zajímavé poznatky o schopnostech člověka, slouží také jako reference pro automatické metody, které jsou popsány v následující kapitole.

4.1 Parametry testování

Testování proběhlo pomocí webové aplikace (viz obrázek 4.2). Experimentu se zúčastnilo celkem 100 lidí, kteří odhadovali nadmořskou výšku kamery pro 50 obrazů (viz obrázek 4.1) . Věk účastníků byl v rozmezí od 11 do 61 let. Zde je dobré zmínit, že věk 11 let nemusí být dostatečný pro kvalifikovaný odhad nadmořské výšky z obrazu. Odhady zmíněného účastníka však nepatřily mezi outliers a proto nebyly z testu vyřazeny. Věk ostatních účastníků je v rozmezí 21-61 let. Obrazy venkovních scén byly náhodně vybrány z testovací sady datasetu *Alps100k* tak, aby rovnoměrně pokrývaly rozsah nadmořských výšek od 79m do 4463m. (viz obrázek 6.5).

Před spuštěním testu byl každý uživatel poučen o účelu testu a správném způsobu jak test vyplnit (např. uživatel určuje nadmořskou výšku kamery a ne vzdáleného objektu na fotografii). Samotný test pak probíhal tak, že se uživateli v náhodném pořadí postupně zobrazovaly obrazy z testovací kolekce. Kromě obrazu nebyla k dispozici žádná doplňující informace. Účastník určoval nadmořskou výšku kamery pomocí posuvníku (viz obrázek 4.2). Tento odhad pak uživatel potvrdil tlačítkem po jehož zmáčknutí se zobrazil další obraz. Průměrná doba trvání testu byla 10 minut.



Obrázek 4.2: Uživatelské rozhraní testu

4.2 Vyhodnocení uživatelského testu

Výsledek uživatelského testu jsem nejdříve analyzoval funkcí *ANOVA* (analýza rozptylu) [34], abych ověřil že odhady nadmořské výšky kamery nejsou náhodné a člověk je schopný určovat nadmořskou výšku na základě fotografie. K tomuto účelu jsem vytvořil nulovou hypotézu H_0*: Při odhadu člověka nezáleží na obrazu scény*. Výsledek analýzy rozptylu je ($F(49,4950)=165,09$[3], $p<0,001$), což znamená, že člověk je schopný odhadovat nadmořskou výšku kamery pouze na základě obrazové informace.

V tuto chvíli je již možné zkoumat charakteristiku odhadu člověka. Na obrázku 6.5 je zobrazena střední hodnota a rozptyl odhadu všech účastníků testu pro každý testovací obraz. Nadmořské výšky do 1000 m je člověk schopný odhadnout s malou chybou. S rostoucí nadmořskou výškou však chyba roste. Nejvíce je to vidět u nadmořských výšek nad 3000 m, kde člověk odhady velmi podhodnocuje. Jedním z důvodů může být to, že většina účastníků neměla velké zkušenosti s pobytem v těchto výškách. Průměrná chyba odhadu člověka je 879,94 m.

[3] F je testová statistika. Vyjadřuje poměr mezi rozptylem "mezi" skupinami a rozptylem "uvnitř" skupin. Hodnota testové statistiky výrazně vyšší než 1 umožňuje hypotézu zamítnout.

5 Návrh řešení a implementace

Tato kapitola popisuje návrh řešení a implementaci systému pro odhad nadmořské výšky kamery z obrazu. Nejdříve je popsán způsob vytvoření unikátní datové sady a její charakteristiky. Dále jsou popsány experimenty s klasifikačními a regresními modely u nichž testuji použití různých architektur a vstupních dat. Na konci kapitoly je popsán návrh výsledného systému pro odhad nadmořské výšky z obrazu.

5.1 Dataset

Metody strojového učení s učitelem potřebují k trénování datovou sadu, která pro všechny vstupní data obsahuje předpovídanou hodnotu (výška, barva, typ objektu atd.). V současnosti neexistuje obsáhlá datová sada, která by pro každý obraz obsahovala údaje o nadmořské výšce kamery. Existují však datové sady pro podobné úlohy popsané v kapitole 3, které jsem prozkoumal:

- **IM2GPS**
  V této práci [29] je vytvořen dataset, který obsahuje GPS souřadnice ze kterých je možné určit nadmořskou výšku pomocí výškových map či aplikací typu *Google Earth*[4]. Problém tohoto datasetu je nedostatek fotografií zachycujících venkovní scény v jednotlivých nadmořských výškách a příliš velký rozsah zeměpisných lokalit.

- **Places205**
  Z tohoto datasetu [30] by bylo možné vytvořit vhodný subset pro řešenou úlohu, protože obsahuje velké množství obrazů venkovních scén a zároveň splňuje požadavky na hustotu a rozmanitost. Veřejně dostupná verze datasetu *Places205* však neobsahuje dodatečné informace k fotografiím (GPS souřadnice či nadmořskou výšku). Kontaktoval jsem proto autory tohoto datasetu s žádostí o poskytnutí chybějících informací o obrazech, které bych mohl využít pro tvorbu správných popisků fotografií. Autoři bohužel data k fotografiím neměli a jediný dataset s informacemi o GPS souřadnicích, který vytvořili je z městského prostředí. Pro tento výzkum jsem proto vytvořil nový dataset obsahující nadmořskou výšku ke každé fotografii (Dále jen *Alps100k*).

5.1.1 Vytvoření datasetu

Přírodní ráz krajiny se s nadmořskou výškou mění. Nadmořská výška například ovlivňuje teplotu, intenzitu slunečního svitu, či množství srážek. Čím vyšší nadmořská výška, tím nepříznivější bývají klimatické podmínky. Zimy ve vyšších polohách bývají delší, sníh taje později než v nížinách a nástup jara bývá opožděn [35].

Další element, který přispívá k přírodní variabilitě je zeměpisná poloha daného místa. Charakter dvou míst se stejnou nadmořskou výškou ale v různých zeměpisných lokalitách se bude velmi pravděpodobně lišit. Pro tvorbu datasetu jsem tedy vybral pouze oblast pohoří Alp. Tato oblast se dá považovat za dostatečně malou a můžeme říct, že krajinný charakter v jednotlivých nadmořských

[4] www.google.cz/intl/cs/earth/

Obrázek 5.1: Ukázka obrazů, které se nehodí do vytvořeného datasetu

hladinách bude velmi podobný. Zároveň je častým cílem turistů a existuje tak velké množství fotek, které tuto oblast zachycují.

K dispozici jsem dostal kolekci čítající téměř 1,2M fotografií. Seznam obsahující názvy všech vrcholů Alp byl vytvořen pomocí metadat ve službě *OpenStreetMap* [36]. Tento seznam byl následně použit pro dotazování obrazů na *Flickru* pomocí hash tagů. Stáhnuty byly veřejné obrazy, které obsahovaly GPS souřadnice. Tyto obrazy nesměly obsahovat hash tagy, které nesouvisejí s venkovními scénami (svatba, oslava atd.). Po stažení byly fotografie ještě automaticky upraveny tak, aby neobsahovaly jednobarevné okraje.

5.1.2 Filtrace Datasetu

Postup popsaný v kapitole 5.1.1 bohužel vedl také ke stažení velkého počtu fotografií nepoužitelných pro úlohu odhadu nadmořských výšek. Příklady takových fotografií jsou zobrazeny na obrázku 5.1.
Z této kolekce fotografií bylo nutné odstranit podobné obrazy. Jedním z možných řešení je použít klasifikátor scén, který obrazy z kolekce klasifikuje do různých kategorií, z nichž budou následně vybrány ty, které se hodí pro řešenou úlohu. K této práci jsem použil model konvoluční neuronové sítě *Places-CNN* [30]. Ta dokáže rozpoznávat až 205 druhů scén a v současnosti se považuje za nejkvalitnější klasifikátor scén (viz kapitola 5.4.2).

Další otázkou bylo vybrat takový seznam kategorií, který by vedl k vytvoření kvalitního datasetu. Systém pro odhad nadmořských výšek kamery z obrazu je primárně určen pro venkovní fotografie pořízené turisty během jejich cest po Alpách. To znamená, že fotografie neobsahují pouze přírodu, ale mohou se v nich nacházet např. horské chaty, lidé a podobné objekty. Tyto fotografie by však měly obsahovat dostatečnou informaci o scéně, která nebude příliš překryta rušivými objekty jako obličej, zvíře atd. (viz obrázek 5.2). Dále je pak nutné, aby fotografie obsahovaly charakteristické krajinné prvky typické pro oblast Alp.

Obrázek 5.2: Ukázka obrazů s nedostatečnou informací o scéně

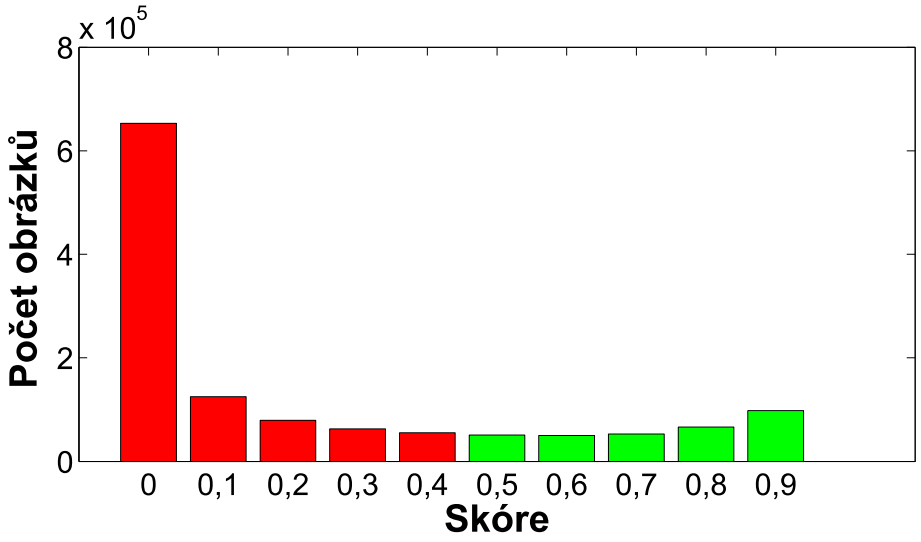
Pro řešení výše uvedených problémů jsem vytvořil skript v jazyce Python s využitím frameworku Caffe (viz kapitola 5.4.1). Vstupním argumentem skriptu je cesta ke složce obsahující obrazy určené ke klasifikaci. Skript pak rekurzivně prochází danou složku a pokud se v ní nachází obraz, tak jej klasifikuje s využitím modelu sítě *Places-CNN*, jehož výstupem jsou pravděpodobnosti pro jednotlivé kategorie scén (lesní cesta, písčité pobřeží, pole, zasněžená hora, moře, jezero, bažina, lyžařské středisko, louka, horská chata, sopka).

Toho jsem využil při sestavování algoritmu pro filtraci fotografií. Vybral jsme kategorie, které souvisí s horami, přírodní krajinou a venkovními scénami. Pro každou fotografii pak byl vypočítán součet pravděpodobností relevantních kategorií. Fotografie je přidána do datasetu pokud je součet větší než experimentálně určený práh 0,50. Hodnota prahu je kompromisem mezi variantami hodnot 0,40 a 0,60. Příliš velký práh by znamenal kvalitní dataset s malým množstvím obrazů. Malý práh by na druhou stranu do datasetu zařadil i obrazy, které by se pro řešenou úlohu nehodily.

Proces výběru relevantních kategorií a správné hodnoty prahu spočíval v manuální kontrole výsledků skriptu, který vytvářel HTML soubor. Na obrázku 5.3 je zobrazena část takového HTML souboru, který kromě samotných obrazů obsahuje údaje o 3 kategoriích s nejvyšším skóre a výsledný součet relevantních kategorií.

Skript byl mnohokrát pouštěn na 250 náhodně vybraných fotografiích z původní kolekce. Při každé iteraci jsem prošel vytvořený HTML soubor a následně změnil hodnotu prahu a seznam relevantních kategorií. Tento proces probíhal tak dlouho dokud skript nedával důvěryhodné výsledky. Následně byl skript upraven tak aby místo HTML souboru vytvářel dataset obsahující zmenšené kopie obrazů (256x256 pixelů), které prošly procesem filtrace. Pomocí klasifikace scén bylo odstraněno 75% fotografií z původní kolekce (rozložení skóre je zobrazeno v grafu na obrázku 5.4). Seznam vhodných kategorií je uveden v příloze A.

Obrázek 5.3: Ukázka výstupu skriptu určeného pro filtraci dodané kolekce fotek



Obrázek 5.4: Statistika filtrace fotografií

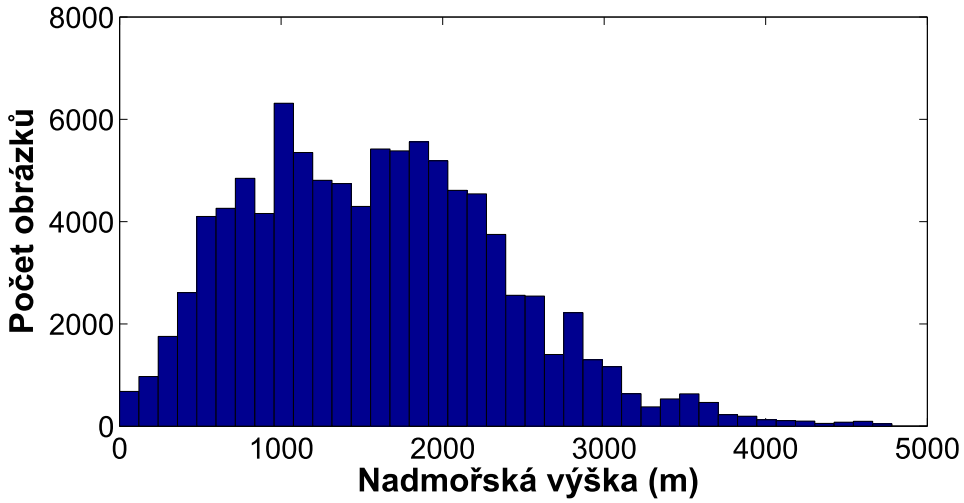
5.1.3 Vytvoření popisků pro data

Získání informací o nadmořské výšce kamery obrazu bylo provedeno ve dvou fázích. Nejdříve jsem vytvořil skript v programu MATLAB, který extrahoval z obrazů informace o GPS souřadnicích. Ty byly v obrazech uloženy ve formě komentářů. Komentář je součástí souborů JPEG. Výstupem skriptu byl textový soubor, který na každém řádků obsahoval cestu k obrazu a odpovídající GPS souřadnice. Textový soubor poté sloužil jako vstup BASH skriptu určenému k transformování GPS souřadnic na nadmořskou výšku. BASH skript využívá nástroje *Gengrid* [37]. Ten z GPS souřadnic určuje nadmořskou výšku pomocí speciálních výškových map, které zachycují oblast Alp.

BASH skript kromě převedení GPS souřadnic na nadmořské výšky zároveň slouží pro rozdělení celého datasetu na trénovací a testovací sadu. To bylo prováděno pomocí funkce modulo, kdy každý osmý a devátý obraz z datasetu byl zařazen do testovací sady. Důvodem bylo zachování shodného rozložení nadmořských výšek v obou sadách (pořadí položek ve vstupním souboru bylo určeno na základě místa pořízení fotografie). Původní kolekce se rozdělila v poměru 80:20. Výstupem skriptu byly 2 textové soubory. První soubor pro každý obraz obsahoval přesnou nadmořskou výšku určenou nástrojem Gengrid. Druhý soubor obsahoval kategorii nadmořské výšky pro každý obraz. BASH skript klasifikoval rozsah nadmořských výšek do intervalů po 100 metrech. Nejvyšší vrchol Alp je hora Mont Blanc s výškou 4807 m. Myšlenkou proto bylo vytvořit 50 kategorií, z nichž by 49 sloužilo pro kladné hodnoty nadmořských výšek a 1 by byla určena pro záporné nadmořské výšky. Ve výsledku se ukázalo, že kolekce neobsahuje obrazy pořízené v místech se zápornou nadmořskou výškou a nebo z míst s nadmořskou výškou nad 4800m. Dataset tedy obsahuje obrazy z 48 kategorií.

Zde je důležité zmínit, že během této fáze došlo k dalšímu zmenšení datasetu. Počet obrazů v datasetu se snížil na 161 607. Nástroj *Gengrid* bylo možné efektivně využít k odstranění fotografií jejichž místo leží mimo oblast pohoří Alp. K dispozici jsem měl pouze výškové mapy Alp a návratová hodnota nástroje *Gengrid* pro GPS souřadnice mimo oblast pokrytou výškovými mapami je nulová. V současnosti existuje několik dalších nástrojů pro zjištění nadmořské výšky z GPS souřadnic. Jedním z nich je například služba *GPS Visualizer*[5], která funguje pro libovolnou oblast. Použití služby *GPS Visualizer* vedlo k menší redukci obrazů v datasetu. Nebezpečí v použití většího datasetu ovšem

[5] www.gpsvisualizer.com/



Obrázek 5.5: Rozložení nadmořských výšek v datasetu

spočívá v zařazení fotografií pořízených mimo Alpy. Ty mohou obsahovat rysy, které nejsou charakteristické pro tuto oblast a degradovat tak úroveň natrénované sítě pro řešenou úlohu. V práci jsem proto použil menší dataset vytvořený pomocí nástroje *Gengrid*. Z výsledného datasetu bylo nakonec potřeba odstranit duplicitní obrazy. Tím se velikost datasetu znovu zmenšila na 98 136 obrazů[6]. Dataset pokrývá většinu území Alp (viz obrázek 5.6) a rozložení nadmořských výšek v něm je zobrazeno na obrázku 5.5.

5.1.4 Úprava datasetu

Na obrázku 5.5 je vidět, že počet obrazů v jednotlivých výškových hladinách datasetu není stejný. Vhodné rozložení obrazů v datasetu se určuje podle typu úlohy. Často se používá uniformní rozložení. Toho by šlo například dosáhnout pomocí metod náhodného oversamplingu a undersamplingu. Oversampling slouží k náhodnému výběru a duplikaci obrazů v kategoriích obsahujících malý počet trénovacích dat. Undersampling naopak slouží k náhodnému vyřazení obrazů z kategorií obsahujících příliš velký počet trénovacích dat [38]. Použití takto upraveného datasetu by ovšem vedlo k přetrénování klasifikátoru/regresoru. Počet obrazů s nadmořskou výškou nad 3000 m je malý a použití metody oversampling by vedlo k vytvoření velkého počtu stejných kopií obrazů. Použitý klasifikátor/regresor by potom nebyl schopný zobecnit řešenou úlohu. Druhý přístup spočívá v tom, že výsledný systém bude operovat nad daty se stejným rozložením jako má trénovací sada [39]. Jelikož není známo nad jakými daty bude systém operovat, lze předpokládat, že rozložení nadmořských výšek v datasetu odpovídá tomu, jak často lidé pořizují snímky v daných nadmořských výškách (rozložení nadmořských výšek v datasetu). Z tohoto důvodu jsem se nepoužil metodu oversamplingu/undersamplingu.

[6] Odstranění duplicitních obrazů vedlo ke změně poměru počtu obrazů v trénovací a testovací sadě na 87:13.

Obrázek 5.6: Pokrytí Alp obrazy z datasetu

5.2 EXIF data

EXIF (*Exchangeable image file format*) je specifikace pro formát metadat, která jsou digitálními fotoaparáty vkládána do pořízených fotografií. EXIF data mohou například obsahovat tyto informace:

- Značku a model fotoaparátu
- Datum a čas pořízení snímku
- Nastavení fotoaparátu (ohnisková vzdálenost, expoziční čas, citlivost, použití blesku atd.)
- Náhled snímku
- GPS souřadnice
- Typ snímané scény
- Informace o autorovi (lze přidat v editoru fotografií)

EXIF data obsahují důležité informace o způsobu pořízení dané fotografie. Tyto informace je možné použít při řešení úlohy odhadu nadmořské výšky kamery z obrazu (způsob jejich použití je popsán v kapitole 5.4.5).

Exif verze 1.0 byla navržena japonskou průmyslovou asociací JEITA (*Japan Electronics and Information Technology Industries Association*) [40]. V současnosti standard nikdo oficiálně nespravuje a není tedy dále vyvíjen. To vede k nejednosti v používání EXIF formátu různými výrobci fotoaparátů. Každý výrobce přidává do EXIF dat specifické informace a to způsobuje velké problémy při jejich automatizovaném zpracování.

EXIF data jsou podporována pouze některými souborovými formáty (např. JPEG a TIFF). V této práci se pracuje pouze s fotografiemi ve formátu JPEG a proto budou některé vlastnosti této specifikace stručně vysvětlený právě na souborovém formátu JPEG.

5.2.1 Získání EXIF dat

Při vytváření původní kolekce fotografií byla většina EXIF dat zahozena. Fotografie obsahovaly velmi omezené množství dodatečných informací (např. GPS souřadnice), které byly uloženy do struktury *komentář*. Pro provedení experimentů při odhadech bylo proto nutné nejdříve získat další EXIF data.

Stažení EXIF dat pro fotografie z původní kolekce bylo možné díky způsobu jejich pojmenování. Fotografie byly pojmenovány hash tagem, kterým jsou jednoznačně identifikovány na *Flickru*.

Skript pro stažení EXIF dat jsem vytvořil v jazyce Python za použití programového API rozhraní Flickr API[7]. Flickr API je možné použít pro interakci s *Flickr* ekosystémem (např. stahování fotografií, stahování informací o fotografiích, nahrávání fotografií, změna informací o fotografiích atd.). Výhodou je možnost dotazovat se *Flickru* pouze na informace o fotografiích. Nemusel jsem proto stahovat všechny fotografie znovu. Flickr API pro dotaz o informacích fotografie vrací strukturu ElementTree (lze nastavit i jiné formáty). ElementTree reprezentuje XML dokument jako stromovou strukturu. Každý uzel stromové struktury odpovídá jednomu XML elementu. Takto stažená EXIF data fotografie jsem parsoval do řetězce obsahující dvojice (název tagu, hodnota tagu). Řetězec dvojic byl následně použit jako vstupní argument nástroje *Exiftool*[8], pomocí kterého byly stažené EXIF data zapsány do fotografií. Exiftool je multiplatformní aplikace spouštěná z příkazové řádky. Slouží k manipulaci s EXIF daty ve fotografiích. Důvodem k použití aplikace Exiftool jsou chybějící Python knihovny pro zápis EXIF dat do fotografií.

EXIF data byla stažena pro 80% datasetu. Zbylé fotografie měly nastavený zákaz sdílet EXIF data. Pomocí nástroje *Exiftool* zároveň nebylo možné zapsat všechny stažená EXIF data do fotografií, protože některé EXIF tagy nejsou zapisovatelné.

5.2.2 Automatické zpracování EXIF dat

EXIF data jsou uložena v IFD (image file directory) struktuře. V hlavičce jsou uvedeny informace o velikosti IFD struktury v bytech a počet položek. Dále jsou uloženy všechny položky. Každá položka je jednoznačně identifikována názvem tagu (např. Model, DateTimeOriginal). Pomocí názvu tagu je možné získat hodnotu položky (nikon D3200, 12:31:25-01/29/2012) [40].

Fotografie v datasetu byly pořízeny širokou škálou různých modelů fotoaparátů. To způsobilo problémy při automatickém zpracování stažených EXIF dat. Různí výrobci do EXIF dat ukládají různé množství informací. Některé vhodné informace pro řešení odhadu nadmořské výšky kamery z obrazu nejsou obsaženy ve všech fotografiích a nelze je použít (typ scény, typ počasí atd.). Dalším problémem je různé pojmenování tagů pro stejnou informaci a různý způsob formátování hodnoty tagu.

Existuje velké množství EXIF tagů obsahujících důležité informace pro odhad nadmořské výšky. Kvůli variaci v množství EXIF dat v obrazech datasetu jsem pro experimenty vybral pouze taková EXIF data, která jsou obsažená ve většině datasetu (způsob jejich použití je popsán v kapitole 5.4.5). Všechny obrazy z datasetu obsahují informace o GPS souřadnicích jejich místa pořízení. V experimentech jsem tuto informaci nepoužíval, protože existují lepší metody pro odhad nadmořské výšky z GPS souřadnic (výškové mapy, model WGS84 atd.). Seznam použitých tagů a jejich význam při odhadu nadmořské výšky uvádím v následujícím výčtu:

[7] www.flickr.com/services/api/
[8] www.sno.phy.queensu.ca/~phil/exiftool/

Obrázek 5.7: Změna scény během dne

- **Čas** - čas pořízení snímků. Řídí se nastavením hodin ve fotoaparátu. Snímky pořízené ze stejného místa v různou denní dobu se budou lišit. Pozice slunce se během dne mění a to se na snímcích může projevit změnou stínů nebo různou intenzitou světla ve scéně (viz obrázek 5.7). Rizikem při použití tohoto tagu je možnost špatného nastavení hodin ve fotoaparátu. Neexistuje však jednoduchý způsob jak zjistit zda byly fotografie pořízeny v čase uvedeném v tagu datetimeoriginal. Předpoklad pro použití této informace byl, že většina fotoaparátů má hodiny nastaveny správně.

- **Datum** - Datum pořízení snímku. Datum pořízení snímku nás informuje o specifických povětrnostních podmínkách, které mohou nastat v dané části roku a délce slunečního svitu během dne. Dva snímky pořízené ze stejného místa ve stejnou denní dobu, ale v různém ročním období se mohou výrazně lišit (viz obrázek 5.8). Informace o pozici dne během roku může odhad nadmořské výšky kamery z obrazu zpřesnit. Jako příklad lze uvést sníh. Sníh se ve scéně může objevit jednak kvůli specifickému ročnímu období nebo vysoké

Formát	Příklad
Slovní reprezentace data	September 8th 2008
YYYY-MM-DD	2008-09-08
YYYY/MM/DD	2008/09/08
YYYY:MM:DD	2008:09:08

Tabulka 5.1: Příklady formátů data

Obrázek 5.8: Změna scény způsobená rozdílnou roční dobou

nadmořské výšce. Při zpracování data bylo nutné pracovat s několika způsoby formátování (viz tabulka 5.1). K parsování řetězce obsahujícího datum jsem použil regulární výrazy. Stejně jako u času zde existuje možnost špatného nastavení data ve fotoaparátu.

- **Ohnisková vzdálenost (ohnisko)** - vzdálenost mezi optickým středem objektivu a digitálním snímačem při zaostření na nekonečno. Dnešní kompaktní fotoaparáty i zrcadlovky používají objektivy, které mají proměnlivé ohniskové vzdálenosti a dovolují snímanou scénu přiblížit či oddálit (ohnisko realizuje pouze výřez části scény na snímek) [41].

- **Expoziční čas** - doba, po kterou je závěrka fotoaparátu otevřena a umožňuje světlu dopadat na obrazový senzor fotoaparátu. Senzor v zásadě počítá fotony dopadajícího světla. Čím delší je expoziční doba, tím více fotonů dopadne na senzor fotoaparátu. Doba expozice ovlivňuje podobu těch částí snímku, které se v době expozice pohybují. Příliš dlouhá doba expozice způsobí jejich rozmazání, velmi krátká doba expozice naopak jejich "zamrznutí" v čase [41].

- **Clonové číslo** - Clonou (otvorem ve středu objektivu) lze řídit množství světla které projde objektivem. Čím větší je průměr clony, tím více světla projde objektivem a dopadne na senzor. Množství světla, které dopadne na senzor, ovšem závisí i na vzdálenosti clony od senzoru (ohniskové vzdálenosti). Proto se v praxi zavedlo clonové číslo, které s ohniskovou vzdáleností počítá [41]. Clonové číslo je definováno jako

$$F = \frac{f}{d}$$

kde F je clonové číslo, f je ohnisková vzdálenost v mm a d je průměr clony v mm.

- **ISO citlivost** - Nastavuje citlivost snímače fotoaparátu na světlo. Ve skutečnosti se mění pouze způsob, jakým fotoaparát manipuluje se získaným světlem. Při zvýšení ISO citlivosti se světlo, které dopadlo na snímač vynásobí, a tím se simuluje zvýšení citlivosti samotného snímače. Z tohoto postupu vyplývá zásadní problém. S násobením získaného světla se násobí i šum, který při expozici na snímači vzniká. Hodnota ISO citlivosti nám říká kolik světla bylo ve scéně během pořizování snímku a zároveň je významným faktorem ovlivňující úroveň šumu se kterou musíme počítat [41].

- **Expoziční koeficient** - Informace o nastavení fotoaparátu (clonové číslo, expoziční čas, ISO) jsem použil k určení absolutního množství světla ve scéně v době pořízení snímku. Množství světla ve scéně EC (expoziční koeficient) je vypočítáno jako

$$EC = 2log_2(F) - log_2(t) - log_2\left(\frac{ISO}{100}\right)$$

kde F je clonové číslo, t je expoziční čas a ISO je citlivost snímacího senzoru [42].

- **Zorné pole** - část prostoru, kterou je schopný fotoaparát zachytit (prostor, ze kterého do něj přicházejí světelné paprsky). Zorné pole určuje část scény, která se zobrazí na snímek. To pomáhá při odhadu projekce scény na senzor fotoaparátu a tedy i při odhadu vzdáleností objektů ve scéně. Informaci o zorném poli obsahuje pouze velmi malá část datasetu. Jedná se však o velmi bohatou informaci a proto jsem tento údaj dopočítal z ostatních metadat pomocí rovnice

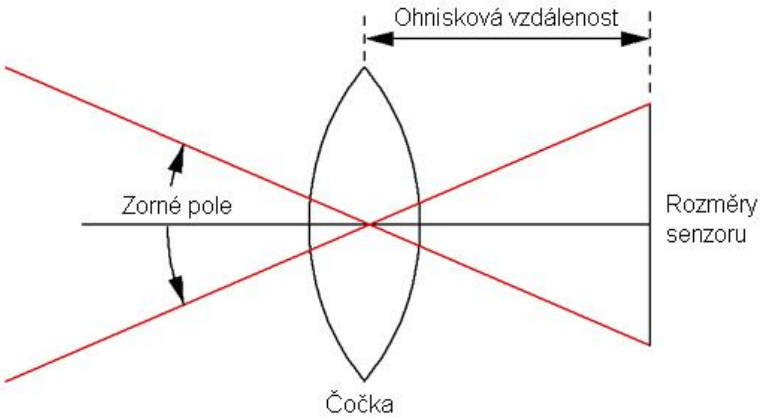
$$FOV = 2atan\left(0{,}5\ \frac{SEN}{f}\right)\frac{180}{\pi}$$

kde FOV je zorné pole, SEN je rozměr senzoru a f je ohnisková vzdálenost. Hodnota zorného pole je závislá na velikosti použitého senzoru (např. oproti větším senzorům mají menší senzory při stejné ohniskové vzdálenosti užší zorné pole - viz obrázek 5.10).

V současnosti neexistuje bezplatný program s databází obsahující velikost senzorů pro všechny modely fotoaparátu. Velikost senzoru jednotlivých modelů jsem proto manuálně hledal na stránkách výrobců. Statisticky jsem zjistil, že při vytvoření databáze pro 100 modelů je možné dopočítat tento údaj pro 40% obrazů z datasetu (seznam modelů v příloze B).

Při zpracování všech použitých tagů jsem kontroloval nestandardní hodnoty (např. defaultní hodnoty, datum z dávné minulosti/budoucnosti atd.). Pokud obraz obsahoval takové hodnoty v EXIF datech, tak nebyl použit v následujících experimentech. V algoritmu pro zpracování EXIF dat se zároveň kontrolují jednotky výše uvedených parametrů. Pokud se jednalo o nestandardní jednotky, tak byla hodnota tagu převedena (např. 2.4 cm na 24.0 mm).

Obrázek 5.9: Fotografie s různou expoziční hodnotou (9EV, 14EV)



Obrázek 5.10: Ilustrace zorného pole

5.3 Extrahování příznaků

Velký vliv na kvalitu výsledků metod strojového učení mají použité příznaky. V tradičním přístupu byly vhodné příznaky navrhovány na základě znalostí a zkušeností odborníků. Pro obecné úlohy klasifikace a vyhledávání fotografií jsou to například GIST [11] a SIFT (*Scale-invariant feature transform*) [10], které zachycují globální respektive lokální vzhled obrazu. Výběr vhodných příznaků pro konkrétní úlohu je dlouhý a náročný proces, zvláště pokud nejsou dostupné prameny o dřívější práci, tak jako v tomto případě.

Je možné použít existující příznaky určené pro odlišné úlohy. Je však velmi složité určit kritérium popisující jednotlivé nadmořské výšky. Na obrázku 5.11 jsou vidět 3 obrazy s různou nadmořskou výškou kamery. Pro člověka je téměř nemožné určit všechny charakteristické rysy jednotlivých nadmořských výšek. Jako příklad lze uvést použití histogramu barev, kdy se bude předpokládat, že místa s menší nadmořskou výškou budou pravděpodobně obsahovat více vegetace a v jejich histogramu barev bude zastoupena větším podílem zelená barva. Naopak místa s větší nadmořskou výškou budou nejspíše pokryty sněhem, nebo zde bude převažovat kamenitý povrch a v histogramu barev tedy bude zastoupena bílá nebo šedá barva. Takových nepřesných předpokladů by bylo možné vytvořit více. Bylo by však nutné provést kvalitní studii přírodních podmínek v dané oblasti pro všechny nadmořské výšky.

Jak již bylo uvedeno dříve, je zde možné spatřovat určitou podobnost s úlohou klasifikace scén. Ta má za cíl rozpoznat scény jako pole, les, moře, venkovní scéna a podobně. S určitou abstrakcí problému lze odhad nadmořské výšky kamery z obrazu chápat jako hlubší rozpoznávání scén v kategoriích scén vyskytujících se v pohoří Alp. Obrazy z těchto kategorií by se tak dále rozlišovaly podle jejich nadmořské výšky.

Experimenty během kterých jsem trénoval klasifikátory/regresory (3-vrstvá neuronová síť) s využitím GIST deskriptoru obrazů a jejich barevných histogramů selhaly. Úspěšnost odhadu byla podobná náhodě. Alternativou k ručně navrženým příznakům, která v posledních letech nabírá na popularitě, je hluboké učení, které využívá rostoucího výkonu výpočetní techniky a velkých datových sad k naučení vhodných příznaků přímo z dat. Jednou z úspěšných metod hlubokého učení pro obrazová data jsou hluboké konvoluční sítě, které dosahují v současnosti nejlepších výsledků na úlohách klasifikace obrazu [22], rozpoznávání lidí podle obličeje [43] a detekce objektů [44].

Obecnost konvolučních sítí je vykoupena většími nároky na množství trénovacích dat a velkou výpočetní náročností jejich trénování. Z důvodu nedostatku dostupných materiálů o dřívějších pracích

Obrázek 5.11: Venkovní scény s různou nadmořskou výškou

na odhadu nadmořské výšky kamery z obrazu jsem použil právě konvoluční neuronové sítě. Návrh architektury a výsledky experimentů jsou popsány v následujících kapitolách.

5.4 Odhad nadmořské výšky pomocí konvolučních sítí

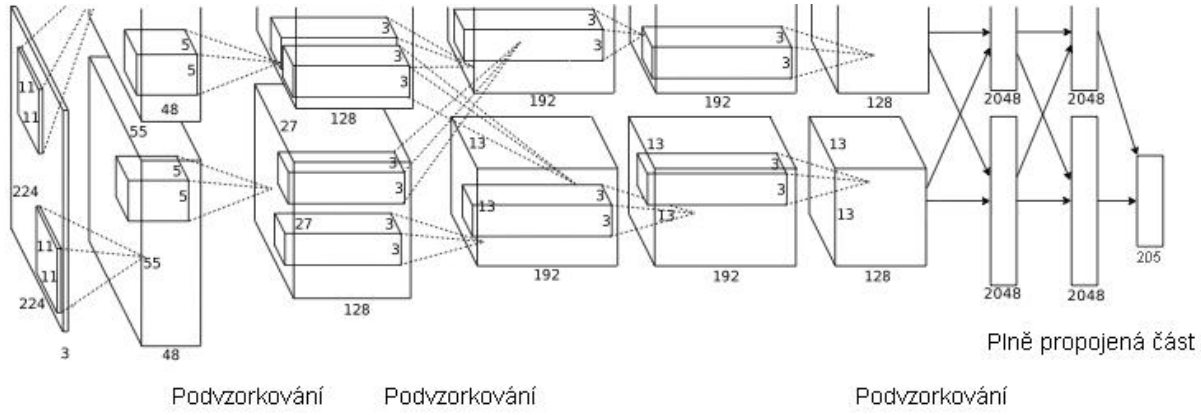
V této sekci je popsán návrh a implementace modelů konvolučních sítí pro odhad nadmořské výšky kamery z obrazu. Automatický odhad nadmořské výšky je složitý problém a svým charakterem je podobný obecným klasifikačním úlohách, kde jsou fotografie řazeny do několika stovek kategorií [22][30]. Konvoluční sítě dosahují v současnosti na takto komplexních úlohách nejlepších výsledků. Sítě používané v těchto úlohách mají milióny parametrů (viz obrázek 5.12) a vyžadují statisíce trénovacích obrazů. Omezení můžeme spatřovat právě v množství potřebných trénovacích dat. Velké množství trénovacích dat je nutné pro natrénování příznaků, které budou schopné danou úlohu dostatečně generalizovat. Velikost vytvořeného datasetu je výrazně menší než datasety použité pro trénování sítí *AlexNet* a *Places-CNN*, a proto jsem zvolil postup adaptace příznaků z existující sítě. Při návrhu všech modelů sítě jsem vycházel z modelu sítě natrénovaného na datasetu *Places205* [22], který je popsán v kapitole 5.4.2.

V provedených experimentech testuji vhodnost zvoleného postupu. Dále zkoumám použitelnost dodatečných informací o obrazu (EXIF data) na úloze odhadu nadmořské výšky. V závěru kapitoly je popsán systém pro odhad nadmořské výšky kamery z obrazu, který je založen na použití modelů natrénovaných během experimentů.

5.4.1 Framework Caffe

K implementaci modelů konvolučních sítí jsem použil framework Caffe[9]. Ten je určený pro návrh konvolučních neuronových sítí. Modely se konfigurují pomocí strukturovaných textových souborů namísto psaní zdrojových kódů. Jeho největší výhodou je optimalizace kódu pro běh na GPU. Trénování větších sítí je tím urychleno a trvá v řádu několika dní. Při provádění některých experimentů bylo potřeba změnit zdrojové kódy frameworku Caffe. Provedené změny jsou popsány v programové dokumentaci.

[9] caffe.berkeleyvision.org



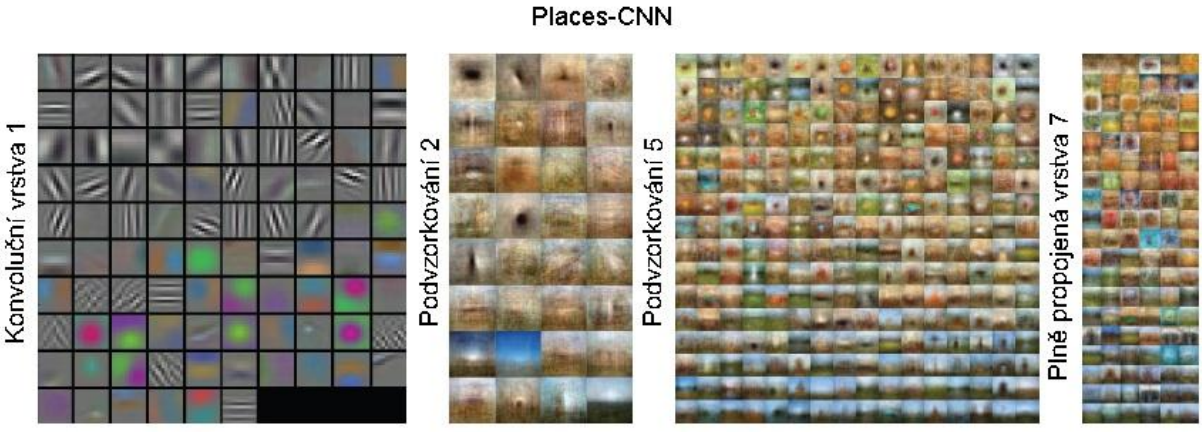
Obrázek 5.12: Architektura konvoluční sítě *Places-CNN*

5.4.2 Places-CNN

Places-CNN [30] je konvoluční neuronová síť určená k rozpoznávání scén. V současnosti dosahuje nejlepších výsledků v úloze klasifikace venkovních/vnitřních scén. Síť je natrénovaná na obrazech z datasetu *Places205*.

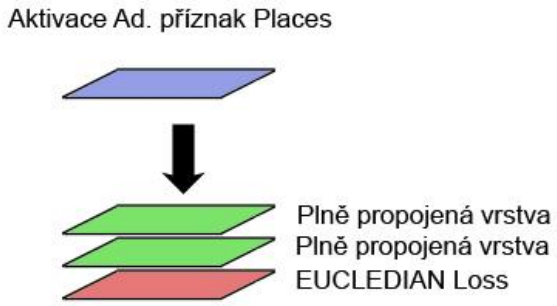
Places205 je subset datové sady *Places database* (viz kapitola 3.5). Obsahuje téměř 2,5 milionu obrazů z 205 kategorií, které byly náhodně vybrány z datové sady *Places database*. Počet obrazů v jednotlivých kategoriích není stejný a pohybuje se v rozmezí 5 000 - 15 000.

Architektura sítě je shodná s architekturou použitou v Caffe reference network [22]. Vstupem sítě jsou barevné 2D obrazy zmenšené na velikost 256x256 pixelů. Hlavním stavebním prvkem sítě jsou konvoluce po kterých následuje ReLU jednotka (viz kapitola 2.5.5). Za první, druhou a pátou konvoluční vrstvu je vložena vrstva podvzorkování (max-pooling), která redukuje rozlišení faktorem čísla 2. Aktivace první a druhé konvoluční vrstvy jsou lokálně normalizovány. Výstup konvoluční části je veden do plně propojených vrstvev, které mají postupně 4096, 2048 a 205 neuronů. Výstup poslední plně propojené vrstvy je vstupem *softmax* funkce, která produkuje distribuci nad 205 třídami. Vizualizace natrénovaných příznaků je zobrazena na obrázku 5.13.



Obrázek 5.13: Natrénované příznaky v síti *Places-CNN* [30]

5.4.3 Příznaky Places



Obrázek 5.14: Architektura sítě *Příznaky Places*

Předešlé práce [43] [45] ukazují, že příznaky extrahované z existujících sítí jsou dostatečně obecné a dávají informaci i pro podobné úkoly, na které nejsou původně určené. V tomto experimentu jsem zjišťoval jak zafungují příznaky sítě *Places-CNN* na datasetu *Alps100k.*

Extrahoval jsem aktivace první plně propojené vrstvy (4096 neuronů) pro všechny obrazy datasetu. Extrahované aktivace reprezentují globální deskriptor obrazu a slouží jako vstup pro 2-vrstvou neuronovou síť, jejíž parametry jsou na začátku trénování náhodně inicializovány (viz obrázek 5.14). Výstupní vrstva této sítě je lineární s jedním neuronem, který přímo odhaduje nadmořskou výšku. Síť byla trénována pomocí optimalizační kritéria *mean square error*, které je vhodné pro regresní problémy, kde lze předpokládat normální rozložení chyb. Regresní model k tomu využívá vrstvu *eucledian loss*, která počítá chybu pomocí funkce

$$\frac{1}{2N}\sum_{i=1}^{N}\left|x_i^1 - x_i^2\right|_2^2$$

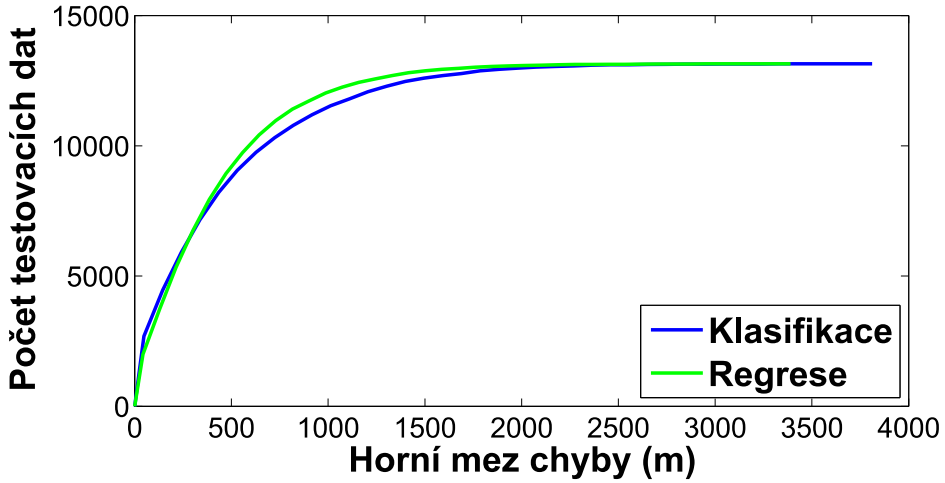
kde N je počet odhadů, x^1 je výstup neuronové sítě a x^2 je reálná hodnota nadmořské výšky.

Po ověření, že příznaky extrahované ze sítě *Places-CNN* fungují na řešené úloze (viz kapitola 6.2) jsem kromě regresního modelu natrénoval i model klasifikační. Myšlenkou bylo zjistit jaká metoda strojového učení bude lépe fungovat při odhadu nadmořské výšky kamery z obrazu. Architektura obou modelů se liší použitou výstupní vrstvou. Výstupní vrstva klasifikačního modelu má 48 neuronů a je vstupem funkce *softmax*, která produkuje distribuci nad 48 třídami.

Experiment porovnávající přesnost odhadu regresního a klasifikačního modelu ukázal, že odhady regresního modelu na testovací sadě jsou přesnější (viz tabulka 5.2). Pravděpodobnost získání odhadu s chybou do určité hranice je možné vyčíst z obrázku 5.15. Na základě těchto výsledků jsem v dalších experimentech pracoval pouze s regresními modely.

Model	RMSE(m)
Regresní model	592,44
Klasifikační model	683,99

Tabulka 5.2 - Porovnání výsledků klasifikačního a regresního modelu na testovací sadě datasetu Alps100k.



Obrázek 5.15: Graf zobrazující počet testovacích dat (osa y) s chybou do určité hodnoty (osa x)

5.4.4 Adaptované příznaky Places

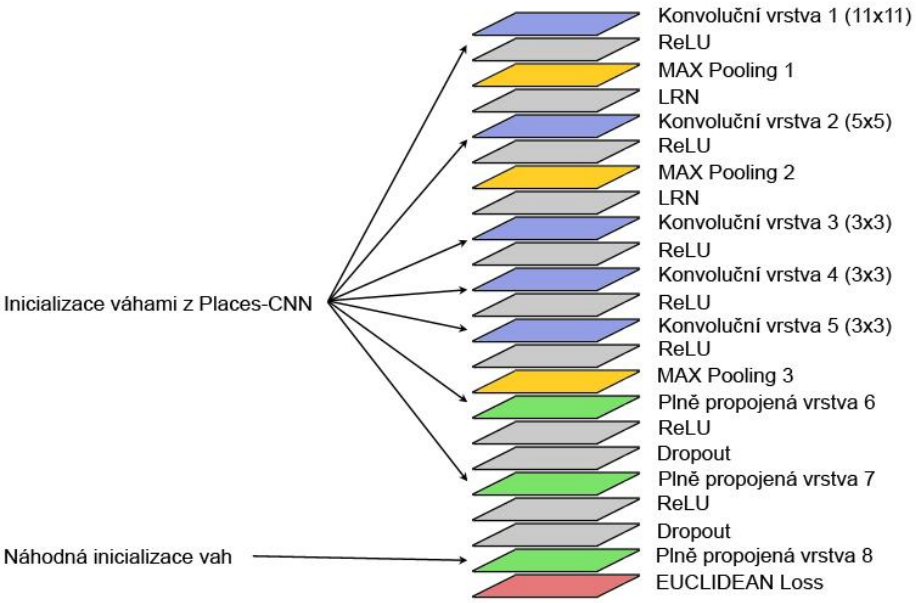
Lepších výsledků než přímým použitím existující sítě pro extrakci příznaků lze většinou dosáhnout adaptací existující sítě pro konkrétní úlohu - tzv. *Fine-tunningem* [44] [46]. Technika *Fine-tunning* slouží k adaptaci vah již existující sítě. Uživateli dovoluje vybrat konkrétní vrstvy, jejichž parametry budou inicializovány pomocí parametrů jiné sítě. Parametry ostatních vrstev sítě jsou inicializovány náhodně.

Trénování takto nastaveného modelu probíhá rychleji. Zároveň je možné dosáhnout lepších výsledků než při náhodné inicializaci všech vah sítě. Metoda *Fine-tunning* se proto využívá v případech, kdy k trénování nemáme dostatek trénovacích dat nebo dostatek času [47]. Při použití techniky *Fine-tunning* je nutné patřičně upravit konfigurační soubor trénování. Myšlenkou je konfigurovat učení sítě tak, aby probíhalo rychle ve vrstvách s náhodně inicializovanými parametry a pomaleji ve vrstvách, jež byly inicializovány parametry druhé sítě.

Tento postup jsem použil při trénování modelu *Adaptované příznaky Places*. Architektura sítě je kromě poslední vrstvy shodná s architekturou sítě *Places-CNN* (viz tabulka 5.3). Kromě poslední vrstvy jsou váhy sítě inicializovány pomocí parametrů z modelu *Places-CNN* (viz obrázek 5.16). Taková sít je pak učena celá pomocí metody *Stochastic Gradient Descent*.

Vrstva	konv1	podv1	konv2	podv2	konv3	konv4	konv5	podv5	pp6	pp7	pp8
Jednotky	96	96	256	256	384	384	256	256	4096	2048	1
Jádro	11x11	3x3	5x5	3x3	3x3	3x3	3x3	3x3	-	-	-
Příznaky	55x55	27x27	27x27	13x13	13x13	13x13	13x13	6x6	-	-	-

Tabulka 5.3: Architektura konvoluční sítě *Adaptované příznaky Places*



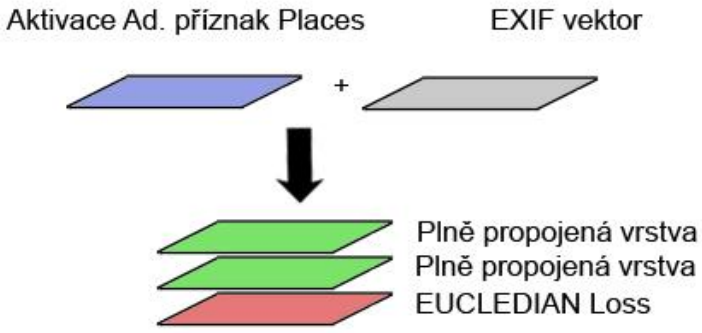
Obrázek 5.16: Architektura konvoluční sítě *Adaptované příznaky Places* ve frameworku Caffe

5.4.5 Adaptované příznaky Places + EXIF

Téměř polovina datasetu obsahuje EXIF data, která mohou být použita jako dodatečné informace o obrazu a upřesnit tak odhad nadmořské výšky. Pro tento experiment jsem vytvořil neuronovou síť jejímž vstupem byly aktivace první plně propojené vrstvy z modelu *Adaptované příznaky Places* a zakódovaná EXIF data ve formě EXIF vektoru (viz obrázek 5.17).

Během implementace každé aplikace založené na neuronových sítích je potřeba nejdříve prozkoumat data se kterými budeme pracovat. Data může být potřeba konvertovat na jinou formu aby byly pro neuronové sítě smysluplné. Klíčem k úspěšnému návrhu aplikace je dostatečné porozumění problému, tak abychom věděli, které informace jsou pro nás důležité. Způsob jakým jsou data transformována a reprezentována ovlivňuje schopnost sítě pochopit problém.

Hodnota dat může být spojitá nebo diskrétní. Někdy je data možné reprezentovat oběma způsoby. Teplotní data mohou být reprezentována skutečnou teplotou nebo například jednou ze čtyř hodnot: zima, chladno, pokojová teplota, horko. Reprezentace pomocí kategorií je vhodná, pokud pracujeme s daty, které lze přirozeně klasifikovat. Umělé řazení spojitých hodnot do kategorií není vhodné, protože pro síť je velmi složité naučit se příklady, jejichž hodnoty jsou velmi blízko k hranicím dvou kategorií.



Obrázek 5.17: Architektura regresního modelu *Adaptované příznaky Places + EXIF*

Běžnou chybou je naopak reprezentovat unikátní hodnoty pomocí spojitých vstupů. Jako příklad uvedu reprezentaci měsíce v roce pomocí čísla od 1 do 12. Neuronová síť při takové reprezentaci předpokládá, že data jsou spojitá a číslo určitým způsobem vyjadřuje kvalitu vstupu (březen je "lepší" než leden, apod.). Z tohoto důvodu je potřeba měsíc v roce reprezentovat pomocí kategorií.

Výběr mezi spojitou a diskrétní reprezentací je často velmi obtížný. Například když pracujeme s daty, které jsou spojité, ale jejich rozložení není uniformní. Pro dosažení nejlepšího výkonu v takových případech je nutné otestovat několik způsobů reprezentace [48].

Jako vhodný způsob kódování všech použitých EXIF dat se ukázal kód 1 z N. Kód 1 z N představuje takový způsob kódování, kdy jednomu stavu z N odpovídá aktivní stav jedné stopy. Jednoduchý příklad kódování 4 kategorií je zobrazen v tabulce 5.4. Před samotným kódováním bylo nutné EXIF data rozdělit do vhodných kategorií. Způsob rozdělení EXIF dat do kategorií je uveden v následujícím výčtu:

ID kategorie	Kód 1 z 4
1	1000
2	0100
3	0010
4	0001

Tabulka 5.4: Příklad kódování 1 z 4

- **ISO citlivost** - Citlivost se standardně udává v ISO jednotkách a odpovídá citlivosti klasického filmu. Každá sousední hodnota na ISO stupnici mění citlivost senzoru na světlo právě 2x. Typická stupnice ISO je:

...	50	100	200	400	800	1600	3200	...

Pokud zvýšíme ISO citlivost 2x (např. z ISO 100 na ISO 200), stačí ke stejné expozici poloviční množství světla (množství fotonů). V praxi se u některých digitálních zrcadlovek používá i jemnější dělení, kdy mezi sousedními hodnotami na výše uvedené stupnici je buď ještě jedna mezihodnota (1/2) nebo dvě mezihodnoty (1/3 a 2/3). Rozsah hodnot ISO citlivosti v datasetu je od 32 do 10 000. Pro kódování 1 z N jsem hodnoty ISO citlivostí rozdělil do 6 kategorií. Kategorie reprezentují intervaly hodnot, které jsou soustředěny kolem základních hodnot ISO stupnice (viz tabulka 5.5). Velikost intervalů není uniformní a postupně se zvětšuje.

ISO stupnice	Biny
100	(<150)
200	(150, 250)
400	(250, 550)
800	(550, 1050)
1600	(1050, 2150)
3200	(2150<)

Tabulka 5.5: Způsob rozdělení ISO citlivostí v datasetu do kategorií

Clonová řada	Biny
1.0	(<1,2)
1.4	(1,2 , 1,68)
2.0	(1,68 , 2,4)
2.8	(2,4 , 3,4)
4.0	(3,4, 4,8)
5.6	(4,8 , 6,7)
8	(6,7 , 10,6)
11	(10,6, 13,2)
16	(13,2, 18,6)
22	(18,6, 26,0)
30	(26,0, 41)
42	(41<)

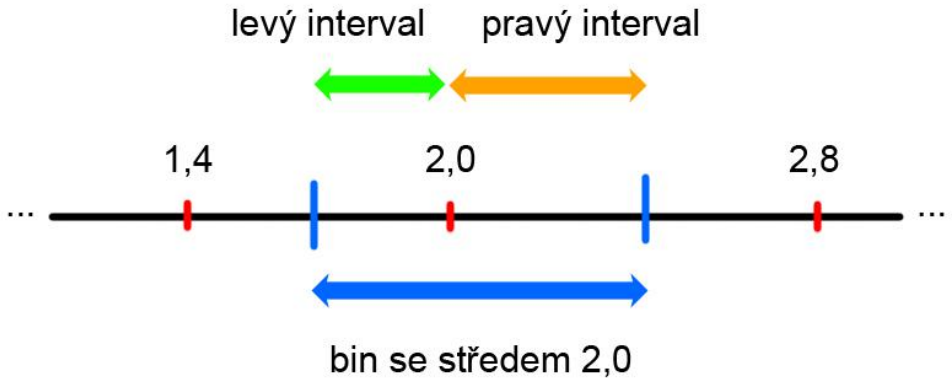
Tabulka 5.6: Způsob rozdělení ISO citlivostí v datasetu do kategorií

- **Clonové číslo** - Množství světla procházející clonou je úměrné ploše otvoru clony. Základní stupnice clonových čísel není řada hodnot 2x, ale jen 1.4x. Při zvětšení průměru clony 1.4x, vzroste plocha otvoru clony 2x. Tím vzroste na dvojnásobek i množství světla. Typická clonová řada je:

...	1	1,4	2	2,8	4	5,6	8	11	16	22	...

Stejně jako u ostatních parametrů nastavení fotoaparátu se můžeme setkat s jemnějším dělením clonové řady. Clonová čísla dělím do 12 kategorií.

Velikost binů není uniformní (viz tabulka 5.6). Každý bin je tvořen 3 částmi: středem, levým intervalem a pravým intervalem (viz obrázek 5.18). Střed binu se počítá stejným způsobem jako clona (násobky $\sqrt{2}$). Levý interval binu představuje polovinu intervalu ohraničeném aktuálním středem a předcházejícím středem. Pravý interval naopak představuje polovinu intervalu ohraničeném aktuálním středem a následujícím středem.



Obrázek 5.18: Části binu se středem v hodnotě 2,0

- **Expoziční čas** - Základní stupnice expozičních časů je:

...	4	2	1	1/2	1/4	1/8	1/15	1/30	1/60	1/125	...

Každá sousední hodnota na stupnici expozičních časů mění množství světla, které dopadne na senzor 2x. Snaha používat "rozumná čísla" na stupnici expozičních časů vedla k jejich zaokrouhlování. To znamená, že krok mezi sousedními hodnotami není vždy dvojnásobek. V praxi se můžeme setkat s jemnějším dělením expozičních časů než je uvedeno na výše uvedené stupnici. Expoziční časy jsem v práci dělil do 9 kategorií (viz tabulka 5.7).

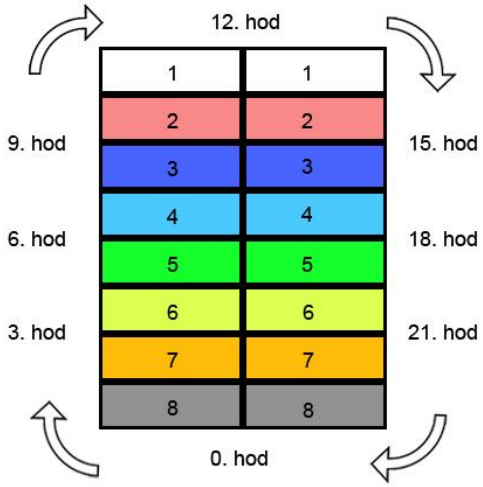
Stupnice expozičních časů	Biny
8,4,2,...	(>1/10)
1/15	(1/10, 1/20)
1/30	(1/20, 1/40)
1/60	(1/40, 1/80)
1/125	(1/80, 1/160)
1/250	(1/160, 1/320)
1/500	(1/320, 1/640)
1/1000	(1/640, 1/1280)
1/2000	(<1/1280)

Tabulka 5.7: Způsob rozdělení expozičních časů do kategorií

- **Expoziční koeficient** - Měří se na záporné logaritmické stupnici o základu dvě – zvýšení o 1 EC tedy odpovídá polovině propuštěného světla. Rozsah expozičních koeficientů ve vytvořeném datasetu je od 0,432 do 21,266. Pro expoziční koeficienty je vytvořeno 17 kategorií lišících se vždy o 1 EC. Expoziční koeficienty vypočítané pro fotografie zaokrouhluji a přidávám do příslušných kategorií. Expoziční koeficienty větší než 16 EC jsou řazeny do jedné kategorie.

- **Zorné pole** - V práci používám vertikální zorné pole i horizontální zorné pole. Zorné pole je reprezentováno jako úhel (viz kapitola 5.2.2). Rozsah horizontálního zorného pole pro obrazy z datasetu je 2,08° až 134,08°. Pro vertikální zorné pole je rozsah 1,39° až 115,34°. Pro oba údaje jsem vytvořil 12 kategorií s krokem po 10 stupních. Všechny úhly větší než 110° jsou řazeny do jedné kategorie.

- **Čas** - Jedná se o specifický údaj, který musí být transformován do smysluplné podoby pro neuronovou síť. Jak již bylo uvedeno na začátku této kapitoly, tak s časem nemůžeme pracovat jako se spojitou veličinou. Dva časové údaje nemají odlišnou kvalitu (11 hodin není "lepších" než 9 hodin). Čas je proto potřeba chápat jako unikátní údaj, kterým můžeme popsat periodické přírodní děje. Denní dobu kóduji jako 1 z 16 a 1 z 8. První přístup spočívá v rozdělení dne na 16 částí. Každá část trvá 90 minut a tvoří jednu kategorii. 90 minut je dostatečně dlouhá doba během které dochází k významným změnám ve scéně (posun slunce, změna stínů atd.). Druhý přístup spočívá v rozdělení dne na 8 částí, které jsou symetrické kolem 12 hodin (viz obrázek 5.19). Předpoklad je, že neuronová síť je schopná lépe využít informaci kódovanou více způsoby.



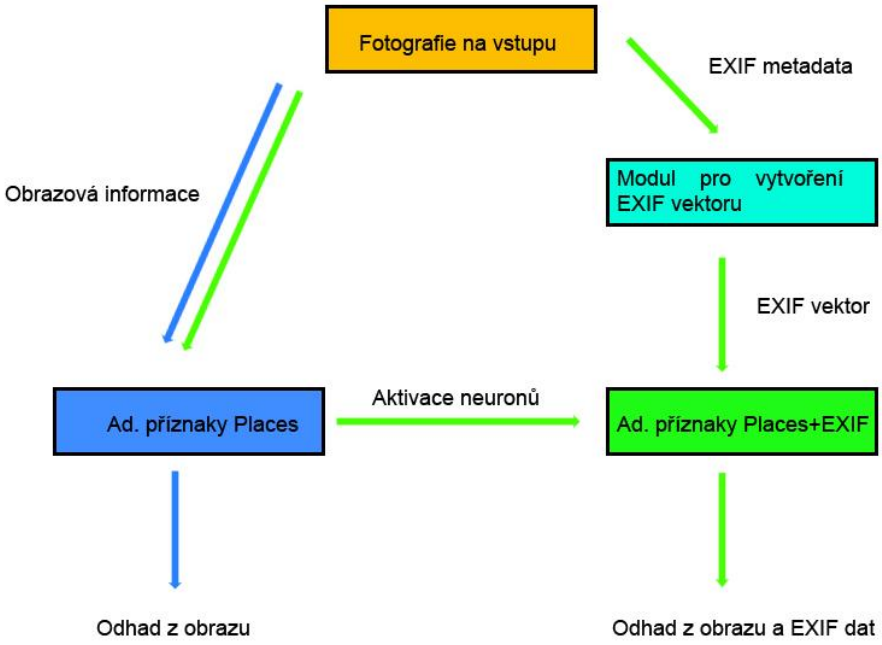
Obrázek 5.19: Rozdělení dne do 8 částí, které jsou symetrické kolem 12. hodiny. Číslo vyjadřuje ID kategorie daného časového intervalu.

- **Datum** - Stejně jako u času se jedná o specifický údaj, který kóduji 2 způsoby. První způsob je kód 1 z 16, kde je rok rozdělen na 16 částí, jejichž délka je 22,8125 dne. Druhý způsob spočívá v kódování data pomocí ročního období (jaro, léto, podzim, zima) jako 1 z 4.

- **Rok** - Fotografie použité pro vytvoření datové sady jsou pořízeny během let 2000 až 2014. Rok představuje časový interval za který planeta Země oběhne Slunce. Pro kódování roku pořízení snímku používám kód 1 z 15.

Všechna kódovaná EXIF data jsou přidána do vektoru jehož počet dimenzí je 127. Největší aktivace extrahované z regresního modelu *Adaptované příznaky Places* se pohybují kolem hodnoty 10. To je výrazně větší než hodnoty obsažené v EXIF vektoru. Pro přiřazení stejné priority informacím v EXIF vektoru je nutné jej vynásobit dostatečně velkou konstantou (viz kapitola 6.2.3).

5.4.6 Systém pro odhad nadmořské výšky kamery z obrazu



Obrázek 5.20: Architektura systému pro odhad nadmořské výšky z obrazu. Zelená cesta představuje způsob odhadu z obrazu a EXIF dat. Modrá cesta reprezentuje odhad pouze z obrazu.

Implementovaný systém je navržen jako command-line aplikace, aby šel jednoduše použít jako součást jiných skriptů (např. geolokalizace atd.). K aplikaci tedy nebylo vytvořeno grafické uživatelské rozhraní a ovládá se pomocí argumentů příkazové řádky. Systém byl implementován v jazyce Python a je volán pomocí BASH skriptu, který se stará o zpracování vstupních argumentů a management proměnných SHELL prostředí, které slouží k nastavení správného běhu frameworku Caffe.

Odhad nadmořské výšky z obrazu je prováděn pomocí regresních modelů vytvořených během výše popsaných experimentů. Aplikace funguje ve 2 základních módech:

1. Odhad nadmořské výšky pouze z obrazu
2. Odhad nadmořské výšky z obrazu při použití EXIF dat

Při provádění odhadu nadmořské výšky pouze z obrazu je využíváno regresního modelu *Adaptované příznaky Places*, který byl natrénován na celém datasetu. Výstupem je číselná hodnota reprezentující odhad konvoluční sítě (viz obrázek 5.16).

Druhý mód odhadu je založen na použití regresního modelu *Adaptované příznaky Places+EXIF*, který byl natrénován na *EXIF subsetu* (viz kapitola 6.2). Aplikace v tomto módu nejdříve přečte EXIF data pomocí Python knihovny ExifRead 2.0.2[10] a zkontroluje zda fotografie obsahuje všechny potřebná data. Pokud EXIF data fotografie nejsou kompletní, tak aplikace vrátí chybu a odhad musí být proveden pouze na základě obrazové informace. Omezením druhého módu je velikost vytvořené databáze *modelů fotoaparátů a velikostí jejich senzorů*, které se používají pro výpočet vertikálního a horizontálního zorného pole. U fotografií obsahujících všechna potřebná EXIF data a jež byly pořízeny fotoaparátem, který se nachází v databázi modelů je vytvořen EXIF vektor jehož velikost je 127. EXIF vektor je generován pomocí knihoven vytvořených při implementaci regresního modelu *Adaptované příznaky Places+EXIF*. Výsledný EXIF vektor je vynásoben

[10] pypi.python.org/pypi/ExifRead

konstantou 40 aby měla mírně vyšší prioritu než aktivace sítě *Adaptované příznaky Places* (viz kapitola 6.2.3). Následně je použit regresní model *Adaptované příznaky Places*, ze kterého jsou extrahovány aktivace 1. plně propojené vrstvy. EXIF vektor a aktivace ze sítě *Adaptované příznaky Places* jsou spojeny ve vstupní vektor jehož velikost je 4223 (4096 + 127). Vstupní vektor je nakonec poslán na vstup regresního modelu *Adaptované příznaky Places+EXIF,* jehož výstupem je odhad nadmořské výšky (viz obrázek 5.17).

Argumenty příkazové řádky jsou podrobně popsány v programové dokumentaci diplomového projektu. Slouží k nastavení módu predikce, charakteru výstupu (pouze odhad nebo obsáhlejší informace o odhadu) a cesty k fotografii.

Na obrázku 5.21 je zobrazena ukázka výstupu implementovaného systému. Pod každým obrazem je uvedena informace o skutečné nadmořské výšce, odhad založený pouze na obrazové informaci a nakonec odhad využívající obrazovou informaci i EXIF data.

Obrázek 5.21: Ukázka výstupu systému pro odhad nadmořské výšky z obrazu

6 Vyhodnocení

Tato kapitola se zabývá vyhodnocením systému pro odhad nadmořské výšky kamery z obrazu, který byl popsán v kapitole 5. Na začátku kapitoly jsou diskutovány výsledky všech natrénovaných konvolučních sítí. Na konci kapitoly je uvedeno porovnání výsledků člověka a konvolučních sítí na úloze odhadu nadmořské výšky z obrazu.

6.1 Porovnání adaptace vah a náhodné inicializace vah

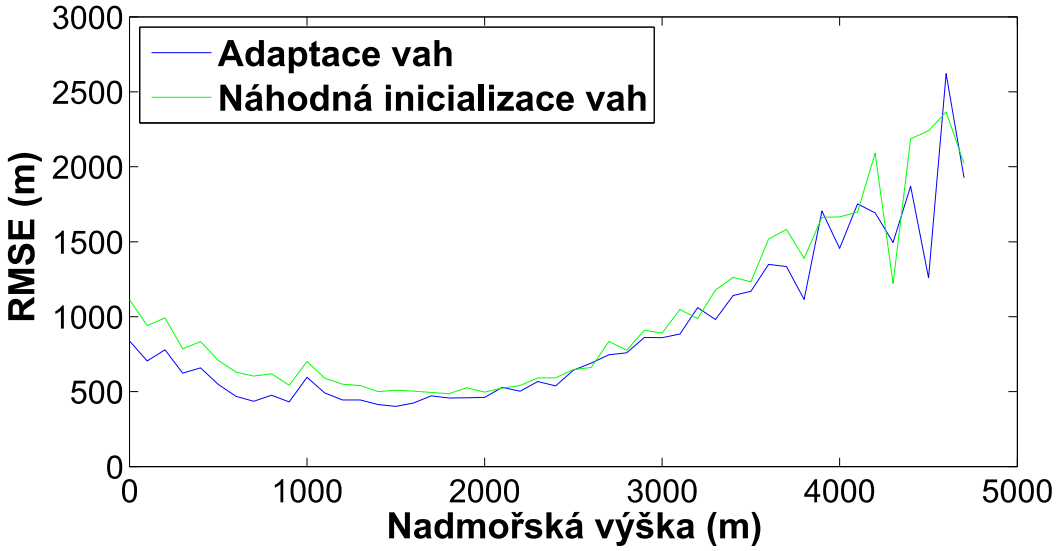
Velikost vytvořeného datasetu *Alps100k* je výrazně menší než datasety použité pro trénování sítí *AlexNet* a *Places-CNN*. Při použití stejné architektury sítě může dojít k přetrénování i přeučení konvoluční sítě. Proto jsem provedl experiment, jehož cílem bylo zjistit vhodnost vytvořeného datasetu při trénování navržené konvoluční sítě s náhodně inicializovanými váhami.

Referenční řešení je založeno na nejlepším neinformovaném regresoru, který používá apriorní znalosti o datasetu. Výstup takového regresoru je pevný a jeho hodnota je rovna průměrné hodnotě nadmořských výšek v dané testovací sadě. Konvoluční síť s náhodně inicializovanými váhami dosahuje na použité testovací sadě lepších výsledků (viz tabulka 6.1). Průměrná chyba odhadu je ovšem stále příliš vysoká. Postup založený na adaptaci příznaků z existující sítě efektivně využívá znalostí naučených na jiné úloze. Průměrná chyba odhadu pro všechny nadmořské výšky je menší (viz tabulka 6.1). Patrné je to především u nejnižších a nejvyšších nadmořských výšek (viz obrázek 6.1). Chyba odhadu je v těchto oblastech vyšší téměř o 350 m a směrem ke středu rozsahu nadmořských výšek klesá. Důvodem větší chyby odhadu je menší počet trénovacích obrazů pro tyto nadmořské výšky. Konvoluční síť s náhodně inicializovanými váhami se pak přetrénuje na omezeném množství obrazů.

Znalosti získané na úloze klasifikace scén pomáhají konvoluční síti s adaptovanými příznaky redukovat vliv přetrénování a tím zmenšit chybu při odhadu. Z tohoto důvodu jsem v dalších experimentech zvolil postup adaptace příznaků z existující sítě.

Model	RMSE(m)
Mean Baseline	786.42
Regresní model s náhodnou inicializací vah	652.07
Regresní model - *Fine-tunning*	560.40

Tabulka 6.1: porovnání regresních modelů na celé testovací sadě



Obrázek 6.1: Porovnání hodnot RMSE podle nadmořské výšky u modelu s náhodně inicializovanými váhami a modelu s adaptovanými váhami.

6.2 Porovnání vytvořených regresních modelů na EXIF subsetu

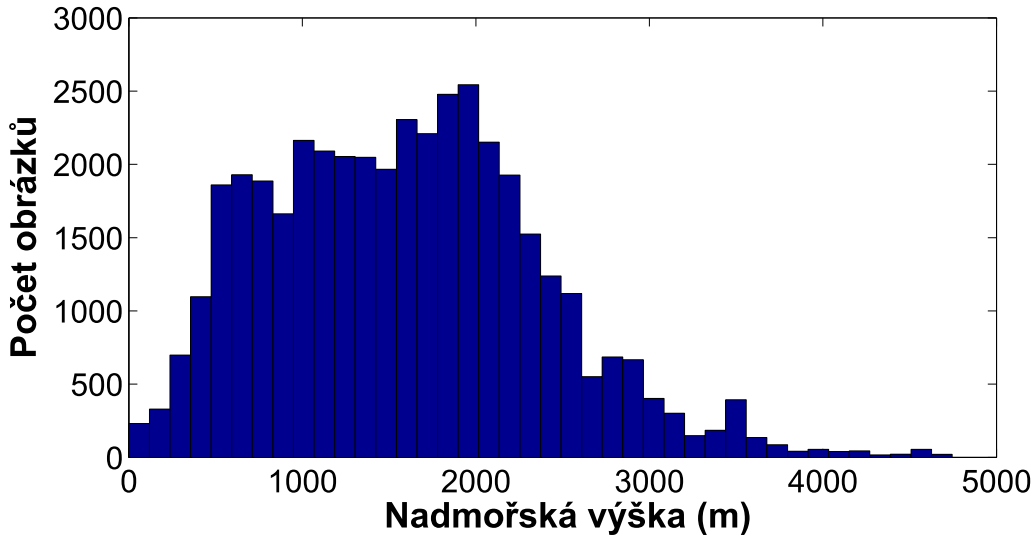
Pro korektní porovnání jednotlivých variant modelů sítí jsem vytvořil subset datasetu *Alps100k* (dále jen *EXIF dataset*, který pro každou fotografii obsahuje EXIF data použitá při tvorbě varianty *Adaptované příznaky Places + EXIF*. Rozložení nadmořských výšek v *EXIF datasetu* je zobrazeno na obrázku 6.2. Trénovací sada obsahuje 34 889 obrazů. Testovací sada má 5046 obrazů. EXIF dataset jsem následně použil pro přetrénování modelů *Příznaky Places* a *Adaptované příznaky Places*. Všechny modely tak lze přímo porovnávat, protože byly trénovány/testovány na stejných datech. V následující části popisuji chování modelů na testovací sadě *EXIF datasetu*.

6.2.1 Příznaky Places

Příznaky extrahované z konvoluční sítě *Places-CNN* se ukázaly jako dostatečně obecné. Průměrná chyba odhadu na testovací sadě *EXIF datasetu* (viz tabulka 6.3) se málo liší od průměrné chyby modelu *Adaptované příznaky Places* (v uživatelském testu dosahuje lepších výsledků než člověk). Dobré výsledky na této úloze přisuzuji faktu, že příznaky sítě *Places-CNN* byly trénovány na vysokém počtu různorodých venkovních scén.

6.2.2 Adaptované příznaky Places

Při experimentu používajícím metodu *Fine-tunning* bylo potřeba brát v úvahu odlišnost přístupu použitém při trénování regresního modelu *Příznaky Places* a regresního modelu *Adaptované příznaky Places*. Srovnáním modelu popsaném v kapitole 5.4.4 a modelu *Příznaky Places* by nebylo jasné, zda lepších výsledků není dosaženo naučenou informací ve vrstvách následujících po 1. plně propojené vrstvě. V experimentu *Příznaky Places* se použily pouze aktivace neuronů 1. plně propojené vrstvy reprezentující globální deskriptor obrazu. K odstranění možného konfliktu jsem použil stejnou



Obrázek 6.2: Rozložení nadmořských výšek v EXIF subsetu

architekturu sítě jako má regresní model *Příznaky Places*. Oba přístupy se lišily pouze v použitých trénovacích datech. Vstupem modelu *Adaptované příznaky Places* vytvořeném v tomto experimentu byly aktivace 1. plně propojené vrstvy regresního modelu, který je popsán v kapitole 5.4.4. Odlišnost obou přístupů tedy spočívá v použití aktivací neuronů ze dvou odlišných konvolučních sítí, kdy jedna byla adaptovaná na vytvořenou datovou sadu.

Adaptace konvoluční sítě na *EXIF dataset* vede k mírnému zlepšení výsledků regrese (viz tabulka 6.3). Velikost *EXIF datasetu* je malá a přínos metody *Fine-tunning* se v tomto experimentu výrazně neprojevil.

6.2.3 Adaptované příznaky Places + EXIF

Předpoklad, že dodatečné informace k obrazu pomohou při odhadu nadmořské výšky kamery se potvrdil. Vstupem konvoluční sítě byla kombinace obrazové informace a EXIF vektoru. EXIF vektor byl násoben konstantou 10, aby hodnoty v obou částech vstupního vektoru byly stejné. Konvoluční síť byla schopna využít větší množství informací k zpřesnění odhadu a zmenšení chyby na hodnotu 530,76 m. Provedený experiment zároveň ukázal vhodnost použitých EXIF dat pro podobné úlohy.

EXIF vektor násobený konstantou 10 by teoreticky měl mít stejnou prioritu ve vstupním vektoru jako část tvořená extrahovanými aktivacemi. Velikost obou částí se však výrazně liší (4096 a 127). Neuronová síť pak během odhadu nemusí dostatečně využít informace obsažené právě v EXIF vektoru. Z tohoto důvodu jsem provedl další experiment, kterým jsem testoval vhodnou velikost konstanty pro násobení původního EXIF vektoru (kód 1 z n). Natrénoval jsem další 3 neuronové sítě se stejnou architekturou, které byly trénovány na odlišných vstupech. Vstupy neuronových sítí se lišily

Velikost konstanty	RMSE(m)
10	530,76
20	514,52
40	491,14
60	501,85

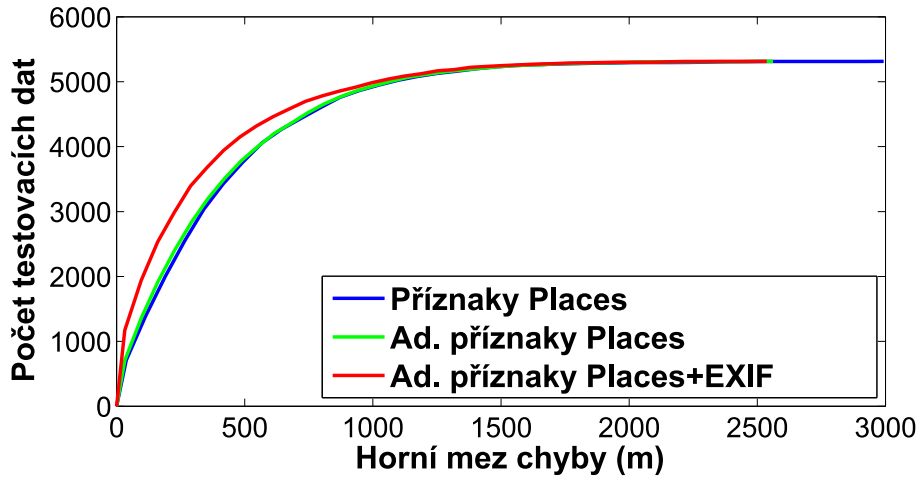
Tabulka 6.2: Vliv podoby EXIF vektoru na průměrnou chybu odhadu

částí reprezentující EXIF vektor. Nové EXIF vektory byly vytvořeny násobením původního vektoru konstantami 20,40 a 60. Postupné zvyšování priority EXIF vektoru vedlo ke snížení chyby odhadu. Minimum chyby odhadu 491,14 m bylo dosaženo s neuronovou sítí používající vstup s EXIF vektorem násobeným konstantou 40. Násobení původního EXIF vektoru konstantou větší než 40 naopak vedlo ke zvýšení chyby od dosaženého minima chyby odhadu (viz tabulka 6.2). Z tohoto důvodu jsem prioritu EXIF vektoru dále nezvyšoval a v experimentech jsem používal neuronovou síť natrénovanou datech obsahujících EXIF vektor násobený konstantou 40.

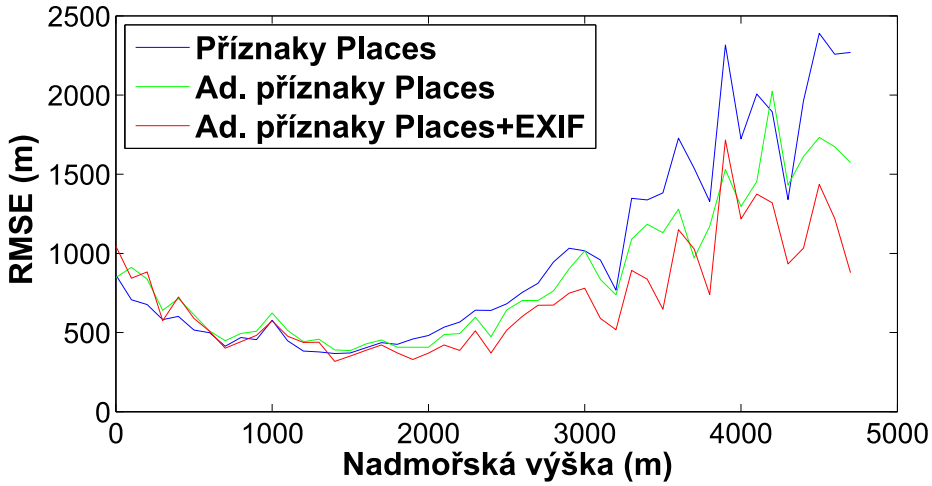
Regresní model *Adaptované příznaky Places+EXIF* dosahuje mezi použitými modely nejlepších výsledků na testovací sadě *EXIF datasetu* (viz obrázek 6.3, tabulka 6.3). Dodatečné informace o obrazu pomohly konvoluční sítí lépe predikovat obrazy s vyšší nadmořskou výškou (viz obrázek 6.4).

Model	RMSE(m)
Příznaky Places	565,71
Adaptované příznaky Places	550,14
Adaptované příznaky Places + EXIF	491,14

Tabulka 6.3 - porovnání regresních modelů na testovací sadě EXIF subsetu



Obrázek 6.3: Graf zobrazující počet testovacích dat (osa y) s chybou do určité hodnoty (osa x)



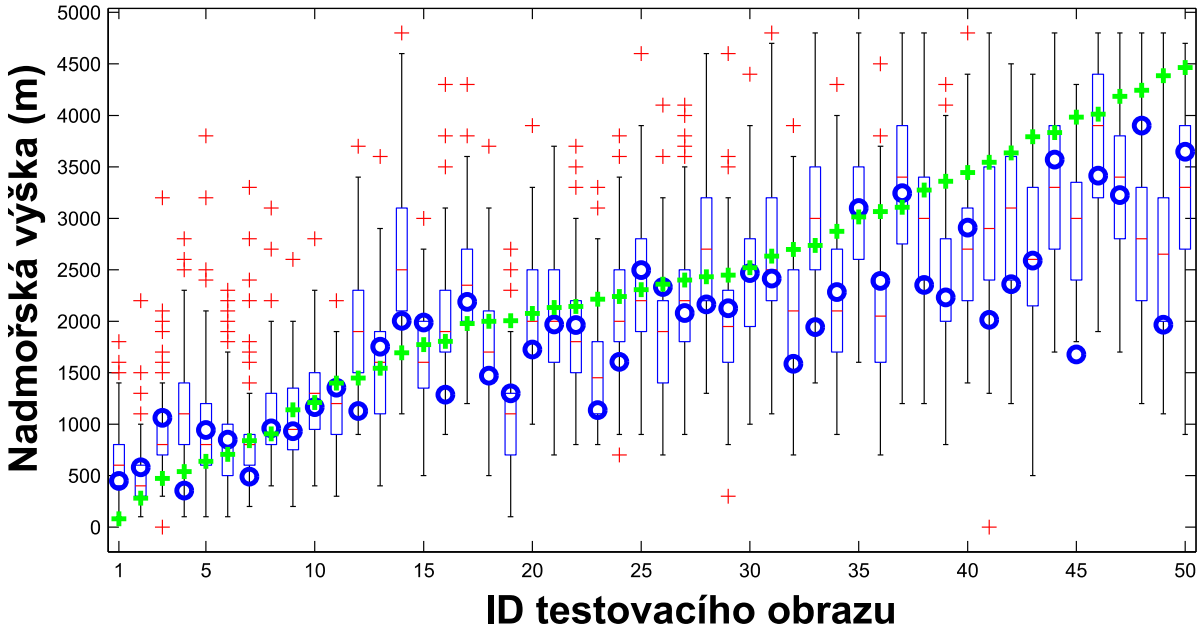
Obrázek 6.4: Porovnání hodnot RMSE podle nadmořské výšky u použitých regresních modelů

6.3 Porovnání výkonu člověka a konvolučních sítí

Na základě provedeného uživatelského testu je možné konstatovat, že konvoluční sítě dosahují v této úloze lepších výsledků než člověk (tabulka 6.4) Jelikož účastníci testu měli k dispozici pouze obrazovou informaci, tak je porovnání provedeno s variantou *Adaptované příznaky Places*[11]. Do grafu na obrázku 6.5 jsou zaneseny odhady lidí i konvoluční sítě pro obrazy použité v uživatelském testu. Konvoluční sítě mají stejně jako lidé největší problémy u obrazů pořízených ve vyšších nadmořských výškách. V této práci to může být způsobeno malým množstvím fotografií z těchto výšek v datasetu.

	RMSE (m)
Mean baseline	1154.43
Člověk	879,94
Adaptované příznaky Places	712.86

Tabulka 6.4: Porovnání průměrné chyby člověka a natrénované konvoluční sítě.



Obrázek 6.5: Odhady člověka a konvoluční sítě na 50 obrazech z uživatelského testu. Každý "box" odpovídá jednomu obrazu a zobrazuje rozložení odhadů lidí. (zelené křížky - správná hodnota, modrá kolečka - *Adaptované příznaky Places*, červená čára - průměrná hodnota odhadu účastníků testu pro daný obraz).

[11] Výrazný rozdíl ve velikosti chyby v tabulkách 6.3 a 6.4 je způsoben odlišným rozložením nadmořských výšek v použitých testovacích sadách.

7 Závěr

Náplní této diplomové práce je automatický odhad nadmořské výšky z obrazu pomocí metod počítačového vidění. Úloha spočívá v odhadu nadmořské výšky kamery z barevné fotografie, která byla pořízena standardním způsobem v oblasti pohoří Alp (fotografie zachycuje scénu z pohledu člověka). Charakteristické rysy krajiny se mění nejen s nadmořskou výškou, ale i s polohou na zemském povrchu. Dvě místa se stejnou nadmořskou výškou, ale jinou zeměpisnou polohou by se mohly velmi lišit. Oblast pohoří Alp je dostatečně malá, aby bylo možné předpokládat podobnost scén ve stejných výškových hladinách. K dispozici jsem dostal kolekci čítající 1,2 milionu fotografií. Tato kolekce nebyla uspořádaná a většina fotografií nesouvisela s horami nebo venkovní krajinou. Nejdříve bylo nutné odstranit nesouvisející fotografie pomocí algoritmu, který využívá model konvoluční sítě *Places-CNN*. Z vybraných fotografií jsem vytvořil datovou sadu, která obsahuje u každého obrazu údaj o nadmořské výšce kamery.

V práci popisuji dva základní způsoby, jak řešit obecné úlohy rozpoznávání obrazu. Dále uvádím informace o úlohách podobných odhadu nadmořské výšky kamery z obrazu (odhad aktuálního počasí, odhad nadmořské výšky z bezpilotního letounu atd.). Z důvodu chybějících informací o této problematice jsem provedl uživatelský test. Pro testování schopností člověka na této úloze jsem vytvořil webovou aplikaci. Uživatelský test ukázal, že průměrná chyba člověka při odhadu nadmořské výšky z obrazu je 879 m. Největší problémy má člověk při určování míst s vysokou nadmořskou výškou. Kromě toho, že získané výsledky poskytují nové a zajímavé poznatky o schopnostech člověka, slouží také jako reference pro automatické metody, které v práci navrhuji.

K řešení problému jsem použil konvoluční neuronové sítě, které v současnosti dosahují v podobných úlohách velmi dobrých výsledků. Konvoluční neuronové sítě dosahují na úloze odhadu nadmořské výšky kamery z obrazu lepších výsledků než člověk. Odhady nadmořské výšky jsou přibližné a průměrná chyba se na obrazech použitých v uživatelském testu pohybuje kolem hodnoty 712 m. Největší chyby dosahují natrénované regresní modely v nadmořských výškách, pro které je ve vytvořené datové sadě nedostatek trénovacích obrazů. Tuto chybu je možné snížit použitím dodatečných informací o obrazu. Systém pro automatický odhad nadmořské výšky založený na konvolučních sítích lze kombinovat s jinými metodami počítačového vidění (BoW atd.) [49].

Na experimentech je dobře prezentována potřeba vhodného datasetu pro řešenou úlohu. Dataset vytvořený v této práci je unikátní sada obrazů z horského prostředí doplněná informací o nadmořské výšce kamery. Tento dataset sice není tak obsáhlý jako obecné datasety používané při trénování konvolučních sítí, ovšem i na něm bylo možné úspěšně provést experimenty, které do budoucna dávají naději na vytvoření řešení, které by se dalo použít v praxi. Pro použití systému v oblastech mimo pohoří Alp je nutné vytvořit nové, obsáhlejší datové sady pro adaptaci systému na odlišné typy venkovních scén. Věřím, že při trénování konvoluční sítě na dostatečně velkém datasetu, který bude splňovat požadavky na hustotu a rozmanitost [30], je možné dosáhnout ještě přesnějších odhadů nadmořské výšky kamery z obrazu.

Literatura

[1] ČADA, Václav. Úvod do geodézie. In: *Přednáškové texty z geodézie* [online]. [cit. 2015-01-06]. Dostupné z: <http://gis.zcu.cz/studium/gen1/html/ch02s03.html#id301837>.

[2] FUKUSHIMA, Kunihiko. Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position [online]. [cit. 2015-05-17]. Dostupné z: <http://www.cs.princeton.edu/courses/archive/spr08/cos598B/Readings/Fukushima1980.pdf>.

[3] LANGLEY, Pat. Elements of Machine Learning. San Francisco: Morgan Kaufmann Publishers, 1996. ISBN 1-55860-301-8.

[4] NG, Andrew. Machine Learning. In: *Coursera.org* [online]. 2014 [cit. 2014-12-27]. Dostupné z: <https://class.coursera.org/ml-005/lecture/>.

[5] JAVIDI, Bahram. Image Recognition and Classification: Algorithms, Systems, and Applications. New York: Eastern Hemisphere Distribution, 2002. ISBN 0-8247-0783-4.

[6] FIKKER, Jaroslav. Šum v digitální fotografii. In: fotoaparat.cz [online]. 2004-07-29 [cit. 2014-12-28]. Dostupné z: <http://www.fotoaparat.cz/article/7193/1>.

[7] VERNON, David. Machine Vision - Automated Visual Inspection and Robot Vision. Prentice Hall Europe, 1991. ISBN 978-0-13-543380-5.

[8] KHALIL, Osama. Viewpoint Invariant Object Detector [online]. [cit. 2014-12-28]. Dostupné z: <http://arxiv.org/ftp/arxiv/papers/1212/1212.0030.pdf>.

[9] ŠPANĚL, Michal. Statistické rozpoznávání a shlukování [online]. 2012 [cit. 2014-12-27]. Dostupné z: <https://www.fit.vutbr.cz/study/courses/POV/private/lectures/pov_02_statisticke_rozpoznavani.pdf>.

[10] LOWE, David. Distinctive Image Features from Scale-Invariant Keypoints [online]. [cit. 2014-10-10]. Dostupné z: <https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf>.

[11] OLIVA, Aude a Antonio TORRALBA. Modeling the shape of the scene: A holistic representation of the spatial envelope [online]. [cit. 2014-10-10]. Dostupné z: <http://www.cnbc.cmu.edu/cns/papers/Oliva-Torralba-IJCV-01.pdf>.

[12] SCHWARZ, Daniel. Lineární a adaptivní zpracování dat: SYSTÉMY a jejich popis v časové doméně [online]. [cit. 2015-03-22]. Dostupné z: <http://www.iba.muni.cz/esf/res/file/bimat-prednasky/linearni-a-adaptivni-zpracovani-dat/LaAZD-02.pdf>.

[13] HLAVÁČ, Václav. Předzpracování obrazu v lokálním okolí [online]. [cit. 2015-03-22]. Dostupné z: <http://cmp.felk.cvut.cz/~hlavac/TeachPresCz/11DigZprObr/21ImagPreprocCz.pdf>.

[14] Convolution [online]. In: *iOS Developer Library*. [cit. 2015-03-22]. Dostupné z: <https://developer.apple.com/library/ios/documentation/Performance/Conceptual/vImage/ConvolutionOperations/ConvolutionOperations.html>.

[15] CUNG, Bianca, Justin HSUEH, Levon KOLESNIKOV a Khang LU. Regression and Machine Learning [online]. [cit. 2015-04-21]. Dostupné z: <http://www.math.ucla.edu/~wittman/10c.1.11s/Lectures/Raids/Regression.pdf>.

[16] LECUN, Yann. Deep Learning Tutorial [online]. In: *Talks by Members of CBLL* .2013-06-16 [cit. 2014-11-26]. Dostupné z: <http://www.cs.nyu.edu/~yann/talks/lecun-ranzato-icml2013.pdf>.

[17] DENG, Li a Yu, DONG. Deep Learning Methods and Applications. In: *http://research.microsoft.com* [online]. [cit. 2014-12-06]. Dostupné z: <http://research.microsoft.com/pubs/209355/DeepLearning-NowPublishing-Vol7-SIG-039.pdf>.

[18] BENGIO, Yoshua. Learning Deep Architectures for AI [online]. 2009 [cit. 2014-12-29]. Dostupné z: <http://www.iro.umontreal.ca/~bengioy/papers/ftml.pdf>.

[19] VOLNÁ, Eva. NEURONOVÉ SÍTĚ 1 [online]. 2008 [cit. 2014-12-30]. Dostupné z: <http://www1.osu.cz/~volna/Neuronove_site_skripta.pdf>.

[20] CHALUPNÍK, Vitalij. Biologické algoritmy (5) - Neuronové sítě. In: *root.cz* [online]. 2012-06-14. [cit. 2014-12-15]. Dostupné z: <http://www.root.cz/clanky/biologicke-algoritmy-5-neuronove-site>.

[21] CULURCIELLO, Eugenio. Artificial and robotic vision - lecture3_1 - convolutional neural networks. In: *Youtube.com* [online]. 2013-01-14 [cit. 2014-12-08]. Dostupné z: <https://www.youtube.com/watch?v=vXj4z_Vo8G0>.

[22] KRIZHEVSKY, Alex, Ilya SUTSKEVER a Geoffrey HINTON. ImageNet Classification with Deep Convolutional Neural Networks [online]. 2012 [cit. 2014-12-15]. Dostupné z: <http://www.cs.toronto.edu/~fritz/absps/imagenet.pdf>.

[23] HRADIŠ, Michal. Deep CNN for Computer Vision. 2014 [cit. 2015-01-04].

[24] LUDWIG, Jamie. Image Convolution [online]. [cit. 2014-11-23]. Dostupné z: <http://web.pdx.edu/~jduh/courses/Archive/geog481w07/Students/Ludwig_ImageConvolution.pdf>.

[25] NAYAK, Amiya a Ivan STOJMENOVIC,. Handbook of Applied Algorithms: Solving Scientific, Engineering, and Practical Problems. Wiley-IEEE Press, 2008. ISBN 978-0-470-04492-6.

[26] BISKUP, Roman. Možnosti neuronových sítí. Praha, 2009. Disertační práce. Česká zemědělská univerzita v Praze, Fakulta provozně ekonomická. Vedoucí práce Anna Čermáková. Dostupné z: <http://www.pef.czu.cz/cs/?dl=1&f=13043>.

[27] SHAMMA, David, Jia LI, Lyndon KENNEDY a Bart THOMÉE. Science Powering Product: Yahoo Weather [online]. 2014-11-24. [cit. 2014-12-22]. Dostupné z: <http://yahoolabs.tumblr.com/post/103469857701/science-powering-product-yahoo-weather>.

[28] ROSER, Martin a Frank MOOSMANN. Classification of Weather Situations on Single Color Images [online]. 2008-06-04 [cit. 2014-12-26]. Dostupné z: <http://www.mrt.kit.edu/z/publ/download/Roser_al2008iv.pdf>.

[29] HAYS, James a Alexei EFROS. IM2GPS: estimating geographic information from a single image [online]. [cit. 2014-11-08]. Dostupné z: <http://graphics.cs.cmu.edu/projects/im2gps/im2gps.pdf>.

[30] ZHOU, Bolei, Agata LAPEDRIZA, Jianxiong XIAO a Aude OLIVA. Learning Deep Features for Scene Recognition using Places Database [online]. [cit. 2014-10-14]. Dostupné z: <http://places.csail.mit.edu/places_NIPS14.pdf>.

[31] XIAO, Jianxiong a Aude OLIVA. Sun database: Large-scale scene recognition from abbey to zoo [online]. [cit. 2015-03-02]. Dostupné z: <http://cs.brown.edu/~hays/papers/sun.pdf>.

[32] QUATTONI, Ariadna a Antonio TORRALBA. Recognizing Indoor Scenes [online]. [cit. 2015-03-02]. Dostupné z: <http://people.csail.mit.edu/torralba/publications/indoor.pdf>.

[33] CHERIAN, Anoop, Jon ANDERSH, Vassilios MORELLAS a Bernard METTLER. Autonomous Altitude Estimation Of A UAV Using A Single Onboard Camera [online]. [cit. 2014-11-05]. Dostupné z: <http://lear.inrialpes.fr/people/cherian/papers/altitudepaperiros09.pdf>.

[34] BEWICK, Viv, Liz CHEEK a Jonathan BALL. Statistics review 9: one-way analysis of variance [online]. [cit. 2015-02-11]. Dostupné z: <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC420045/>.

[35] BURROUGHS, William. Climate: Into the 21st Century. Cambridge University Press,2003. ISBN 9780521792028.

[36] CURRAN, Kevin, John CRUMLISH a Gavin FISHER. OpenStreetMap [online]. [cit. 2014-10-10]. Dostupné z: <http://www.irma-international.org/viewtitle/68811/>.

[37] BABOUD, Lionel, Martin ČADÍK a Elmar EISEMANN. Automatic Photo-to-terrain Alignment for the Annotation of Mountain Pictures [online]. [cit. 2014-10-28]. Dostupné z: <http://resources.mpi-inf.mpg.de/photo-to-terrain/photo-to-terrain.pdf>.

[38] HE, Haibo. Learning from Imbalanced Data [online]. 2009 [cit. 2014-11-28]. Dostupné z: <http://www.ele.uri.edu/faculty/he/PDFfiles/ImbalancedLearning.pdf>.

[39] PROVOST, Foster, Machine Learning from Imbalanced Data Sets 101 [online]. [cit. 2014-11-28]. Dostupné z: <http://pages.stern.nyu.edu/~fprovost/Papers/skew.PDF>.

[40] Technical Standardization Committee on AV & IT Storage Systems and Equipment. Exchangeable image file format for digital still cameras: Exif Version 2.2 [online]. [cit. 2015-01-23]. Dostupné z: <http://www.exiv2.org/Exif2-2.PDF>.

[41] SOUKUP, Roman. Škola digitální fotografie. Grada, 2005. ISBN 80-247-1077-3.

[42] PIHAN, Roman. Fotografie a fototechniky [online]. [cit. 2014-01-24]. Dostupné z: <http://fotoroman.cz/techniques3/expozice1_zaklad.htm>.

[43] TAIGMAN, Yaniv, Ming, YANG a Lior WOLF. DeepFace: Closing the Gap to Human-Level Performance in Face Verification [online]. [cit. 2015-03-15]. Dostupné z: <http://www.cs.toronto.edu/~ranzato/publications/taigman_cvpr14.pdf>.

[44] GIRSHICK, Ross, Jeff DONAHUE, Trevor DARRELL a Jitendra MALIK. Rich feature hierarchies for accurate object detection and semantic segmentation [online]. [cit. 2015-03-15]. Dostupné z: <http://www.cs.berkeley.edu/~rbg/papers/r-cnn-cvpr.pdf >.

[45] DONAHUE, Jeff, Yangqing JIA, Oriol VINYALS a Judy HOFFMAN. DeCAF: a Deep Convolutional Activation Feature for Generic Visual Recognition [online]. [cit. 2015-03-18]. Dostupné z: <http://www.micc.unifi.it/downloads/readingroup/decaf_slides.pdf>.

[46] AGRAWAL, Pulkit, Ross GIRSHICK a Jitendra MALIK. Analyzing the Performance of Multilayer Neural Networks for Object Recognition [online]. [cit. 2015-03-06]. Dostupné z: <http://arxiv.org/pdf/1407.1610.pdf>.

[47] YANGQING, Jia. Fine-tuning CaffeNet for Style Recognition on "Flickr Style" Data [online]. [cit. 2014-11-20]. Dostupné z: <http://caffe.berkeleyvision.org/gathered/examples/finetune_flickr_style.html>.

[48] LAWRENCE, Jeannette. Data Preparation for a Neural Network [online]. [cit. 2015-03-29]. Dostupné z: <http://www.sysc.pdx.edu/classes/NNDataPrep.pdf>.

[49] ČADÍK, Martin, Jan VAŠÍČEK, Michal HRADIŠ, Filip RADENOVIČ a Ondřej CHUM. Camera Elevation Estimation from a Single Landscape Photograph. [cit. 2015-05-15].

Seznam příloh

Příloha A. Seznam vybraných kategorií.
Příloha B. Seznam fotoaparátů v databázi.
Příloha C. Programová dokumentace.
Příloha D. DVD obsahující elektronickou verzi technické zprávy, programovou dokumentaci a zdrojové soubory ke všem skriptům.

Příloha A - Seznam vybraných kategorií

1. butte
2. chalet
3. crevasse
4. mountain
5. mountain snowy
6. pasture
7. sea cliff
8. ski resort
9. ski slope
10. sky
11. snowfield
12. valley
13. volcano
14. castle
15. wheat field
16. field
17. field wild
18. forest road
19. forest path
20. creek
21. river
22. pond
23. badlands sandbar
24. swamp
25. bayou
26. marsh
27. corn field
28. rock arch

Příloha B - Seznam fotoaparátů v databázi

1. Canon EOS 40D
2. NIKON D7000
3. Canon EOS 5D Mark II
4. DSLR-A290
5. NIKON D90
6. Canon EOS 7D
7. Canon DIGITAL IXUS 860 IS
8. Canon EOS 400D DIGITAL
9. NIKON D80
10. NIKON D300
11. Canon EOS 350D DIGITAL
12. Canon EOS 30D
13. Canon EOS 50D
14. Canon EOS 450D
15. Canon EOS 550D
16. NIKON D70
17. DSC-HX5V
18. Canon EOS 500D
19. NIKON D3X
20. Canon PowerShot G9
21. NIKON D700
22. NIKON D200
23. Canon EOS 60D
24. NIKON D800
25. DSC-V1
26. DMC-TZ5
27. SLT-A55V
28. NIKON D60
29. NIKON D300S
30. NIKON D5000
31. NIKON D3100
32. DSC-H5
33. Canon EOS 600D
34. NIKON D40
35. Canon DIGITAL IXUS 970 IS
36. Canon EOS 1100D
37. NIKON D5100
38. Canon DIGITAL IXUS 870 IS
39. Canon PowerShot D10
40. IPHONE 4
41. IPHONE 4S
42. IPHONE 5
43. DMC-FZ38
44. SLT-A77V
45. NIKON D3000
46. COOLPIX L5
47. DSLR-A700
48. Canon EOS 5D Mark III

49. Canon PowerShot G11
50. DSC-R1
51. DMC-TZ10
52. CANON EOS 10D
53. CANON EOS 20D
54. CANON EOS 30D
55. CANON EOS 40D
56. CANON EOS 50D
57. CANON EOS 60D
58. CANON EOS 300D
59. CANON EOS 350D
60. CANON EOS 400D
61. CANON EOS 450D
62. CANON EOS 500D
63. CANON EOS 500D DIGITAL
64. CANON EOS 550D
65. CANON EOS 550D DIGITAL
66. CANON EOS 600D
67. CANON EOS 600D DIGITAL
68. CANON EOS 1000D
69. CANON EOS 1000D DIGITAL
70. CANON EOS 1100D
71. CANON EOS 1100D DIGITAL
72. CANON EOS 5D
73. CANON EOS 5D MARK II
74. CANON EOS 7D
75. CANON EOS 1DS MARK II
76. CANON EOS 1D MARK III
77. CANON EOS 1DS MARK III
78. FILM 35MM
79. FILM 645
80. FILM 6X7
81. OLYMPUS E-5
82. CYBERSHOT
83. CANON EOS 6D
84. DMC-TZ20
85. CANON POWERSHOT A640
86. DSLR-A200
87. PENTAX K-3
88. COOLPIX P90
89. DMC-LX3
90. NIKON D50
91. DSLR-A900
92. DYNAX 7D
93. DMC-TZ7
94. M9 DIGITAL CAMERA
95. DMC-TZ3
96. CANON POWERSHOT S100
97. C750UZ
98. CANON POWERSHOT S95
99. NIKON D40X
100. NIKON D70S

Příloha C - Programová dokumentace

Materiály na přiloženém DVD jsou rozděleny na 4 části:

1. Zdrojové kódy skriptů použité pro trénování konvolučních sítí.
2. Zdrojové kódy webové aplikace.
3. Zdrojové kódy MATLAB skriptů, které slouží pro vyhodnocení všech experimentů.
4. Seznamy vytvořených verzí datasetu.

Upozornění:

DVD obsahuje vybrané skripty a k nim ukázková zdrojová data. Kvůli velikosti vytvořených dat nebylo možné vše přiložit na DVD. Kompletní data jsou k dispozici zde:

- /mnt/matylda1/xvasic21/DATASET - dataset pro trénování konvolučních sítí,
- /mnt/matylda1/xvasic21/DATASET_ORIG_EXIF - dataset s obrazy v původním rozlišení a přidanými EXIF daty.

Konvoluční sítě trénované na SGE jsou k dispozici ve složce:

- /mnt/matylda1/xvasic21/CAFFE/caffe/examples

Potřebný software:

Ke spuštění vytvořených skriptů je potřeba tento software:

- Framework Caffe (http://caffe.berkeleyvision.org/)
- Python 2.7 + použité knihovny
- Exiftool (http://www.sno.phy.queensu.ca/~phil/exiftool/)
- Gengrid (k vyžádání u: http://www.fit.vutbr.cz/~cadik/)
- MATLAB

C.1 Konvoluční sítě

Tato kapitola představuje uživatelskou příručku ke skriptům tvořících jádro provedených experimentů.

C.1.1 Získání nadmořských výšek k obrazům z kolekce fotografií

*(Zdrojové kódy jsou umístěny ve složce: **/source_code/altitude_extraction**)*

Nejdříve bylo nutné extrahovat GPS souřadnice z fotografií v kolekci. K tomu slouží skript ***/Matlab/get_GPS.m***. Skript postupně prochází seznam fotografií a z nich čte GPS souřadnice.

Pro spuštění skriptu na SGE jsem vytvořil skript ***/Matlab/SGE_GPS_script.sh***. Výstup obsahuje cesty k obrazům a jejich GPS souřadnice. Tyto údaje reprezentují vstup skriptu ***/Gengrid/run_gengrid_SGE.sh***, který volá nástroj Gengrid pro získaní nadmořských výšek. Dále rozděluje obrazy na trénovací a testovací sadu pomocí funkce modulo. Pro spuštění skriptu na SGE je vytvořen skript ***/Gengrid/Gengrid_SGE.sh***.

Ve složce ***/GPS_Visualizer*** je umístěn výstup nástroje GPS Visualizer[12] a skript ***parse_file.py*** pro formátování výstupních dat do podoby vhodné pro CAFFE framework.

C.1.2 Filtrace kolekce fotografií - HTML

*(Zdrojové kódy jsou umístěny ve složce: **/source_code/filtration_HTML**)*

Stažená kolekce fotografií obsahovala nesouvisející snímky, které bylo potřeba filtrovat. K tomuto účelu jsem vytvořil python skript využívající framework CAFFE a natrénovaný model Places-CNN[13]. Nejdříve, ale bylo nutné manuálně vybrat související kategorie venkovních scén. K tomu složí skript ***classification_HTML.py***.

Funkce ***processFile(currentDir, net, filename)*** postupně prochází všechny soubory v uživatelem zadané složce a pokud se jedná o fotografii s příponou jpg, tak se u ní predikuje typ scény.

Výstup sítě je distribuce nad 205 třídami. Skript generuje HTML soubor, který obsahuje fotografii a 3 kategorie s největší pravděpodobností. Dále je uvedeno skóre pro vybrané kategorie. Ukázka výstupu v souboru: ***classification_ahornberg.html*.**

C.1.3 Filtrace kolekce fotografií - dataset

*(Zdrojové kódy jsou umístěny ve složce: **/source_code/filtration**)*

Pro vytvoření datasetu k trénování konvoluční sítě slouží skript ***classification_final.py***. Jedná se o upravený skript z předchozí kapitoly, který místo generování HTML souboru vhodné obrazy zmenší a zkopíruje do cílové složky. Pro spouštění skriptu na SGE je vytvořen skript ***SGE_script.sh***. Konfigurační soubory k modelu sítě se nachází ve složce ***/Settings***.

C.1.4 Trénování sítě - Fine-tunning

*(Zdrojové kódy jsou umístěny ve složce: **/source_code/altitudes_france_reg**)*

[12] www.gpsvisualizer.com
[13] https://github.com/BVLC/caffe/wiki/Model-Zoo

Ke spuštění trénování sítě Adaptované příznaky Places na SGE slouží skript ***train_altitudes_SGE.sh***. Soubor ***/Settings/solver.prototxt*** obsahuje konfiguraci trénování sítě. Soubor ***/Settings/train_val.prototxt*** konfiguruje architekturu trénované sítě:

```
...
layers {
        name: "conv1"
         type: CONVOLUTION
        bottom: "data_altitudes"
        top: "conv1"
        blobs_lr: 1
        blobs_lr: 2
        weight_decay: 1
        weight_decay: 0
        convolution_param {
                num_output: 96
                kernel_size: 11
                stride: 4
                weight_filler {
                        type: "gaussian"
                        std: 0.01
                  }
                 bias_filler {
                         type: "constant"
                        value: 0
                } }}
...
```

Složka ***/Settings*** dále obsahuje natrénovaný model places-CNN, který je určen k inicializaci vah na začátku trénování sítě.

C.1.5 Existující příznaky

*(Zdrojové kódy jsou umístěny ve složce: **/source_code/standard_features**)*

Skripty ve složce slouží k otestování existujících příznaků na úloze odhadu nadmořské výšky z obrazu. Jedná se o příznaky, které se při klasifikaci scén často používaj:

1. Barevný histogram obrazu
2. GIST deskriptor

C.1.5.1 Barevný histogram obrazu

Skript ***/COLOR_HIST/colorHist.py*** generuje soubor obsahující ke každému obrazu barevný histogram 16^3. Tento vektor se následně použije k trénování 3-vrstvé neuronové sítě. K použití frameworku CAFFE na neobrazová data je nutné vytvořit LEVELDB/LMDB databáze. K tomu je určen skript ***/result/leveldb/leveldbtest.py***. Konfigurační soubory pro trénování sítě se nachází ve složce ***/net/settings/***. Trénování se spouští pomocí skriptu ***/net/create_net.sh***.

C.1.5.2 GIST deskriptor

GIST deskriptor ke každému obrazu je vytvořen pomocí skriptu */GIST_ALTITUDE/features/process_dir.m*. Skript využívá funkce implementované od autorů GIST deskriptoru[14]. Výstupem skriptu je soubor obsahující cestu k obrazu a jeho GIST deskriptor. Stejně jako u barevného histogramu je potřeba nejdříve vytvořit leveldb databázi pomocí skriptu */leveldb/leveldbtest.py*. Konfigurace trénování a jeho spuštění je obdobné jako v případě barevného histogramu.

C.1.6 Stáhnutí EXIF dat

(Zdrojové kódy jsou umístěny ve složce: /source_code/flickr_download_exif)

EXIF data byla stažena pomocí skriptu *write_exif.sh*. Ten je navržen tak, aby mohl běžet dlouhodobě a nevadil mu případný výpadek elektřiny či restartování počítače. Vstupem skriptu je seznam obrazů *photo_list.txt* umístěný ve složce */config/*. Skript *write_exif.sh* postupně čte cesty k obrazům, ze kterých extrahuje hash tag. Hash tag je použit během posílání požadavků na Flickr. K tomu slouží skript *download_exif.py* umístěný ve složce */lib*. Skript *download_exif.py* dotazuje Flickr pomocí knihovny flickrapi. Výsledek ve formě ElementTree parsuje a ukládá ve formě, která může být přímo použita jako vstup nástroje exiftool. Zápis do obrazu přímo z tohoto skriptu není implementován z důvodu chybějících python knihoven pro zápis EXIF metadat. K dispozici jsou pouze knihovny pro čtení EXIF metadat.

Skript *write_exif.sh* použije výstup skriptu *download_exif.py* jako vstup pro nástroj exiftool. Skript *write_exif.sh* ukládá index aktuálně zpracovávaného řádku ze souboru *photo_list.txt*, aby v případě neočekávána situace mohl po opětovném spuštění pokračovat od poslední pozice. Index je uložen v souboru */config/settings.txt*.

C.1.7 Automatické zpracování EXIF dat

(Zdrojové kódy jsou umístěny ve složce: /source_code/exif_processing)

Složka obsahuje skripty pro automatické zpracování EXIF dat a výpočet statistik. Ve složkách *exif_data* a *exif_data_final* jsou textové soubory, které obsahují stažená EXIF data (EXIF data byla také uložena do textového souboru).

Ve složkách */features/DATE*, */features/EV*, */FEATURES/FOV* jsou umístěny skripty pro výpočet statistik z textových souborů umístěných ve složkách *exif_data* a *exif_final*. Mezi počítané statistiky patří počet validních hodnot pro daný tag, min, max, most common, mean hodnoty.

Ve složce */features/EXIF_FRANCE* se nachází skript *create_exif_final.py*, který z textových souborů obsahujících stažená EXIF data vytváří EXIF vektory ke všem použitelným obrazům a vytváří seznam pro trénování sítě. K tomu používá skripty, které jsou umístěny ve složce */lib*. Výstup skriptu je uložen v textových souborech umístěných ve složce */result_final/test/* a *result_final/train/*.

C.1.8 Změny v Caffe frameworku

(Zdrojové kódy jsou umístěny ve složce: /source_code/caffe_changes)

Python rozhraní frameworku Caffe je implementováno pouze pro standardní vstup ve formě obrazů. Bylo potřeba vytvořit 2 nové funkce, které na vstup sítě dají v očekávaném formátu obecný vektor čísel a nebo vektor čísel současně s obrazem. K tomu jsou v souboru *classifier.py* vytvořeny 2 funkce:

[14] people.csail.mit.edu/torralba/code/spatialenvelope

- *predict_vector(self, features)* - pro vektor,
- *predict_2 (self, inputs, features, oversample)* - pro vektor a obraz.

C.1.9 Regresní modely

(Zdrojové kódy jsou umístěny ve složce: /source_code/regression_models)

Složka obsahuje skripty pro experimenty s použitím různých vstupních dat.

C.1.9.1 Places

Při experimentech jsou nejdříve extrahovány aktivace 1. plně propojené vrstvy sítě *Adaptované příznaky Places* pomocí skriptu */data_extraction/data_extraction.py*. Aktivace jsou uloženy do textových souborů ve složce */Places/data/activations_france*.

Framework Caffe je implementován tak, že třídy klasifikovaných dat mohou být pouze typu integer. Pokud chceme používat labely typu double, tak je nutné vytvořit leveldb/lmdb databázi, která tyto labely obsahuje. Ve výsledku to znamená, že kromě leveldb/lmdb databáze pro data je nutná leveldb/lmdb databáze pro labely dat. Obě databáze je nutné vytvořit pro trénovací i testovací sadu. Při vytváření databází je nutné data a labely indexovat stejným způsobem, aby byly při trénování použity správným způsobem. K vytvoření databáze dat slouží skript */Places/leveldb/create_data_leveldb.py*, který vytvoří leveldb databázi ve složce */Places/leveldb/result/data*. Pro labely je vytvořen skript /Places/leveldb/create_label_leveldb.py, který ukládá databázi do složky *Places/leveldb/result/labels*.

Ve složce */Places/net* jsou všechny potřebné soubory pro trénování sítě. Trénování je spuštěno skriptem */Places/net /create_net.sh* (umístění konfiguračních souborů je stejné jako bylo již výše popsáno).

C.1.9.2 Places+EXIF

Experiment je implementován podobným způsobem jako síť *Places*. Liší se podobou vstupních dat, kdy je použit EXIF vektor. Ve složce */Places+EXIF/data/france/activations* jsou uloženy aktivace neuronů pro všechny obrazy. Ve složce */Places+EXIF/data/france/exif* jsou uloženy EXIF vektory pro všechny obrazy. Složka */Places+EXIF/data/france/exif/tools* obsahuje skript, který násobí EXIF vektory uložené v textovém souboru uživatelem definovanou hodnotou. Výstupem je textový soubor obsahující ke všem obrazům vynásobené EXIF vektory.

Ve složce */Places+EXIF/data/france/feature_vector/tools* je umístěn skript *join_data_final.py*, který čte textové soubory s aktivacemi a EXIF vektory. Obě části spojuje do jednoho vektoru, který následně uloží do textového souboru. Vytvoření leveldb databáze a trénování sítě je stejné jako v případě sítě *Places*.

C.1.9.3 PlacesCNN

Postup stejný jako v případě sítě *Places*. Rozdíl je ve vstupu, kterým jsou aktivace neuronů ze sítě Places-CNN a ne *Adaptované příznaky Places*.

C.1.10 Výstup regresních modelů

(Zdrojové kódy jsou umístěny ve složce: */source_code/regression_models_evaluation*)

Predikce nadmořské výšky pro všechny obrazy testovací sady je implementována pomocí skriptu *get_results.py*. Skript použije natrénované modely k predikci nadmořské výšky. Výstupem je textový soubor obsahující cestu k obrazu, reálnou nadmořskou výšku a predikovanou nadmořskou výšku. Tyto textové soubory jsou později použity v Matlab skriptech pro vyhodnocení experimentů.

C.2 Systém pro odhad nadmořské výšky z obrazu

(Zdrojové kódy jsou umístěny ve složce: /source_code/altitudes_system)

Implementovaný systém je navržen jako command-line aplikace, aby šel jednoduše použít jako součást jiných skriptů (např. geo-lokalizace, atd.). K aplikaci tedy nebylo vytvořeno grafické uživatelské rozhraní a ovládá se pomocí argumentů příkazové řádky. Systém byl implementován v jazyce Python a je volán pomocí BASH skriptu, který se stará o zpracování vstupních argumentů a management proměnných SHELL prostředí, které slouží k nastavení správného běhu frameworku Caffe.

Odhad nadmořské výšky z obrazu je prováděn pomocí 2 regresních modelů vytvořených během výše popsaných experimentů. Aplikace funguje ve 2 základních módech:

1. Odhad nadmořské výšky pouze z obrazu.
2. Odhad nadmořské výšky z obrazu při použití EXIF dat.

Při provádění odhadu nadmořské výšky pouze z obrazu je využíváno regresního modelu Adaptované příznaky Places, který byl natrénován na celém datasetu. Výstupem je číselná hodnota reprezentující odhad konvoluční sítě.

Druhý mód odhadu je založen na použití regresního modelu *Adaptované příznaky Places+EXIF*, který byl natrénován na EXIF subsetu. Aplikace v tomto módu nejdříve přečte EXIF data pomocí Python knihovny ExifRead 2.0.2 a zkontroluje, zda fotografie obsahuje všechny potřebná data. Pokud EXIF data fotografie nejsou kompletní, tak aplikace vrátí chybu a odhad musí být proveden pouze na základě obrazové informace. Omezením druhého módu je velikost vytvořené databáze modelů fotoaparátů a velikostí jejich senzorů, které se používají pro výpočet vertikálního a horizontálního zorného pole. U fotografií obsahujících všechna potřebná EXIF data a jež byly pořízeny fotoaparátem, který se nachází v databázi je vytvořen EXIF vektor jehož velikost je 127. EXIF vektor je generován pomocí knihoven vytvořených při implementaci regresního modelu *Adaptované příznaky Places+EXIF*. Výsledný EXIF vektor je vynásoben konstantou 40 aby měla mírně vyšší prioritu než aktivace sítě *Adaptované příznaky Places*. Následně je použit regresní model *Adaptované příznaky Places*, ze kterého jsou extrahovány aktivace 1. plně propojené vrstvy. EXIF vektor a aktivace ze sítě adaptované příznaky Places jsou spojeny ve vstupní vektor, jehož velikost je 4223 (4096 + 127). Vstupní vektor je nakonec poslán na vstup regresního modelu Adaptované příznaky Places+EXIF, jehož výstupem je odhad nadmořské výšky.

Argumenty příkazové řádky:

- h - výpis help na standardní výstup,
- i - nastavení cesty ke vstupnímu obrazu,
- p - způsob predikce:
  - basic - predikce pouze z obrazové informace,
  - exif - predikce z obrazu a EXIF dat,
- o - nastavení výstupu:
  - 0 - pouze nadmořská výška,
  - 1 - Výpis dodatečných informací o obrazu (výpis EXIF dat),
- c - nastavení cesty ke složce, kde je nainstalován framework Caffe.

Výstup systému:
Číselná hodnota reprezentující predikovanou nadmořskou výšku. V případě nastavení úplného výpisu pomocí parametru p se v případě predikce i z EXIF dat vypisují na výstup použitá EXIF data.

C.3 Webová aplikace

*(Zdrojové kódy jsou umístěny ve složce: **/user_test_app/**)*

C.3.1 Testovací obrazy

Testovací obrazy byly vybrány z testovací sady. Nejprve byl spuštěn skript ***/get_random_photos/create_usertest_dataset_radnom.py***. Ten vybral z každé kategorie 100 obrazů pokud to bylo možné. Následně jsem manuálně vybral z každé kategorie jeden až dva kvalitní obrazy.

C.3.2 Uživatelský test

Úvodní stránka webové aplikace ***index.php*** obsahuje krátký formulář. Údaje zadané ve formuláři jsou kontrolovány pomocí regulárních výrazů. Stránka ***test.php*** je jádrem celé webové aplikace. Při načtení stránky se generuje pole čísel 1-50 v náhodném pořadí. Následně je načten obraz podle prvního indexu z pole. Hodnota nadmořské výšky se nastavuje pomocí slideru z knihovny jQuery. Po zmáčknutí tlačítka "Potvrdit volbu" je načten nový obraz a odpověď uživatele se uloží do pole odpovědí.

Po odhadu všech 50 testovacích obrazů je načtena stránka ***test_results.php***, která uloží odpovědi do databáze a přesměruje webový prohlížeč na stránku ***test_results_output.php***, která uživateli zobrazí výsledek testu a obsahuje druhou část formuláře "Připomínky k testu" (V případě odeslání připomínek se aktualizuje záznam v databázi pro aktuálního uživatele). Akce je rozdělena do dvou skriptů, aby se výsledky uživatele neukládaly do databáze vícekrát pokud by uživatel obnovil stránku. Údaje z formuláře a výsledky testu jsou předávány pomocí metod POST a GET.

Stránky ***check.php***, ***check_time.php*** a ***check_results.php*** implementují funkci "následného procházení odpovědí". Uživatelé testu si mohou po ukončení testu znovu projít své odpovědi a porovnat je s reálnými nadmořskými výškami. To probíhá ve 3 fázích:

1. Uživatel nastaví pohlaví a věk, které zadal při vyplňování testu.
2. Následně vybere čas, kdy test prováděl (zobrazí se mu seznam časů, které vyhovují parametrům "věk" a "pohlaví").
3. V poslední fázi se uživateli zobrazí tabulka s obrazy jejich odhady a reálnými nadmořskými výškami.

C.4 Datové sady

Popis souborů ve složce ***/datasets/FINAL***.

- Složka ***EXIF*** - obsahuje seznamy "EXIF subsetu".
    - Složka ***EXIF/altitudes*** obsahuje trénovací/testovací sady s přesnou nadmořskou výškou.
    - Složka ***EXIF/categories*** obsahuje trénovací/testovací sady s kategoriemi nadmořských výšek.
    - Složka ***EXIF/data*** obsahuje soubory s EXIF vektorem pro daný obraz.
- Složka ***FULL*** - obsahuje seznamy "celého datasetu".
    - Složka ***FULL/altitudes*** obsahuje trénovací/testovací sady s přesnou nadmořskou výškou .
    - Složka ***FULL/categories*** obsahuje trénovací/testovací sady s kategoriemi nadmořských výšek.
- Složka ***USER_TEST*** obsahuje kolekci fotografií použitých při uživatelském testu. Jsou zde dále informace o nadmořských výškách, kategoriích a cesty k obrazům.
- Složka ***GPS*** obsahuje seznam GPS souřadnicí k celému datasetu Alps100k. Dále je zde skript pro výpočet statistik, které byly počítány pro vytvoření heatmapy pokrytí.
- Složka ***CORRECT_GPS*** obsahuje seznamy fotografií, které měly GPS souřadnice uloženy v EXIF datech. Ve složce jsou umístěny i skripty pro vytvoření seznamů.

C.5 Vyhodnocení a grafy

*(Zdrojové kódy jsou umístěny ve složce: **/vyhodnoceni**)*

Složka obsahuje Matlab skripty pro vyhodnocení prováděných experimentů (RMSE,MSE atd.) a vytvoření grafů v diplomové práci:

- ***Alps100k_histograms*** - histogramy vytvořených datasetů,
- ***photos_filtration*** - statistiky filtrace původní kolekce fotografií,
- ***random_vs_finetunning*** - porovnání adaptace vah a náhodné inicializace vah,
- ***regression_models_error_histogram*** - histogram chyb v různých nadmořských hladinách pro natrénované regresní modely,
- ***regression_models_stats*** - statistiky regresních modelů a tvorba kumulativního grafu,
- ***regression_vs_classification*** - porovnání klasifikačního a regresního modelu,
- ***user_test_ANOVAN*** - vyhodnocení uživatelského testu pomocí funkce anovan,
- ***user_test_results*** - vyhodnocení uživatelského testu (RMSE CNN, člověk), tvorba Boxplotu.