

Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra kybernetiky

DIPLOMOVÁ PRÁCE

PLZEŇ, 2021/2022

Adam Klečka

Anotace

Zatím nic

Annotation

Zatím nic

Poděkování

XXX

Obsah

1 Úvod do problematiky	5
2 ResNet, řešení pro hluboké sítě	6
3 I3D	8
3.1 I3D ResNet50	8
3.2 Klíčové snímky	10
3.3 Trénování sítě	10
4 Dataset	13
4.1 HAA500	13
4.1.1 Zpracování RGB datasetu	14
4.1.2 Zpracování segmentačního datasetu	14
4.1.3 Zpracování optického toku	16
4.2 Kinetics-400	17
5 Augmentace	20
6 Použitá metrika	21
7 Experimenty s I3D ResNet50	22
7.1 Implementace modelu I3D ResNet50	22
7.2 Test s malým subsetem <i>HAA500</i>	22
7.3 Trénování na celém datasetu	23
7.4 Normalizace dat	28
7.5 Spojení třech modelů	29
7.6 Klasifikátory	31
8 Závěr	32
9 Seznam literatury	33
10 Seznam obrázků	35
11 Seznam tabulek	36

1 Úvod do problematiky

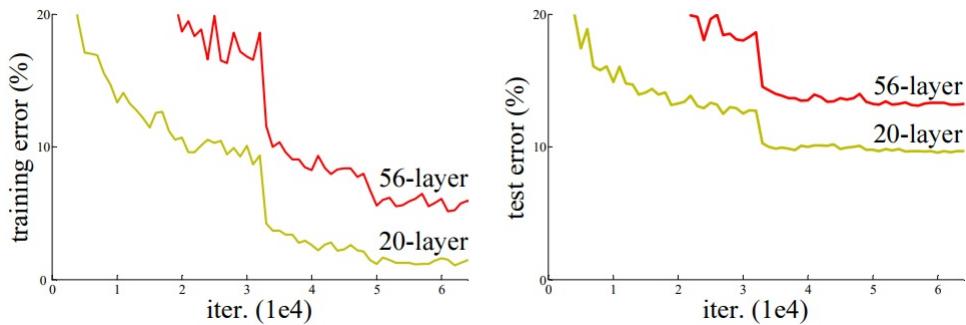
Lidé snadno rozpoznávají činnosti, které jsou nahrané na videu. Tato schopnost je pro člověka přirozená. V průběhu sledování videa pozorujeme objekty a jejich pohyb a díky tomu dokážeme díky zkušenostem vyhodnotit co se na videozáznamu děje. Počítačová automatizace tohoto procesu je náročná, ale je také zajímavá. Uplatnění může mít například pro automatické labelování krátkých videí na sociálních sítí, vyhledávání videa na základě obsahu, shrnutí děje videa nebo třeba monitorování chování starších osob. U sledování osob můžeme rozpoznávat denní činnosti jako je chůze, běh nebo pád. Při automatickém labelování videí můžeme rozpoznávat různé sporty nebo libovolné volnočasové aktivity.

Při rozpoznávání činností z videa dosahují velmi dobrých výsledků metody založené na principu konvolučních neuronových sítí. Budeme se zabývat metodou *I3D* [12], která takzvaně nafoukne standartní architektury *CNN* (*Convolutional neural network*) pro klasifikování obrázků jako jsou VGG-16 [14], Inception [15] nebo architekturu ResNet [10], která bude pro tuto práci klícová.

2 ResNet, řešení pro hluboké sítě

ResNet (Residual Network) [10] je konvoluční neuronová síť. Základ konvolučních sítí představují konvoluční filtry, které jsou ve skrytých vrstvách aplikovány na vstupní data. Vstupní data v případě počítačového vidění je nejčastější RGB obrázek. Podle úlohy mají síť jiné výstupy. Častá úloha na kterou je ResNet používán je klasifikace obrázků.

ResNet architektura řeší problém velmi hlubokých neuronových sítí. Zjistilo se, že pokud navýšíme hloubku neuronové sítě, můžeme tím dosáhnout lepších výsledků a menších hodnot loss funkce při trénování. Optimalizace ale takové hluboké neuronové sítě je obtížná. Ukázalo se, že hluboké sítě například s počtem 56 vrstev, dosahovaly při trénování horších výsledků než ta samá architektura s pouze 20 vrstvami. To je ukázáno na Obrázku 1, kde je průběh trénování architektury CIFRA-10. Jak je vidět, neuronová síť s 20 vrstvami dosáhla lepších trénovacích i testovacích výsledků než síť s 56 vrstvami.

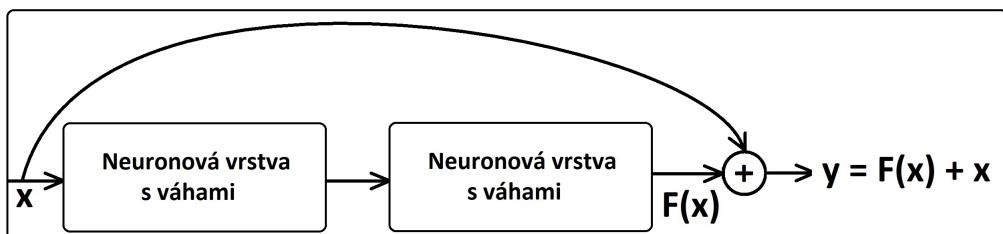


Obrázek 1

CIFRA-10 trénovací a testovací error

původní obrázek: <https://arxiv.org/pdf/1512.03385.pdf> (strana 1)

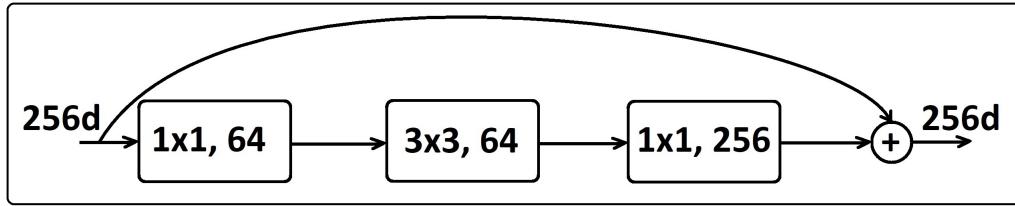
Důvodem může být špatná optimalizace sítě, ztrácející se gradient nebo špatné trénování po přidáním nových vrstev do natrénované sítě. ResNet přišel s řešením a to přidáním residuálních bloků (viz. Obrázek 2).



Obrázek 2
Residuální blok

Residuální blok obsahuje takzvaný *skip connection*, ten přičte vstupní hodnoty neuronových vrstev sítě k výstupním hodnotám. Tyto *skip connections* učí neuronové vrstvy tak, aby výstup y z nich byl roven rozdílu vstupu x . Z obrázku 2 je výstup značen $y = F(x) + x$, my ale požadujeme, aby výstup byl $y = F(x)$, proto je síť nucená učit se residuum. Sítím které využívají tuto techniku se říká ResNet.

Pro hlubší varianty ResNetu byly navrženy takzvané *bottleneck* bloky. Ty mají za účel navýšit rychlosť výpočtu s využitím snížením dimenze vstupu. Pokud máme například 256 dimenzionální vstup, můžeme ho promítnout do 64d a na konci bloku zpět rozšířit na 256 dimenzí (viz Obrázek 3).



Obrázek 3
Bottleneck blok

Díky ResNet architektuře jsme schopni vytvářet opravdu hluboké neuronové sítě. Velmi populární jsou ResNety s hloubkou 50,101 a 152. Tato práce se zaměří hlavně ResNet s hloubkou 50, zkráceně už jen ResNet50.

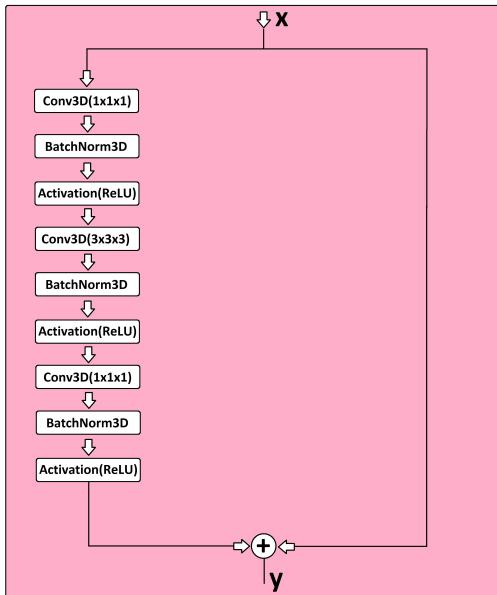
3 I3D

I3D (*Inflated 3D ConvNets*) [12] je způsob, jak z videí sbírat časo-prostorové informace. V úvodu už byla zmínka o takzvaném nafukování standartních konvolučních neuronových sítí pro klasifikaci 2D obrazu. Princip nafouknutí je takový, že vezmeme například funkční a léty prověřenou architekturu ResNet50 [10] a převedeme sdružovací filtry a konvoluční jádra z doby 2D ($N \times N$) na 3D ($N \times N \times N$). To znamená, že pokud máme ve vrstvě konvoluční jádro o rozměrech (3x3), převedeme ho na tvar (3x3x3). Vstupem do nafouknuté sítě je pak posloupnost 2D obrázků videa. Můžeme tedy říci, že přidaný třetí rozměr reprezentuje tok času videa. 3D konvoluční jádra se pohybují napříč časem a prostorem získávají z videa důležité příznaky.

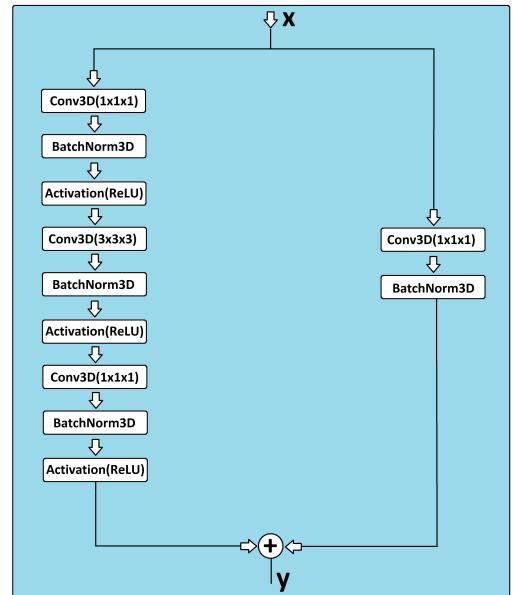
Tyto I3D modely dosahují kvalitních výsledků, pokud k nim implementujeme předtrénované váhy z ImageNet[7]. To lze docílit pomocí takzvaného *nudného* videa. Je to video složené pouze z jednoho snímku, které se opakuje a je časově neměnné. Trénování 3D sítě podle *nudného* videa se sítí učí klasifikovat obrázky. Díky linearitě se ukázalo, že váhy 2D filtrů jsou jen N násobky vah 3D filtrů. Na základě této znalosti můžeme vzít stejné váhy 2D filtrů a N krát je nakopírovat do 3D filtru a vydělit počtem N .

3.1 I3D ResNet50

Architektura [11] ResNet50 byla nafouknutá a nyní se z ní stala síť, která operuje s časově-prostorovými závislosti ze snímku. I3D ResNet50 obsahuje celkem 48 konvolučních vrstev, jeden *max-pool* a jeden *average-pool*. Síť I3D ResNet je složená ze x bloků. První blok nazveme například blok A (Obrázek 4).



Obrázek 4
Blok A



Obrázek 5
Blok B

V levé větvi bloku A (Obrázek 4) jsou na vstup x použité konvoluční jádro o rozměru (1x1x1) s hodnotou *stride* = (1x1x1), hodnota *stride* představuje krok po kterém se konvoluční jádro posouvá po vstupních datech. Tato konvoluční vrstva má za úkol snížit dimenzi vstupu. Jedná se tedy o *bottleneck* blok. Další vrstva v bloku se nazývá *BatchNorm3D* a ta

provádí normalizaci dat, které napomáhá ke stabilizaci sítě během trénovacího procesu. Další vrstva v pořadí je aktivační funkce, ta je zde použitá *ReLU* (*Rectified Linear Unit*) funkce. Má za úkol rozhodnout o výstupu neuronů způsobem (vzoreček(1)):

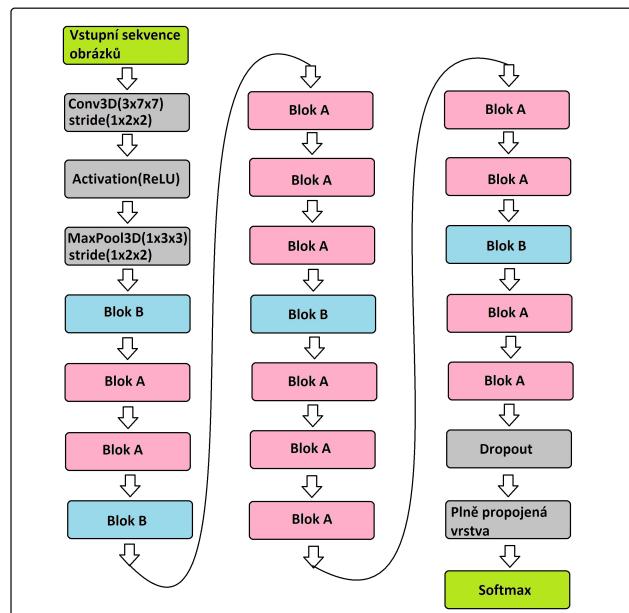
$$f(x) = \begin{cases} 0 & \text{jestli } x < 0 \\ x & \text{jestli } x \geq 0 \end{cases} \quad (1)$$

Aktivační funkce ReLU vrací nulovou hodnotu, při jakémkoliv záporném vstupu.

Tyto tři vrstvy se v levé větvi Bloku A opakují ještě 2x, s tím že podruhé máme konvoluční jádro o rozměrech (3x3x3) a na závěr konvoluční jádro (1x1x1), které má v *bottleneck* bloku za úkol zvýšit zpět svoji dimenzi. V pravé větvi Bloku A není žádná konvoluční vrstva, její účel je pouze při součtu poskytnout vstupní data a na konci bloku je sečít s výstupním signálem zpracované z levé větve.

I3D ResNet50 je tvořen ještě dalším blokem, který si pojmenujme například Blok B, ten je znázorněný na Obrázku 5. Je skoro totožný jako Blok A s rozdílem, že v pravé části větve má konvoluční vrstvu. Díky tomu tento blok na výstupu navýšuje svoji dimenzi v porovnání s výstupními dimenzenami vstupních dat, zatímco Blok A má jak na výstupu tak i na vstupu stejnou dimenzi.

Celá architektura sítě I3D ResNet50 je znázorněná na Obrázku 6. Na začátku je na vstupní posloupnost obrázků aplikována vrstva s konvolučním jádrem o rozloze (3x7x7) s krokovou hodnotou *stride* = (1x2x2). Následuje *ReLU* aktivační funkce, maximální sdružení a sekvence několika A a B bloků. Na závěr je na data aplikován dropout a plně propojená lineární vrstva, která převede data na vektor o velikosti rozpoznávaných činností. Tento vektor určuje pravděpodobnosti všech tříd. Softmax z vektoru vybere třídu s největší pravděpodobností a tu klasifikuje jako rozpoznanou činnost.



Obrázek 6
Blokově znázorněná architektura I3D ResNet 50

3.2 Klíčové snímky

Videa se skládají z několika snímků. Do sítě I3D ResNet50 ale požadujeme fixní vstupní počet klíčových snímků. Nejednodušší případ by byl, kdyby byla všechna videa stejně dlouhá. V takovém případě bychom mohli fixně vybírat klíčové snímky, které by vstupovaly do neuronové sítě. S takovým ideálním případem počítat nemůžeme a musíme volit strategii, při které se budou vhodně vybírat obrázky z videa tak, aby v nich byla zaznamenán průběh celého videa.

Vstupní počet klíčových snímků budeme volit 32. Jako strategii výběru 32 klíčových snímků byla vybrána metoda poměrně jednoduchá a intuitivní. Vždy chceme pokrýt celé video klíčovými snímky. Proto nejprve proběhne spočítání indexu kroků γ (vzoreček(2)):

$$\gamma = \frac{H}{K}, \quad (2)$$

kde γ reprezentuje index kroku, H je počet snímků videa a K je počet klíčových snímků. Pokud získáme hodnotu $\gamma < 1$, indikuje nám to, že máme méně dat než je požadovaný počet klíčovým snímků. V takovém případě se snímky rovnoměrně nakopírují za sebe. Vždy samozřejmě nemůžeme provést rovnoměrné kopírování, proto se v takovém případě vyberou náhodné snímky, které budou nakopírovány víckrát, či méněkrát. Tento případ bude demonstrován na příkladě, kde vektor $Index_{ks}$ reprezentuje vybrané klíčové snímky:

$$\begin{aligned} H &= 10, \\ K &= 32, \\ Index_{ks} &= [1, 1, 1, 2, 2, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 6, \\ &\quad 6, 6, 6, 7, 7, 8, 8, 8, 8, 9, 9, 9, 9, 10, 10, 10]. \end{aligned}$$

Za podmínky, že se bude hodnota γ rovna 1, tak je výběr klíčový snímků velmi jednoduchý, celé video se poskládá z klíčových snímků. Pokud ale bude $\gamma > 1$ a zároveň $\gamma < 2$, tak proběhne výběr klíčových snímků náhodně. Opět to bude demonstrováno na příkladě:

$$\begin{aligned} H &= 50, \\ K &= 32, \\ Index_{ks} &= [3, 5, 6, 8, 9, 11, 12, 14, 15, 16, 17, 18, 19, 21, 23, \\ &\quad 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 39, 42, 43, 44, 46, 49]. \end{aligned}$$

Na závěr zbývá situace, kdy $\gamma \geq 2$, nyní se rozdělí obrázky do skupin se stejným krokem. Velikost kroku je rovna γ zaokrouhleno dolů. Následně se z těchto vybraných skupin vybere jedna, která bude představovat klíčové snímky videa:

$$\begin{aligned} H &= 100, \\ K &= 32, \\ Index_{ks} &= [3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48, \\ &\quad 51, 54, 57, 60, 63, 66, 69, 72, 75, 78, 81, 84, 87, 90, 93, 96]. \end{aligned}$$

3.3 Trénování sítě

Trénování neuronových [2] sítí je proces, při kterém jsou sítí předkládána anotovaná data, síť je zpracuje a na základě výsledku upravuje váhy neuronů. Můžeme začínat s náhodně

inicializovanými hodnotami, nebo vycházet z nějakého předtrénovaného modelu na jiném datasetu. Pokud využijeme předtrénované váhy jiného modelu, očekává se rychlejší doba trénování. Velmi důležitá při trénování je takzvaná *Loss* funkce. Ta nám během trénování udává hodnotu, jak kvalitně je naše neuronová síť nastavená na rozpoznávání činností z videí. Velmi známá *Loss* funkce je *Cross-Entropy*, se kterou budeme provádět trénování I3D ResNet50. *Cross-Entropy* můžeme zapsat matematicky jako:

$$L(p, q) = - \sum_x^N p(x) \cdot \log(q(x)), \quad (3)$$

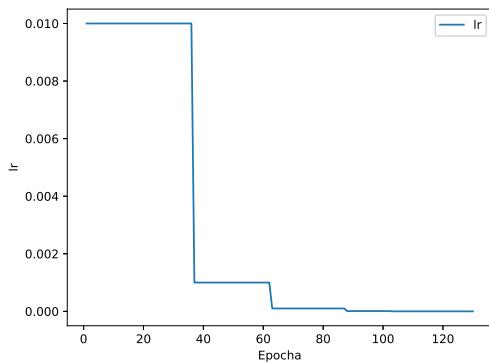
kde x je index třídy, N celkový počet tříd, $p()$ je binární indikátor správné klasifikace a funkce $q()$ reprezentuje výstup softmaxu. Čím přesněji bude síť správně klasifikovat trénovací videa, tím se bude hodnota *Loss* funkce snižovat. Cílem trénování je tedy co nejvíce snížit hodnotu této funkce.

Celý trénovací proces se dá pojmenovat jako úloha pro minimalizování *Loss* funkce. Je zde několik metod, které se snaží tuto úlohu řešit, říká se jim optimizéry. Jedním z velmi oblíbených algoritmů je takzvaný SGD (*Stochastic Gradient Descent*), matematicky jej lze zapsat jako (Vzoreček 3) :

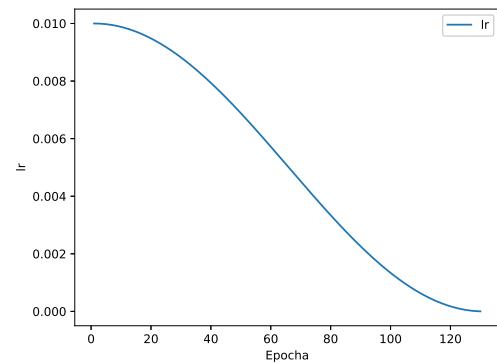
$$w_j = w_j - lr \frac{\partial L}{\partial w_j}, \quad (4)$$

kde w_j jsou váhy neuronové sítě, lr je hodnota učení a zlomek $\partial L / \partial w_j$ je takzvaný gradient. Gradient v matematice reprezentuje směr největšího růstu, v tomto případě má stejný význam, udává směr největšího růstu *Loss* funkce. My ale požadujeme směr nejmenšího růstu, abychom minimalizovali *Loss* funkci, stačí tedy vzít zápornou hodnotu gradientu. Váhy sítě se pak budou v tomto směru upravovat. Záporný gradient je ještě vážený hyper parametrem hodnoty učení lr , ten nabývá obvykle malých hodnot. Jediné co ještě potřebujeme k trénování je numericky vypočítat gradient $\partial L / \partial w_j$. Pro to lze použít metodu zvanou *Backpropagation*, kdy zpětně procházíme síť a získáváme gradienty vah mezi jednotlivými vrstvami.

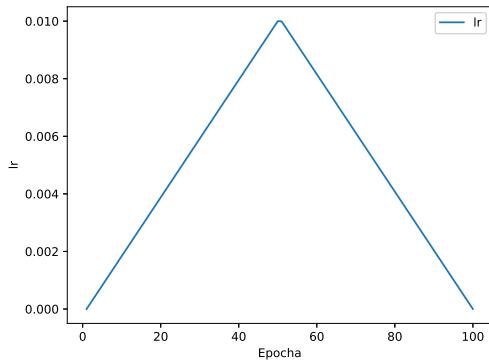
Abychom při trénování dokončovali ideálně ke globálnímu minimu *Loss* funkce, je důležité také zvolit vhodnou strategii úpravy konstanty učení lr . Těch je hned několik. Můžeme zvolit například běžné krokové snížení N epochách (Obrázek 7), kosínovou strategie (Obrázek 8) , cyklickou (Obrázek 9) nebo třeba restartovací strategii (Obrázek 10).



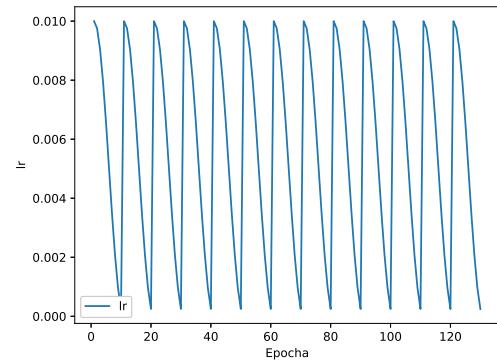
Obrázek 7
Kroková strategie lr



Obrázek 8
Kosínová strategie lr



Obrázek 9
Cyklická strategie lr (jeden cyklus)



Obrázek 10
Strategie restartováním lr

4 Dataset

Dataset je důležitá část trénování modelů neuronový sítí. Proto zvolení kvalitního datasetu, kde si budeme jistý správně anotovanými daty, je naprostý základ. Set data pro rozpoznávání činností musí obsahovat videa a informaci o jejich správné klasifikaci přiložené v externím souboru. Těmto informacím o správné klasifikace se říká *anotace*. Celý dataset by pak měl být rozdělen do třech částí a to konkrétně na:

1. trénovací data,
2. validační data,
3. testovací data.

Trénovací data slouží k trénování modelu a obvykle tato množina bývá nejobjemnější. Pomocí validačních dat se během trénovacího procesu validuje natrénovaný model a my tak máme hrubou představu o tom, jak si model povede na neviděných datech. Poslední jsou testovací data. Ty slouží k otestování a následné vyhodnocení natrénovaného modelu.

4.1 HAA500

Human-Centric Atomic Action (HAA500) [6] je ručně anotovaná sada dat videí s lidskou činností. Videa v *HAA500* byla pečlivě vybírána tak, aby zachycoval nepřerušovaný pohyb lidských postav při různých činnostech. Jednotlivé třídy datasetu jsou hodně dopodrobna popsány, tak aby byly jedinečné a rozlišitelné od ostatních. Například činnost hraní fotbalu je rozčleněna na fotbalová střela, fotbalová příhrávka, fotbalové vhazování, dribbling s míčem při fotbale, fotbalová hlavička a fotbalový zákrok brankáře (viz. Obrázek 11, 12, 13). Videa byla vybírána také tak, aby osoba provozující klasifikovanou činnost byla dominantní osobou videa. Hlavní část videa je zaměřena na člověka a u videí je průměrně 69.7% detekovatelných kloubů hlavní osoby. *HAA500* poskytne pro tuto práci data, které využijeme k trénování a testování modelů I3D ResNet50.



Obrázek 11
HAA500 fotbalový dribbling



Obrázek 12
HAA500 fotbalová hlavička



Obrázek 13
HAA500 fotbalový zákrok brankáře

Celková sada dat *HAA500* obsahuje 500 tříd. V každé třídě je celkem 20 HD videí (720x1280). Dohromady dataset poskytuje 10000 videí. Videia jsou tedy perfektně distribuována mezi jednotlivými třídami. Data do trénovací, validační a testovací množiny byla rozdělena v poměru 16:1:3 (viz Tabulka 1).

	Počet videí
Trénovací data	8000
Validační data	500
Testovací data	1500
Celkový počet dat	10000

Tabulka 1
Počet dat

4.1.1 Zpracování RGB datasetu

Videa poskytnuté datasetem *HAA500* byla uložená ve formátu *.mp4*. Jedná se o formát který obvykle obsahuje stopy videa a audia. Pro práci s neuronovou sítí bylo nutné převést videa z *.mp4* do posloupnosti *.jpg* obrázků a každou tuto posloupnost uložit do samostatné složky. Videia byla do trénovací, validační a testovací množiny rozdělena způsobem, že prvních 16 videí se přiřadilo k trénovací množině, 17. video k validační a zbytek (18,19,20) k testovací množině dat. Díky tomu jsme získali požadované rozdělení 8000 trénovacích, 500 validačních a 1500 testovacích dat. Na úrovni předzpracování se ještě obrázky zmenšily z velikosti (720x1280) na (224x224). Informace o názvu videa, správné klasifikaci, délce videa a jakým způsobem jsou pojmenované *.jpg* obrázky, byly uložené v textovém souboru. Celkem bylo tímto způsobem získáno 594300 oanotovaných obrázků.

4.1.2 Zpracování segmentačního datasetu

Segmentace [18] je metoda počítačového vidění, která rozděluje obraz do segmentů, které spolu nějakým způsobem souvisí. Segmenty můžou reprezentovat jednotlivé objekty na obrázku. Sémantická segmentace pak rozděluje celý obrázek do segmentů a tím se snižuje složitost celého obrazu. Všem pixelům patřící do jednoho segmentu je přiřazena společná značka, obvykle společná barva. Myšlenka za použitím segmentace je taková, že při rozpoznávání činností nám jde především o pohyb lidských osob, které pro nás představují hlavní objekty. S využitím sémantické segmentace můžeme odstranit rušivé pohybující se objekty, které jsou v pozadí a pro hlavní činnost jsou nepodstatné. Dále získáme pro všechny třídy stejné pozadí a tím se odstraní problém s nestacionární kamerou, takže se učení bude moci zaměřit především na pohyb hlavních objektů videa.

Způsobů a metod jak provést segmentaci je celá řada. My se ale zaměříme na použití sémantické segmentace s využitím konvolučních neuronových sítí, přesněji na architekturu *DeepLab v3* [5] s *backbone* sítí ResNet50. Tato architektura funguje na principu encoder a dekoder, kde encoder je zopovědný za získání *feature maps* z obrázku a dekoder používá převzorkování, aby získal zpět detaily objektů a jejich prostorové rozměry z nízkorozměrných *feature maps*. Budeme využívat předtrénované váhy z subsetu *COCO train2017* [13]. Tento předtrénovaný model obsahuje celkem 20 tříd, které je síť schopna segmentovat. Na datasetu *COCO val2017* [13] dosáhl tento model hodnoty *mean IoU* = 66.4 a *global pixelwise acc* = 92.4.

Síť *DeepLabv3-ResNet50* s předtrénovaným modelem byla aplikována na videa z datasetu *HAA500* a získali jsme sémanticky segmentované obrazy. Úplně stejným způsobem jako při zpracování RGB datasetu byl dataset připraven, tedy videa byla rozkouskována do posloupnosti obrázku ve tvaru *.jpg* a i stejným způsobem byla data rozdělena do trénovací, validační a testovací množiny. Ukázky sémanticky segmentovaných videí jsou vidět na Obrázcích 14 a 15. Je na nich vidět, že lidé na videích byli korektně segmentováni a označeni oranžovou barvou.



Obrázek 14
Ukázka segmentace - hod diskem



Obrázek 15
Ukázka segmentace - pád na kole

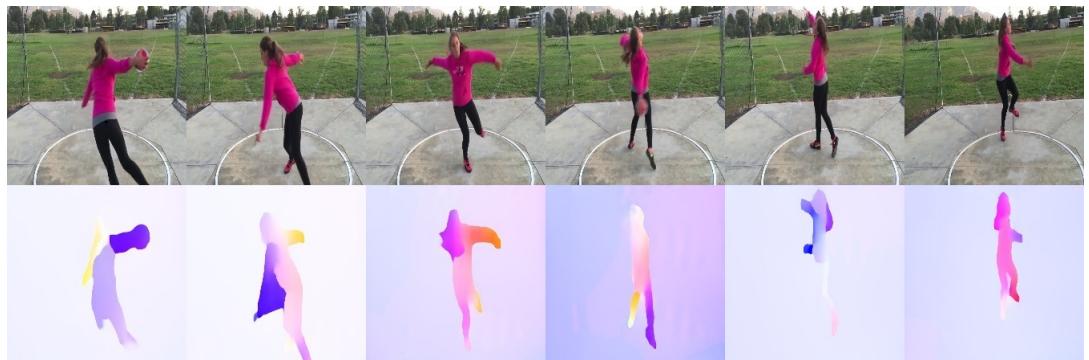
4.1.3 Zpracování optického toku

Optický tok [8] je technika používaná k popisu pohybu objektů z obrázků. Princip je založený na výpočtu rychlosti bodů v rámci několika snímků a predikováním kde by se body mohly na následujících snímcích nacházet. Metoda má uplatnění hlavně pro videa, protože snímky videa mají obvykle malý časový krok. Bude zajímavé pozorovat, jaké výsledky s I3D Resnet50 sítí dosáhneme s daty po aplikování optického toku, protože se na obrázcích zvýrazní pohybující se objekty, což by mohlo mít za důsledek zlepšení trénování a následné testování sítě.

Je několik přístupů jak řešit optický tok. My využijeme výpočet optického toku s pomocí neuronových sítí a to architektury *RAFT* [16] *Recurrent All-Pairs Field Transforms*. S využitím neuronové sítě, kde je vše založené na latentních príznakových mapách, je vše přesnější a počítané s větší efektivitou, než kdybychom počítali optický tok tradičními metodami. Architektura *RAFT* může být rozdělena na dvě části encoder a iterátor. První část encoder je podobná struktuře encoder-decoder, která získává latentní *feature maps* z obrázku. Iterátor je druh rekurentní neuronové sítě, která predikuje sekvence toku.

Předtrénované váhy pro model *RAFT* byly trénovány na datasetech *Chairs[1] + Things[17]* + *Sintel [3]* fine-tuning + *Kitti [9]* fine-tuning . Tento model dosáhl na testovacím setu hodnot *kitti_test_fl_all = 5.19*.

Na data z *HAA500* jsme aplikovali *RAFT* s předtrénovanými váhami. Do sítě vždy vstupovali 2 po sobě jdoucí snímků a výstupem byl predikovaný pohyb mezi dvěma snímkami. Výstupní obrázek byl interpretován paletou barev duhy, kde bílá symbolizuje žádný pohyb, červená malý a modrá velký pohyb. Ukázky optického toku jsou znázorněny na Obrázku 16 a 17. U obrázku 16 je vidět zabarvení pozadí. To je způsobené pohybující se kamerou. U Obrázku 17 je statická kamera a proto dobře vynikl pohyb máchnutí pálkou. Opět byla data stejně rozdělena a uložena stejným způsobem jako u RGB dat a segmentačních dat.



Obrázek 16
Ukázka segmentace - hod diskem



Obrázek 17
Ukázka segmentace - baseball odpal míčku

4.2 Kinetics-400

Další důležitý dataset je *Kinetics-400*. Při experimentech budou využity předtrénované váhy modelu I3D ResNet50 na tomto datasetu. Proto je důležité nahlédnout na videa z datasetu a zjistit, na jakých videích se model trénoval.

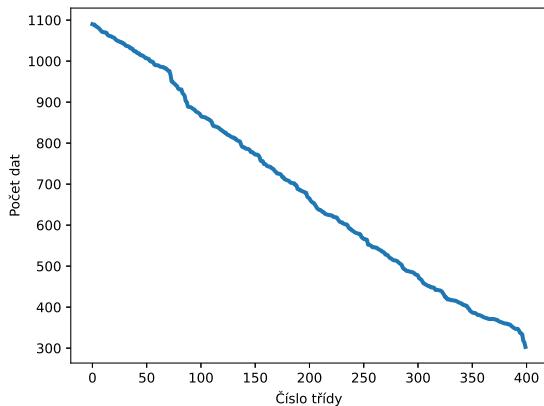
Kinetics 400 je kolekce videí s lidskou činností. Tyto videa jsou reprezentovány pomocí URL odkazu na videa internetového serveru *YouTube.com*. U každého URL odkazu je zaznamenaný počáteční a koncový čas označené činnosti, přičemž každé video trvá zhruba 10 vteřin. Činnosti jsou zaměřeny na člověka a pokrývají širokou škálu tříd včetně interakcí mezi člověkem a objektem, jako je hraní na hudební nástroje a také interakcí mezi lidmi, jako je podání ruky. V tomto datasetu je celkem 400 tříd. Každá třída obsahuje minimálně 400 videoklipů. Všechny tyto informace o URL odkazu, času ve kterém se vyskytuje činnost a korektně pojmenovaná činnost, jsou uloženy ve formátu *.json* nebo *.csv*.

Na základě dostupného datasetu *Kinetics 400* proběhla základní analýza dat. Celkový počet videí je zaznamenán v Tabulce 2. Dataset byl už rozdělen do třech množin dat, trénovací, validační a testovací.

	Počet videí
Trénovací data	219 782
Validační data	18 035
Testovací data	35 357
Celkový počet dat	273 174

Tabulka 2
Počet dat

Při analýze datasetu je důležité zjistit jak jsou data v jednotlivých třídách rozprostřeny. Distribuce dat mezi třídami je zobrazena na Obrázku 18.



Obrázek 18
Distribuce tříd

Z Obrázku 2 je patrné, že zde není žádná třída, která by výrazně v počtu dat převyšovala ostatní třídy. Pro každou třídu je také dostatečné množství videí. Pět nejčetnějších tříd je:

1. snowkitting: 1090 videí,
2. sfoukávání svíček: 1089 videí,
3. bobování: 1089 videí,
4. jízda na kánoi nebo kayaku: 1085 videí,
5. hra na harpu: 1084 videí.

Ukázky videí jsou na Obrázcích 19,20 a 21.



Obrázek 19
Kinetics-400 snowkitting



Obrázek 20
Kinetics-400 sfoukávání svíček



Obrázek 21
Kinetics-400 bobování

5 Augmentace

Pokud nejsou trénovací data různorodá, hrozí riziko, že při trénovacím procesu dojde k přetrénování na trénovací data. Je to situace, kdy se snižuje *Loss* funkce na trénovacích datech, ale zvyšuje se na validačních. Abychom tomuto zabránili, je třeba k trénovacímu algoritmu aplikovat augmentační přístupy. Tyto přístupy se snaží o změny vstupních dat v průběhu trénování sítě.

První použitá augmentace pro naše účely je založená na zvětšení videa, následném výběru oblasti a jejímu vyříznu do původního rozměru obrázku (224x224). S touto metodou je ale třeba dát pozor, protože dataset *HAA500* je významným tím, že jsou činnosti centrovány do středu videa. Proto je vhodné volit malý poměr zvětšení, aby bylo ve výsledku nevyřízly důležité pixely z videa. Bylo otestováno, že velké zvětšení oblasti například z videa (224 x 224) na (320 x 320) pixelů a následné náhodné vyříznutí oblasti zpět na původní rozměry docházelo k horším výsledkům, než kdyby nebyla použitá žádná augmentační metoda. Proto se v této práci se video zvětšuje z rozměrů $\langle 224 \times 224, 260 \times 260 \rangle$, vždy na čtverec a následuje náhodné oríznutí zpět na (224×224) . Dále byla přidána další augmentační metoda, která každou epochu upravuje data způsobem náhodného horizontálního otočení videa s pravděpodobností $p_{flip} = 0.5$. Statisticky bude tedy každé druhé video být otočeno.

Pomocí těchto metod jsou neuronové sítě předkládány každou epochu lehce odlišná data. Ukázka augmentace je zobrazeny na Obrázku 22, kde v horní části je originální video a ve spodní části je ukázka augmentovaného videa.



Obrázek 22
Ukázka augmentace (pád na kole)

6 Použitá metrika

Při porovnávání několika modelů je třeba zvolit vhodnou metriku, díky které můžeme určit kvalitu rozpoznávání činností. Velmi častou volenou metrikou pro rozpoznávání akcí z videí je takzvaná *Top1* přesnost, k ní jsou často přidávány *Top3* a *Top5* přesnosti. Všechny tyto metriky mají společnou jednotku [%]. *Top1* přesnost udává kolik činností neuronová síť dokázala správně rozpoznat. Matematicky lze zapsat jako:

$$Top1 = \frac{TP}{FP + TP} [\%], \quad (5)$$

kde TP jsou správně rozpoznaná videa a součet ve jmenovateli $FP + TP$ je počet všech testovaných videí. Za TP se berou třídy, který mají největší pravděpodobnost z výstupní lineární vrstvy sítě. Výstupní lineární vrstva má podobu N vektoru, kde N je počet tříd. Neuronová síť přiřazuje každé činnosti věrohodnost a my tak dokážeme získat informaci i o 2., 3., 4. a 5. nejvěrohodnější třídě podle natrénované sítě. Pak můžeme určit i *Top3* a *Top5* přesnost. Kde u *Top3* přesnosti se bude brát jako TP správně klasifikovaná třída na 1., 2. nebo 3. pozici. Pro úplné vyjasnění si výpočet metrik *Top1* a *Top3* ukážeme na příkladě.

Č.	Správná třída	1. místo	2. místo	3. místo	TP (<i>Top1</i>)	TP (<i>Top3</i>)
1)	lukostřelba	lukostřelba	střelba z pistole	hra na kytaru	+1	+1
2)	hod diskem	hod míčem	hod oštěpem	pojídání jablka	+0	+0
3)	lukostřelba	střelba ze zbraně	lukostřelba	hod oštěpem	+0	+1
4)	hod diskem	hod diskem	hod míčem	frisbee	+1	+1
5)	pojídání jablka	pojídání zmrzliny	pojídání pizzy	pojídání jablka	+0	+1
				Součet	2	4

Tabulka 3
Příklad testování úspěšnosti klasifikace videí

Na příkladu z Tabulky 3 bylo testováno celkem 5 videí. Síť dokázala rozpoznat 2 videa z pěti na prvním místě, takže z tohoto testování získala hodnotu $Top1 = 40\%$. Dále se správná 4 videa nacházely na prvních 3 pozicích a tedy $Top3 = 80\%$.

Podle článku **HAA500: Human-Centric Atomic Action Dataset with Curated Videos** [6] byl RGB set dat *HAA500* natrénován na model *I3D* a bylo získáno:

- $Top1 = 33.53\%$,
- $Top3 = 53.40\%$.

K těmto hodnotám bychom se s architekturou *I3D ResNet* chtěli co nejvíce přiblížit.

7 Experimenty s I3D ResNet50

7.1 Implementace modelu I3D ResNet50

Pro experimenty, které budou v následujících kapitolách provedený, byl implementován model I3D ResNet50 [4] do programovacího jazyka *Python*. Trénování sítě se bude provádět s *Cross Entropy Loss* funkcí a metodikou *backpropagation*. Všechny modely budou trénovány s *optimizerem SGD*. Cílem těchto experimentů je natrénovat nejlepší model ve smyslu největší hodnoty *Top1* na testovací množině dat a porovnat modely trénované na klasických RGB obrázcích, segmentvaných obrázcích a obrázcích po aplikaci optického toku.

7.2 Test s malým subsetem *HAA500*

První experiment byl proveden s malým subsetem RGB datasetu *HAA500*. Malý subset obsahoval 20 tříd se sportovními aktivitami. Celkový počet dat byl tedy viz. Tabulka 4.

	Počet videí
Trénovací data	320
Validační data	20
Testovací data	60
Celkový počet dat	400

Tabulka 4
Počet dat

Trénování proběhlo se sítí I3D ResNet50 pro 4,8,16 a 32 klíčových snímků videa s náhodně inicializovanými váhami modelu. Pro trénování byl použitý kosínový *scheduler* se základní hodnotou $lr = 0.01$ a modely se trénovaly 50 epoch. Hlavním cílem toho experimentu bylo především otestovat funkčnost trénování neuronové sítě. Tento postup je vhodný předtím než proběhne trénování pro všechna data, protože je možné takto odhalit drobné chyby programovacího kódu, které by mohly způsobovat nepřesnosti při trénovacím procesu. Při tomto experimentu s malým subsetem *HAA500* se také zkoumala přesnost klasifikace videí v závislosti na počtu zvolených klíčových snímků systému. Záznam experimentu je znázorněný v Tabulce 5.

	Top1 [%]	Top3 [%]	Top5 [%]
4 klíčové snímků			
8 klíčových snímků			
16 klíčových snímků			
32 klíčových snímků			

Tabulka 5
Závislost přesnosti rozpoznávání a počtu klíčových snímků

Z Tabulky 5 je patrné, že pro systém pracující s více klíčovými snímkami roste také přesnost klasifikace testovaných videí. Díky této znalosti bude v následujících experimentech pracováno se systémy s 32 klíčovými snímkami, jediná jejich nevýhoda je ale větší výpočetní složitost a tím také delší doba trénování a testování.

7.3 Trénování na celém datasetu

Následující trénovací experiment byl proveden se všemi daty z trénovací a validační množiny datasetu *HAA500*. Vycházeli jsem už z předtrénovaných vah *I3D-ResNet50* natrénovaných na datasetu *Kinetics-400* a byl proveden takzvaný *fine-tuning*. Při tomto procesu načteme stejné váhy modelu a pouze odstraníme váhy poslední plně propojené lineární vrstvy, která přiřadí všem třídám hodnoty věrohodnosti. Díky tomuto bude možné natrénovat model na 500 odlišných tříd z našeho datasetu.

Trénovací *optimizer* byl použitý *SGD* s různými *schedulery*. Ve většině případů se síť trénovala 130 epoch. Jako nejlepší model byl vyhodnocen ten, který při trénovacím procesu dosáhl největší *Top1[%]* přesnost na validačních datech. Následně byl tento nejlepší model vyhodnocen na testovací množině dat. Celý trénovací experiment za cílem dosažení co nejlepší hodnoty *Top1[%]* na testovacím setu je zaznamenaný v Tabulce 6.

Pokus	Optimizer Scheduler	lr	nastavení	Top1[%] (val)	Top1[%] (test)	Top3[%] (test)	Top5[%] (test)
1.0.	SGD step	0.01	step=20	52.0	51.067	71.533	78.600
1.1.	SGD step	0.001	step=20	73.6	70.000	86.933	90.400
1.2.	SGD step	0.0001	step=20	63.8	60.000	84.067	90.533
2.0.	SGD cosine	0.01	eta_min=0	56.2	56.000	74.333	80.266
2.1	SGD cosine	0.001	eta_min=0	75.0	72.200	86.400	90.267
2.3	SGD cosine	0.0005	eta_min=0	74.4	71.467	87.200	91.400
2.4	SGD cosine	0.0001	eta_min=0	73.0	69.867	86.667	91.800
3.0	SGD cyclic	10^{-8}	max_lr= 0.001 step_size_up = 15 mode = triangular	74.2	73.400	87.733	91.267
3.1	SGD cyclic	10^{-8}	max_lr= 0.001 step_size_up = 15 mode = triangular2	73.6	71.00	87.133	91.20
3.0	SGD cyclic	10^{-8}	max_lr= 0.001 step_size_up = 15 mode = exp_range	73.8	71.133	87.467	91.533
4.0	SGD plateau	0.01	mode=min	53.0	52.933	81.600	79.600
4.1	SGD plateau	0.001	mode=min	73.4	70.067	86.467	89.867
5.0	SGD restart	0.01	T_0 = 10, T_mult= 1, eta_min= 10^{-6}	52.2	51.4	70.866	77.666
5.1	SGD restart	0.001	T_0 = 20, T_mult= 1, eta_min= 10^{-8}	74.4	71.133	87.533	91.333
5.2	SGD restart	0.005	T_0 = 20, T_mult= 1, eta_min= 10^{-8}	75.4	72.400	87.600	91.200

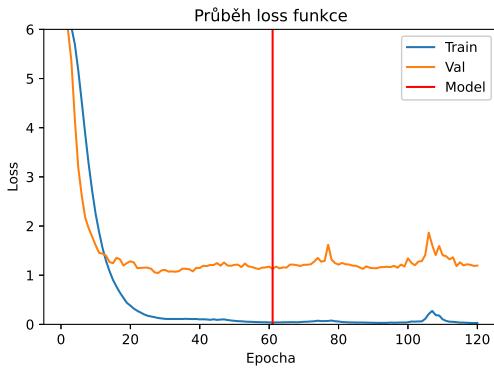
Tabulka 6
Trénování I3D ResNet50 na rgb datech

Pokus	Optimizer Scheduler	lr	nastavení	Top1 [%] (val)	Top1 [%] (test)	Top3 [%] (test)	Top5 [%] (test)
3.	SGD step	0.01	step= 20	29.8	28.000	44.933	52.733
1.1	SGD step	0.001	step= 20	41.2	42.267	57.667	65.000
1.2	SGD step	0.0001	step= 20	26.0	22.800	39.867	50.333
5.2.	SGD cosine	0.01	eta_min=0	37	34.733	51.067	58.000
5.1.	SGD cosine	0.005	eta_min=0	41.4	39.733	55.267	62.467
2.1.	SGD cosine	0.001	eta_min=0	43.6	44.133	60.933	67.867
2.2.	SGD cosine	0.0005	eta_min=0	41.6	42.667	58.400	65.000
2.3.	SGD cosine	0.0001	eta_min=0	36.6	33.733	50.667	60.533
3.0	SGD cyclic	10^{-8}	max_lr= 0.001 step_size_up = 15 mode = triangular	42.0	40.267	64.533	64.533
3.1	SGD cyclic	10^{-8}	max_lr= 0.001 step_size_up = 15 mode = triangular2	41.8	40.667	58.533	65.400
3.0	SGD cyclic	10^{-8}	max_lr= 0.001 step_size_up = 15 mode = exp_range	39.2	39.667	56.600	64.600
4.0	SGD plateau	0.01	mode=min	37.2	35.933	52.267	60.533
4.1	SGD plateau	0.001	mode=min	44.2	42.200	59.800	66.600
5.0	SGD restart	0.01	T_0 = 10, T_mult= 1, eta_min= 10^{-6}	36.6	36.067	53.800	62.067
5.1	SGD restart	0.001	T_0 = 20, T_mult= 1, eta_min= 10^{-8}	43.8	43.000	85.733	65.800
5.2	SGD restart	0.005	T_0 = 20, T_mult= 1, eta_min= 10^{-8}	42.8	39.867	57.133	64.067

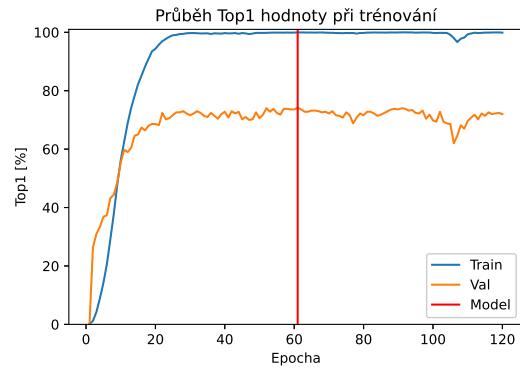
Tabulka 7
Trénování I3D ResNet50 na segmentačních datech

Pokus	Optimizer scheduler	lr	nastavení	Top1 [%] (val)	Top1 [%] (test)	Top3 [%] (test)	Top5 [%] (test)
1.0.	SGD step	0.01	step= 20	26.4	23.066	39.800	47.333
1.1.	SGD step	0.001	step= 20	43.8	41.467	61.333	69.267
1.2.	SGD step	0.0001	step= 20	22.6	20.467	37.133	45.867
2.0.	SGD cosine	0.01	eta_min=0	36.4	29.2	45.800	53.800
2.2.	SGD cosine	0.005	eta_min=0	44.2	41.133	60.267	66.867
2.1.	SGD cosine	0.001	eta_min=0	46.2	44.000	61.533	69.066
2.3.	SGD cosine	0.0001	eta_min=0	8.0	5.600	12.200	16.400
3.0	SGD cyclic	10^{-8}	max_lr= 0.001 step_size_up = 15 mode = triangular	42.8	42.200	59.267	65.800
3.1	SGD cyclic	10^{-8}	max_lr= 0.001 step_size_up = 15 mode = triangular2	40.4	40.200	58.800	66.800
3.0	SGD cyclic	10^{-8}	max_lr= 0.001 step_size_up = 15 mode = exp_range	46.0	41.467	60.200	67.133
4.0	SGD plateau	0.01	mode=min	31.2	26.200	43.067	50.200
4.0	SGD plateau	0.001	mode=min	44.4	43.2	62.333	68.333
5.0	SGD restart	0.01	T_0 = 10, T_mult= 1, eta_min= 10^{-6}	36.0	30.733	46.867	54.400
5.0.1	SGD restart	0.01	T_0 = 20, T_mult= 1, eta_min= 10^{-8}	34.6	31.733	47.867	54.667
5.1	SGD restart	0.001	T_0 = 20, T_mult= 1, eta_min= 10^{-8}	45.8	42.267	60.533	67.867
5.2	SGD restart	0.005	T_0 = 20, T_mult= 1, eta_min= 10^{-8}	44.6	40.000	58.267	64.267

Tabulka 8
Trénování I3D ResNet50 na datech optického toku

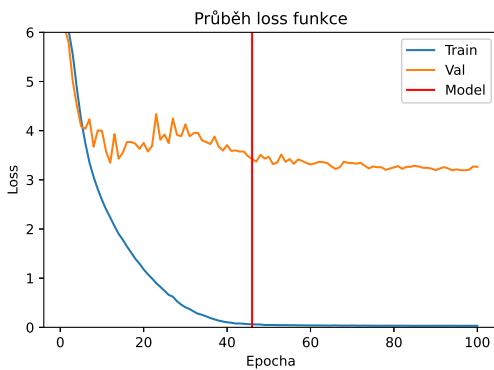


Obrázek 23
Průběh loss funkce při trénování,
(rgb data, pokus **3.0**)

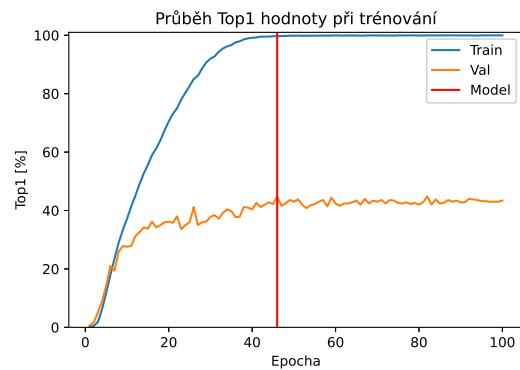


Obrázek 24
Průběh Top1 hodnoty při trénování,
(rgb data, pokus **3.0**)

Tady ještě budu popisovat grafy



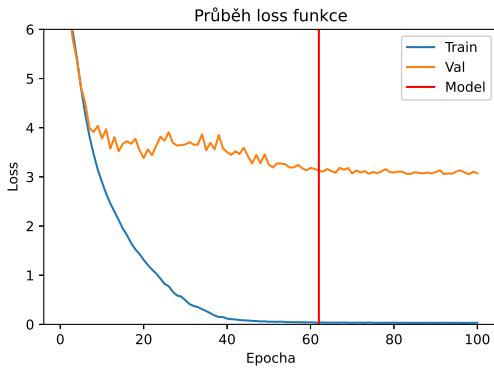
Obrázek 25
Průběh loss funkce při trénování,
(segmentační data, pokus **2.1**)



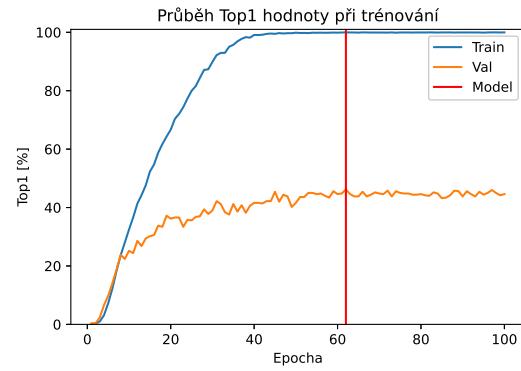
Obrázek 26
Průběh Top1 hodnoty při trénování,
(segmentační data, pokus **2.1**)

Tady ještě budu popisovat grafy

Tady ještě budu popisovat grafy



Obrázek 27
Průběh loss funkce při trénování - data optického toku,
(data optického toku, pokus **2.1**)



Obrázek 28
Průběh Top1 hodnoty při trénování - data optického toku,
(data optického toku, pokus **2.1**)

7.4 Normalizace dat

Data	mean	std
ImageNet	[0.485, 0.456, 0.406]	[0.485, 0.456, 0.406]
RGB	x	x
Segmentační data	[0.131, 0.084, 0.012]	[0.294, 0.192, 0.031]
Data optického toku	[0.912, 0.895, 0.910]	[0.178, 0.191, 0.190]

Tabulka 9
Ukázka hodnot *mean* a *std*

Data	Top1 [%] ImageNet	Top1 [%] vlastní hodnoty
Segmetnační data x	x	x
Data optického toku	x	x

Tabulka 10
Hodnoty Top1 při různých normalizacích dat

7.5 Spojení třech modelů

Nejlepší výsledky:

Data	Top1 [%]	Top3 [%]	Top5 [%]
RGB data	70.267	88.000	91.933
Segmentační data	39.733	55.267	62.467
Data s optickým tokem	36.066	54.333	61.400

Tabulka 11
Tabulka nejlepších výsledků

$$X = \alpha X_{rgb} + \beta X_{segmentace} + \gamma X_{opticky_tok} \quad (6)$$

α	β	γ	Top1 [%]	Top3 [%]	Top5 [%]
1.0	1.0	1.0	69.267	84.200	89.400
1.0	0	1.0	68.867	84.533	89.933
1.0	1.0	0	69.000	84.933	90.333
0	1.0	1.0	46.200	62.400	69.333

Tabulka 12
Tabulka přesnosti rozpoznávání pro různé parametry α, β, γ

	α	β	γ	Top1 [%]	Top3 [%]	Top5 [%]
Test na validační setu	0.55	0.35	0.29	76.000	86.200	91.200
Test na testovacím setu	0.55	0.35	0.29	72.200	85.200	90.400

Tabulka 13
Tabulka přesnosti rozpoznávání pro různé parametry α, β, γ

$$X = X_{rgb} \cdot X_{segmentace} \cdot X_{opticky_tok} \quad (7)$$

	Top1 [%]	Top3 [%]	Top5 [%]
Test na validační setu	73.800	87.200	92.400
Test na testovacím setu	72.867	88.667	92.133

Tabulka 14

Tabulka přesnosti rozpoznávání pro různé parametry $\alpha, \beta\gamma$

$$X = X_{rgb} \cdot (\beta X_{segmentace} + \gamma X_{opticky_tok}) \quad (8)$$

	β	γ	Top1 [%]	Top3 [%]	Top5 [%]
Test na validační setu	0.80	0.13	77.800	89.800	94.20
Test na testovacím setu	0.80	0.13	74.800	89.933	93.400

Tabulka 15

Tabulka přesnosti rozpoznávání pro různé parametry $\alpha, \beta\gamma$

7.6 Klasifikátory

Trénování na validační množině (500 vzorů), pak testováno na testovací množině :

Klasifikátor	Top1 [%]	Top3 [%]	Top5 [%]
LinearSVC(C=0.5)	50.6	70.19	76.86
LinearSVC(C=1)	50.6	70.19	76.86
SVM(kernel = linear)	40.8	51.47	53.60
SVM(kernel = rbf)	40.8	51.47	53.60
KNN(n=1)	40.8	41.0	41.33
KNN(n=3)	10.33	51.47	51.73

Tabulka 16

t1

Trénování na trénovací množině (8000 vzorů), pak testováno na testovací množině :

Klasifikátor	Top1 [%]	Top3 [%]	Top5 [%]
LinearSVC(C=0.5)	58.13	83.93	88.8
LinearSVC(C=1)	59.19	83.53	88.53
SVM(kernel = linear)	52.80	79.33	81.53
SVM(kernel = rbf)	48.40	52.93	52.20
KNN (n=1)	51.6	51.73	51.93
KNN (n=3)	48.8	56.13	56.13

Tabulka 17

t2

Trénování na trénovací množině + validační množině (8500 vzorků), pak testováno na testovací množině:

Klasifikátor	Top1 [%]	Top3 [%]	Top5 [%]
LinearSVC(C=0.5)	66.00	84.40	88.80
LinearSVC(C=1)	67.06	83.6	88.33
SVM(kernel = linear)	58.067	68.40	71.27
SVM(kernel = rbf)	49.73	56.47	60.33
KNN (n=1)	52.27	52.33	52.47
KNN (n=3)	47.27	58.93	59.13

Tabulka 18

t3

8 Závěr

Zatím nic

9 Seznam literatury

Odkazy

- [1] Mathieu Aubry et al. "Seeing 3D chairs: exemplar part-based 2D-3D alignment using a large dataset of CAD models". In: *CVPR*. 2014.
- [2] Vitaly Bushaev. *How do we 'train' neural networks ?* 2018. URL: <https://towardsdatascience.com/how-do-we-train-neural-networks-edd985562b73>.
- [3] D. J. Butler et al. "A naturalistic open source movie for optical flow evaluation". In: *European Conf. on Computer Vision (ECCV)*. Ed. A. Fitzgibbon et al. (Eds.) Part IV, LNCS 7577. Springer-Verlag, říj. 2012, s. 611–625.
- [4] Chun-Fu Chen et al. "Deep Analysis of CNN-based Spatio-temporal Representations for Action Recognition". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Čvn. 2021.
- [5] Liang-Chieh Chen et al. *Rethinking Atrous Convolution for Semantic Image Segmentation*. 2017. DOI: 10.48550/ARXIV.1706.05587. URL: <https://arxiv.org/abs/1706.05587>.
- [6] Jihoon Chung et al. "HAA500: Human-Centric Atomic Action Dataset with Curated Videos". In: *ICCV 2021*.
- [7] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, s. 248–255.
- [8] Shuchen Du. *Understanding Optical Flow amp; Raft*. 2020. URL: <https://towardsdatascience.com/understanding-optical-flow-raft-accb38132fba>.
- [9] Andreas Geiger et al. "Vision meets Robotics: The KITTI Dataset". In: *International Journal of Robotics Research (IJRR)* (2013).
- [10] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. DOI: 10.48550/ARXIV.1512.03385. URL: <https://arxiv.org/abs/1512.03385>.
- [11] Qingge Ji et al. "Optimized Deep Convolutional Neural Networks for Identification of Macular Diseases from Optical Coherence Tomography Images". In: *Algorithms* 12 (ún. 2019), s. 51. DOI: 10.3390/a12030051.
- [12] Will Kay et al. "The Kinetics Human Action Video Dataset". In: *CoRR* abs/1705.06950 (2017). arXiv: 1705.06950. URL: <http://arxiv.org/abs/1705.06950>.
- [13] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context*. 2014. DOI: 10.48550/ARXIV.1405.0312. URL: <https://arxiv.org/abs/1405.0312>.
- [14] Karen Simonyan a Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2014. DOI: 10.48550/ARXIV.1409.1556. URL: <https://arxiv.org/abs/1409.1556>.
- [15] Christian Szegedy et al. *Rethinking the Inception Architecture for Computer Vision*. 2015. DOI: 10.48550/ARXIV.1512.00567. URL: <https://arxiv.org/abs/1512.00567>.
- [16] Zachary Teed a Jia Deng. *RAFT: Recurrent All-Pairs Field Transforms for Optical Flow*. 2020. DOI: 10.48550/ARXIV.2003.12039. URL: <https://arxiv.org/abs/2003.12039>.
- [17] Jonathan Tremblay, Thang To a Stan Birchfield. *Falling Things: A Synthetic Dataset for 3D Object Detection and Pose Estimation*. 2018. DOI: 10.48550/ARXIV.1804.06534. URL: <https://arxiv.org/abs/1804.06534>.

- [18] Mrinal Tyagi. *Image segmentation nbsp;; Part 1*. 2021. URL: <https://towardsdatascience.com/image-segmentation-part-1-9f3db1ac1c50>.

10 Seznam obrázků

Seznam obrázků

1	CIFRA-10 trénovací a testovací error původní obrázek: https://arxiv.org/pdf/1512.03385.pdf (strana 1)	6
2	Residuální blok	6
3	Bottleneck blok	7
4	Blok A	8
5	Blok B	8
6	Blokově znázorněná architektura I3D ResNet 50	9
7	Kroková strategie <i>lr</i>	12
8	Kosínová strategie <i>lr</i>	12
9	Cyklická strategie <i>lr</i> (jeden cyklus)	12
10	Strategie restartováním <i>lr</i>	12
11	<i>HAA500</i> fotbalový dribbling	13
12	<i>HAA500</i> fotbalová hlavička	13
13	<i>HAA500</i> fotbalový zákrok brankáře	14
14	Ukázka segmentace - hod diskem	15
15	Ukázka segmentace - pád na kole	15
16	Ukázka segmentace - hod diskem	16
17	Ukázka segmentace - baseball odpal míčku	17
18	Distribuce tříd	18
19	<i>Kinetics-400</i> snowkitting	18
20	<i>Kinetics-400</i> sfoukávání svíček	18
21	<i>Kinetics-400</i> bobování	19
22	Ukázka augmentace (pád na kole)	20
23	Průběh loss funkce při trénování, (rgb data, pokus 3.0)	27
24	Průběh Top1 hodnoty při trénování, (rgb data, pokus 3.0)	27
25	Průběh loss funkce při trénování, (segmentační data, pokus 2.1)	27
26	Průběh Top1 hodnoty při trénování, (segmentační data, pokus 2.1)	27
27	Průběh loss funkce při trénování - data optického toku, (data optického toku, pokus 2.1)	28
28	Průběh Top1 hodnoty při trénování - data optického toku, (data optického toku, pokus 2.1)	28

11 Seznam tabulek

Seznam tabulek

1	Počet dat	14
2	Počet dat	17
3	Příklad testování úspěšnosti klasifikace videí	21
4	Počet dat	22
5	Závislost přesnosti rozpoznávání a počtu klíčových snímků	22
6	Trénování I3D ResNet50 na rgb datech	24
7	Trénování I3D ResNet50 na segmentačních datech	25
8	Trénování I3D ResNet50 na datech optického toku	26
9	Ukázka hodnot <i>mean</i> a <i>std</i>	28
10	Hodnoty Top1 při různých normalizacích dat	28
11	Tabulka nejlepších výsledků	29
12	Tabulka přesnosti rozpoznávání pro různé parametry $\alpha, \beta\gamma$	29
13	Tabulka přesnosti rozpoznávání pro různé parametry $\alpha, \beta\gamma$	29
14	Tabulka přesnosti rozpoznávání pro různé parametry $\alpha, \beta\gamma$	30
15	Tabulka přesnosti rozpoznávání pro různé parametry $\alpha, \beta\gamma$	30
16	t1	31
17	t2	31
18	t3	31