

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

DIPLOMOVÁ PRÁCE

PLZEŇ, 2021/2022

Adam Klečka

Anotace

Diplomová práce se zabývá úlohou rozpoznávání činností z videonahrávek. Pro řešení této úlohy byla zvolena metoda I3D, která transformuje konvoluční neuronové sítě tak, aby mohly pracovat s časo-prostorovými informacemi uložené ve videích. Jako *backbone* pro I3D byla zvolen model ResNet50. V práci budou prováděny experimenty s datasetem *HAA500*, na který budou aplikovány metodiky předzpracování videa pomocí segmentace a optického toku. Modely natrénované na odlišných datech se v závěru práce budou mezi sebou porovnávat.

Klíčová slova: strojové vidění, rozpoznávání činností, I3D, ResNet, I3D ResNet50, segmentace, optický tok

Annotation

This work deals with the task of action recognition from video recordings. The I3D method was chosen to solve this task. This method inflates convolutional neural networks so that they can work with spatio-temporal information stored in videos. The ResNet50 model was chosen as the *backbone* for I3D. Experiments in this work will be conducted with the *HAA500* dataset. On this dataset will be applied video preprocessing methodologies using segmentation and optical flow. The models trained on different data will be compared with each other at the end of the research work.

Keywords: computer vision, action recognition, I3D, ResNet, I3D ResNet50, segmentation, optical flow

Poděkování

Především bych rád poděkoval vedoucímu mé diplomové práce Ing. Ivanu Gruberovi, Ph.D. za užitečné rady a čas strávený při konzultacích. Dále bych chtěl poděkovat organizaci MetaCentrum za poskytnutí výpočetních prostředků pro splnění této práce.

Obsah

1	Úvod	5
2	ResNet	7
3	I3D	9
3.1	I3D ResNet50	9
3.2	Klíčové snímky	11
3.3	Trénování sítě	12
4	Dataset	15
4.1	Kinetics-400	15
4.2	HAA500	17
4.2.1	Zpracování RGB datasetu	18
4.2.2	Zpracování segmentačního datasetu	18
4.2.3	Zpracování optického toku	19
5	Augmentace	22
6	Použitá metrika	23
7	Experimenty s I3D ResNet50	24
7.1	Implementace modelu I3D ResNet50	24
7.2	Vliv počtu klíčových snímků	24
7.3	Vliv využití předtrénovaných vah na datasetu <i>Kinetics-400</i>	25
7.4	Vliv normalizace dat	26
7.5	Trénování na RBG, segmentačních datech a datech po použití optického toku	28
7.6	Porovnání modelů	34
7.7	Spojení modelů	36
7.8	Využití klasifikátorů	38
8	Závěr	41
9	Seznam literatury	43
10	Seznam obrázků a tabulek	45

1 Úvod

Lidé snadno rozpoznávají činnosti, které jsou nahrané na videu. Tato schopnost je pro nás přirozená. V průběhu sledování videa pozorujeme objekty a jejich pohyby. Díky zkušenostem dokážeme pak vyhodnotit co se na videozáznamu děje a určit jakou činnost člověk na video nahrávce vykonával. Tato práce se bude zabývat počítacovou automatizací tohoto procesu. Pokusíme se vytvořit rozpoznávací systém, který bude z videonahrávek jednoznačně určovat lidské činnosti.

Rozpoznávání činností z video nahrávek patří do oblasti umělé inteligence. Tato úloha má zásadní význam pro řadu aplikací v reálném světě. Uplatnění může mít například pro automatické labelování krátkých videí na sociálních sítích, 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 například 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* [13], která transformuje standardní architektury *CNN* (*Convolutional neural network*) pro klasifikování obrázků. *I3D* transformuje *max-pool* a konvoluční filtry z dvojrozměrné podoby ($N \times N$) do trojrozměrné ($N \times N \times N$). Díky tomu pak může sítě sbírat časoprostorové informace z videa. Často volené *backbone* sítě pro *I3D* trasformaci jsou *VGG-16* [17], *Inception* [18] nebo architektura *ResNet* [10], která bude pro tuto práci klíčová, protože s ní budou prováděny experimenty. V práci bude vysvětlen princip residuálních bloků a jejich *skip connection*. Dále bude vysvětleno fungování *bottleneck* bloků a důkladně bude probrána transformovaná architektura *ResNet* s hloubkou 50.

Cílem experimentů je natrénovat co nejlepší model ve smyslu *Top1* přenosti s vybraným datasetem. Pro trénování a testování sítě *I3D ResNet50* bude využitý dataset s názvem *HAA500*[6]. Ten obsahuje ručně anotovaná videa 500-ti odlišných lidských činností, celkem má 10000 videí. Pro každou činnost z datasetu připadá 20 unikátních videí. Při trénování budou využity předtrénované váhy na datasetu *Kinetics-400* [12]. Otestujeme a porovnáme jakých výsledků dosáhneme na předtrénovaných vahách a na náhodně inicializovaných. Experimentálně bude také testována závislost mezi zvoleným počtem klíčových snímků a rozpoznávací přesnosti *Top1[%]*.

Tato práce bude testovat různé metody předzpracování obrazu videí a dokumentovat jejich počin při trénování a klasifikování pomocí *I3D ResNet50* modelu. Za účelem zjednodušení obrazu bude na videa z datasetu *HAA500* aplikována segmentace a otestujeme jakých klasifikačních výsledků bude díky segmentaci dosaženo. Pro segmentaci bude využit model *DeepLab v3* [5] s *backbone* sítí *ResNet50*. Na RGB videa bude také aplikovaná metoda optického toku pomocí modelu *RAFT* [19]. Optický tok predikuje a zvýrazňuje pohyb objektů na videu. I na data optického toku bude provedeno trénování sítě *I3D ResNet50* a bude důkladně porovnán s ostatními modely.

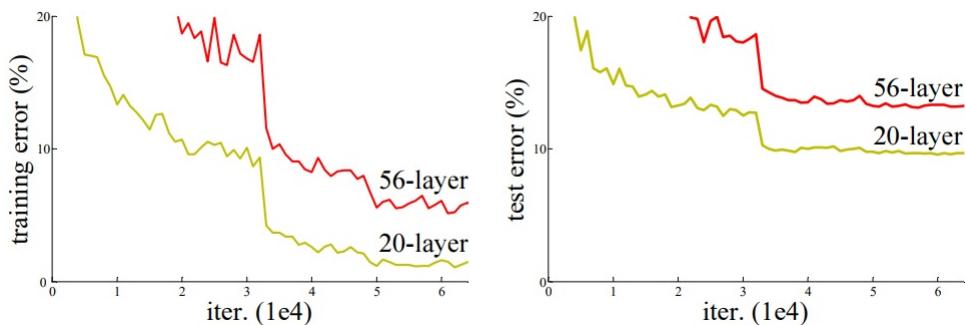
V závěru práce se pokusíme s spojení modelů *I3D ResNet50* natrénovaných na klasifi-

sických RGB datech, segmentovaných datech a datech reprezentující optický tok. Spojení bude prováděno pomocí sloučení výstupních *softmax* matic z modelů. Bude vyzkoušeno také trénování klasifikátorů.

2 ResNet

ResNet (Residual Network) [10] je konvoluční neuronová síť. Základ konvolučních sítí představují konvoluční filtry, které provádějí konvoluční operaci na vstupní data. Sítě *ResNet* se často využívají pro klasifikaci RGB obrázků, které reprezentují vstupní data.

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 ukázанé na Obrázku 1, kde je průběh trénování hluboké sítě na datasetu *CIFAR-10* [14]. Jak je vidět, neuronová síť s 20 vrstvami dosáhla lepších trénovacích i testovacích výsledků než síť s 56 vrstvami.

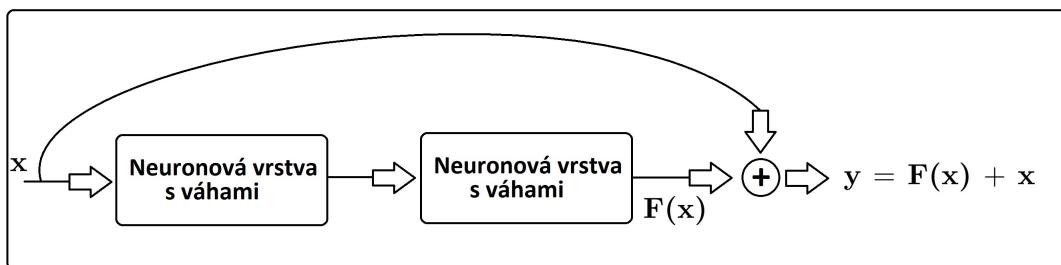


Obrázek 1

CIFAR-10 trénovací a testovací error

původní obrázek: <https://arxiv.org/pdf/1512.03385.pdf> [10] (strana 1)

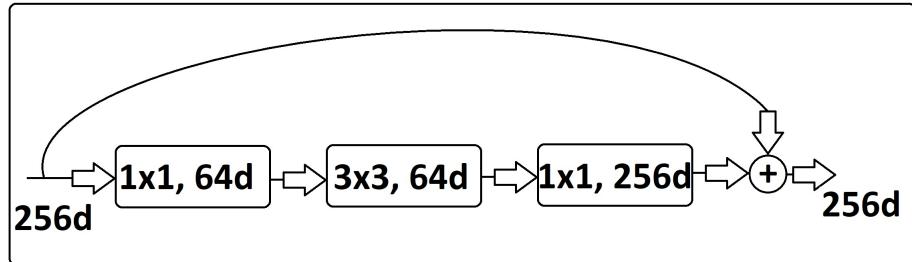
Důvodem může být špatná optimalizace sítě, ztrácející se gradient nebo špatné trénování po přidání nových vrstev do natrénované sítě. ResNet přišel s řešením a to přidáním residiálních bloků (viz. Obrázek 2).



Obrázek 2
Residiální blok

Residiální 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 pomocí konvoluční vrstvy s velikostí jádra (1x1) do 64d a na konci bloku zpět rozšířit na 256d(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.

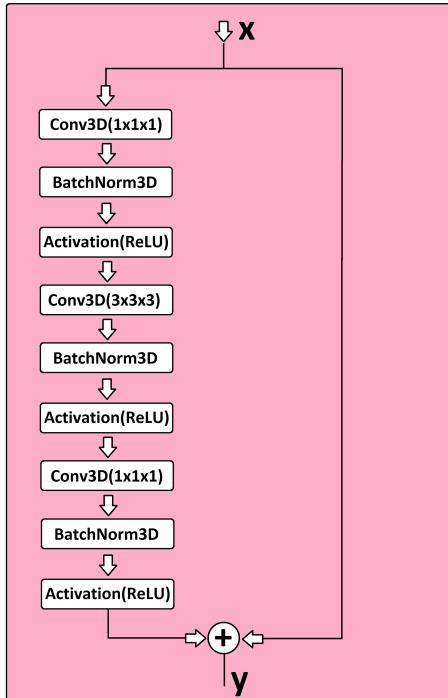
3 I3D

I3D (Inflated 3D ConvNets) [13] je způsob, jak z videí sbírat časo-prostorové informace. V úvodu už byla zmínka o takzvaném transformování konvolučních neuronových sítí pro klasifikaci 2D obrazu. Princip *I3D* transformace je takový, že se jako *backbone* síť zvolí funkční a léty prověřená architektura na klasifikování obrázků jako je například VGG-16 [17], Inception [18] nebo ResNet [10]. Po zvolení *backbone* sítě se převedou její *max-pool* a konvoluční filtry z podoby 2D ($N \times N$) na podobu 3D ($N \times N \times N$). To znamená, že pokud je ve vrstvě konvoluční jádro o rozměrech (3x3), převede se na tvar (3x3x3). Vstupem do takto transformované sítě je pak posloupnost 2D obrázků videa. Můžeme tedy říci, že rozšířený třetí rozměr reprezentuje tok času videa. 3D konvoluční jádra se díky tomu dokáží pohybovat napříč časem a prostorem a získávají tak z videa důležité příznaky.

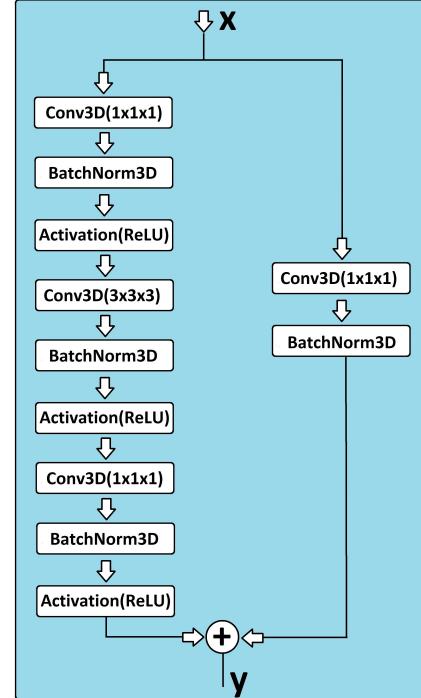
I3D modely dosahují kvalitních výsledků, pokud jsou k nim implementovány předtrénované váhy z ImageNet[7]. To lze docílit pomocí takzvaného *nudného* videa. Je to video složené pouze z jednoho snímku, které se opakuje a je časově neměnné. Trénování 3D sítě podle *nudného* videa se síť vlastně učí klasifikovat obrázky, jakoby by byla síť 2D. 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 transformována a nyní se z ní stala síť, která operuje s trojrozměrnými konvolučními filtry a získává časové závislosti z videa. I3D ResNet50 obsahuje celkem 48 konvolučních vrstev, jeden *max-pool*. Vrstvy sítě I3D ResNet50 jsou poskládané ze 2 typů bloků. První blok nazveme například blok A (Obrázek 4).



Obrázek 4
Blok A



Obrázek 5
Blok B

V levé větvi bloku A (Obrázek 4) je na vstupní data použité konvoluční jádro o rozměru $(1 \times 1 \times 1)$ s hodnotou $stride = (1 \times 1 \times 1)$, hodnota $stride$ představuje krok po kterém se konvoluční jádro posouvá po vstupních datech. Tato konvoluční vrstva má za úkol snížit dimenze 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 (rovnice (1)):

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

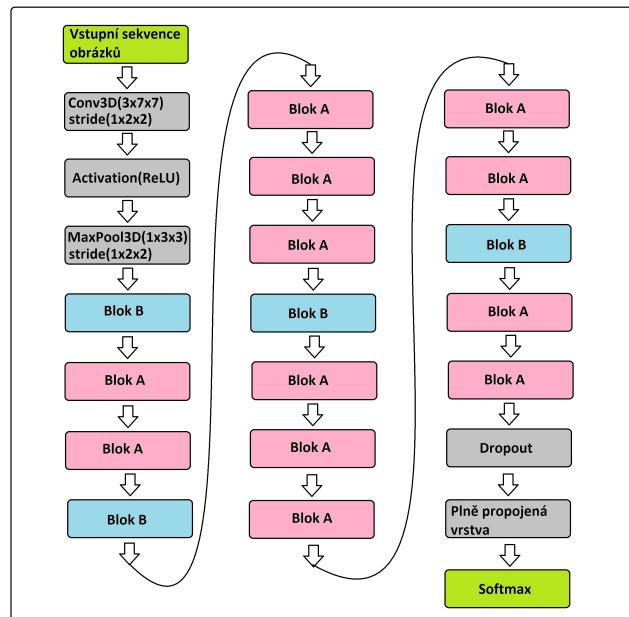
Aktivační funkce ReLU vrací nulovou hodnotu, při jakémkoliv záporném vstupu, v opačném případě vrací hodnotu vstupu x .

Tyto tři vrstvy (Conv3D, BatchNorm3D, ReLU) se v levé větvi Bloku A opakují ještě 2x, s tím že podruhé má konvoluční jádro o rozloze $(3 \times 3 \times 3)$ a na závěr konvoluční jádro $(1 \times 1 \times 1)$, které má v *bottleneck* bloku za úkol zvýšit zpět dimenze na původní hodnoty. V pravé větvi Bloku A není žádná konvoluční vrstva, její účel je pouze poskytnout vstupní data pro součet na konci bloku se zpracovanými daty z levé větve Bloku A.

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 s konvolučním jádrem $(1 \times 1 \times 1)$. Díky tomu blok dochází na výstupu ke snížení prostorové dimenze, zatímco Blok A má jak na výstupu tak i na vstupu stejnou

dimenzi.

Celá architektura sítě I3D ResNet50 poskládaná z Bloků A a B je znázorněná na Obrázku 6. Na začátku je na vstupní posloupnost obrázků aplikována vrstva s konvolučním jádrem o rozměru $(3 \times 7 \times 7)$ s krokovou hodnotou $stride = (1 \times 2 \times 2)$. Následuje $ReLU$ aktivační funkce, $max - pool$ 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* vrstva zpracuje hodnoty z plně propojené vrstvy a každé třídě přiřadí pravděpodobnost. Součet všech pravděpodobností ze *softmax* matice je roven hodnotě 1.



Obrázek 6
Blokově znázorněná architektura I3D ResNet 50

3.2 Klíčové snímky

Videa se skládají z několika snímků. V praxi z důvodu omezené *VRAM* paměti musíme jako vstup do sítě I3D ResNet50 volit fixní počet klíčových snímků. Z celého videa se tedy musí vybírat klíčové snímkovy, v nich by měly být obsaženy všechny důležité okamžiky činnosti. Ideálně by měly tyto snímkovy měly být vybírány z celého videa. Kdyby se například klíčové snímkovy braly pouze z prostřední části videa, hrozila by zde situace vynechání důležitých snímků, které definují činnost.

Nejednoduší případ by byl, kdyby byla všechna videa stejně dlouhá. V takovém případě bychom mohli vybírat klíčové snímkovy s definovaným krokem, které by pak 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.

Počet klíčových snímků bude volen 32. Jako strategii výběru 32 klíčových snímků byla vybrána poměrně jednoduchá a intuitivní metoda. Vždy chceme pokrýt celé video klíčovými snímkami. Proto nejprve proběhne spočítání indexu kroků γ (rovnice (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 video má méně obrázků než je požadovaný počet klíčovým snímků. V takovém případě se snímkы rovnoměrně nakopírují za sebe. Vždy samozřejmě nemůžeme provést rovnoměrné kopírování, proto se v takovém případě vyberou náhodné snímkы, které budou nakopírovány víckrát, či méněkrát. Tento případ bude demonstrován na příkladě, kde vektor $Index_{ks}$ reprezentuje vybrané klíčové snímkы z videa:

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

Za podmínky, že se bude hodnota γ rovna 1, tak je výběr klíčových 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 $\gamma \geq 2$. V takovém případě se rozdělí obrázky videa do skupin se stejným krokem. Velikost kroku je rovna hodnotě γ zaokrouhleno dolů. Následně se z těchto vybraných skupin vybere jedna, která bude představovat klíčové snímkы 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 (rovnice rovnice (3)):

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

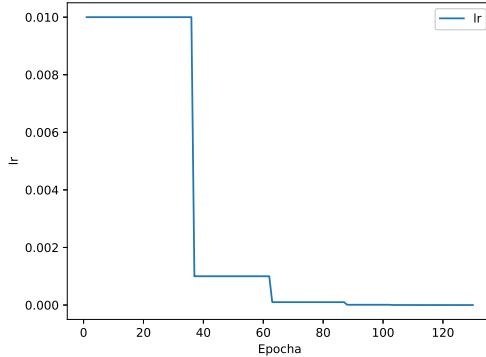
kde x je index třídy, N celkový počet tříd, $p()$ je binární indikátor správné klasifikace a funkce $q()$ reprezentuje výstup softmaxu. Čím přesněji bude síť správně klasifikovat trénovací videa, tím se bude hodnota *Loss* funkce snižovat. Cílem trénování je tedy co nejvíce snížit hodnotu této funkce.

Celý trénovací proces se dá pojmetout jako úloha pro minimalizování *Loss* funkce. Je zde několik metod, které se snaží tuto úlohu řešit, růžíká se jím optimizéry. Jedním z velmi oblíbených algoritmů je takzvaný SGD (*Stochastic Gradient Descent*), matematicky jej lze zapsat jako (rovnice (4)) :

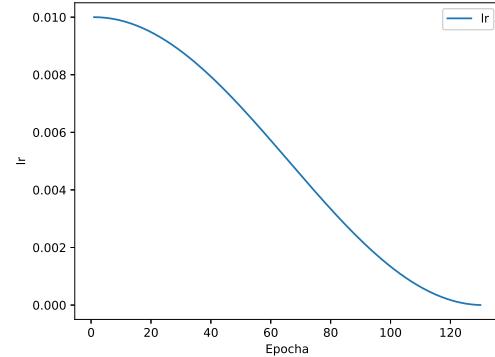
$$w_j = w_j - lr \frac{\partial L}{\partial w_j}, \quad (4)$$

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 vyčíslit gradient $\partial L / \partial w_j$. Pro to lze použít metodu zvanou *Backpropagation* [15], kdy zpětně procházíme síť a získáváme gradienty vah mezi jednotlivými vrstvami.

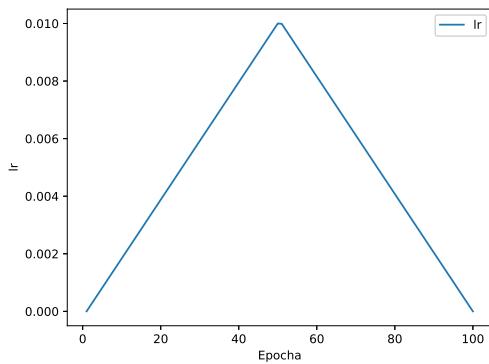
Abychom při trénování dokonvergovali ideálně ke globálnímu minimu *Loss* funkce, je důležité také zvolit vhodnou strategii úpravy konstanty učení lr . Těch je hned několik. Můžeme zvolit například běžné krokové snížení N epochách (Obrázek 7), kosínovou strategie (Obrázek 8) , cyklickou (Obrázek 9) nebo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



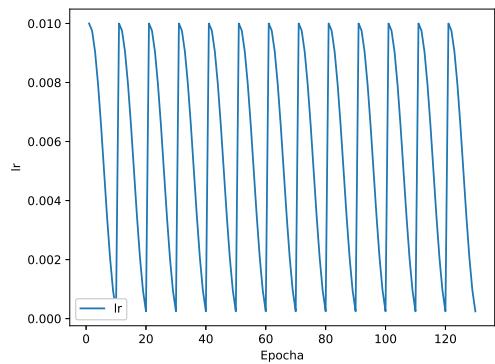
Obrázek 7
Kroková strategie lr



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 klasifikaci 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 Kinetics-400

Důležitý dataset pro tuto práci je *Kinetics-400* [12]. 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.

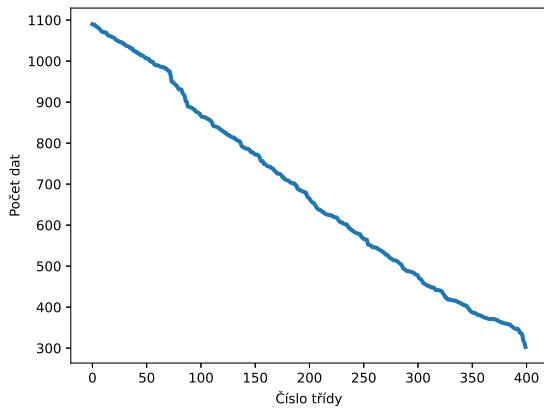
Kinetics 400 je kolekce videí s lidskou činností. Tyto videa jsou reprezentovány pomocí URL odkazu na videa internetového serveru *YouTube.com*. U každého URL odkazu je zaznamenaný počáteční a koncový čas označené činnosti, přičemž každé video trvá zhruba 10 vteřin. Činnosti jsou zaměřeny na člověka 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 1. 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 1
Počet dat

Při analýze datasetu je důležité zjistit jak jsou data v jednotlivých třídách rozprostřeny. Distribuce dat mezi třídami je zobrazena na Obrázku 11.



Obrázek 11
Distribuce tříd

Z Obrázku 11 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 12,13 a 14.



Obrázek 12
Kinetics-400 snowkitting



Obrázek 13
Kinetics-400 sfoukávání svíček



Obrázek 14
Kinetics-400 bobování

4.2 HAA500

Human-Centric Atomic Action (HAA500) [6] je ručně anotovaná sada dat videí s lidskou činností. Videá 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 15, 16, 17). Videá byla vybírána také tak, aby osoba provozující klasifikovanou činnost byla dominantní osobou videa. Hlavní část videa je zaměřená 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 15
HAA500 fotbalový dribbling



Obrázek 16
HAA500 fotbalová hlavička



Obrázek 17
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 2).

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

Tabulka 2
Počet dat

4.2.1 Zpracování RGB datasetu

Videa poskytnuté datasetem *HAA500* byla uložena 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.

4.2.2 Zpracování segmentačního datasetu

Segmentace [21] 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* [16]. Tento

předtrénovaný model obsahuje celkem 20 tříd, které je síť schopna segmentovat. Na datasetu *COCO val2017*[16] 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 18 a 19. Je na nich vidět, že lidé na videích byli korektně segmentováni a označeni oranžovou barvou.



Obrázek 18
Ukázka segmentace - hod diskem



Obrázek 19
Ukázka segmentace - pád na kole

4.2.3 Zpracování optického toku

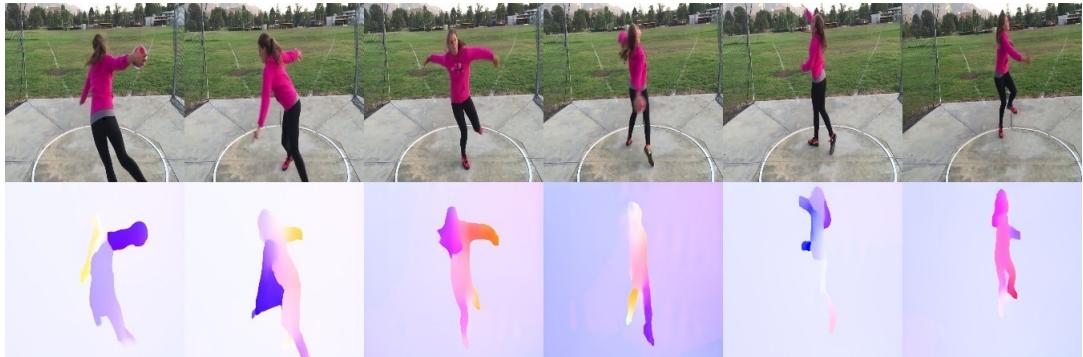
Optický tok [8] je technika používaná k popisu pohybu objektů z obrázků. Princip je založený na výpočtu rychlosti bodů v rámci několika snímků a predikováním kde by se body mohly na následujících snímcích nacházet. Metoda má uplatnění hlavně pro videa, protože snímky videa mají obvykle malý časový krok. Bude zajímavé pozorovat, jaké výsledky s I3D Resnet50 síti 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* [19] *Recurrent All-Pairs Field Transforms*. S využitím neuronové sítě, kde je vše založené na latentních příznakových mapách, je vše přesnější a počítané s větší efektivitu, 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*[20] + *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 snímky 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 20 a 21. U obrázku 20 je vidět zabarvení pozadí. To je způsobené pohybující se kamerou. U Obrázku 21 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 20
Ukázka segmentace - hod diskem



Obrázek 21
Ukázka segmentace - baseball odpal míčku

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, abychom ve výsledku nevyřízly důležité pixely z videa. Bylo otestováno, že velké zvětšení oblasti například z videa (224x224) na (320x320) 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 oříznutí zpět na (224x224). 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 bude otočeno.

Pomocí těchto metod jsou neuronové sítě předkládány každou epochou 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 ResNet50* chtěli co nejvíce přiblížit.

7 Experimenty s I3D ResNet50

7.1 Implementace modelu I3D ResNet50

Pro experimenty, které budou v následujících kapitolách provedené, byl implementován model I3D ResNet50 [4] pomocí programovacího jazyka *Python*. Při trénování sítě se bude využívat *Cross Entropy Loss* funkce a pro získání gradientu bude využita metodika *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 videích, segmentvaných videích a videích po aplikaci optického toku.

7.2 Vliv počtu klíčových snímků

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 subsetu *HAA500*

Trénování proběhlo se sítí I3D ResNet50 pro 4,8,16 a 32 klíčových snímků videa s náhodně inicializovanými váhami modelu. Bylo zvoleno nastavení trénovacích parametrů jako:

1. *optimizer*: SGD,
2. *scheduler*: kosínový,
3. *lr*: 0.001,
4. počet epoch: 50

Vedlejší cíl toho experimentu bylo otestování funkčnosti programovacího kódu 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.

Hlavní cíl tohoto experimentu s malým subsetem *HAA500* bylo experimentální zjištění přesnosti klasifikace videí v závislosti na počtu zvolených klíčových snímků systému. Natrénované modely byly otestovány na testovací množině dat a záznam experimentu je znázorněný v Tabulce 5.

	Top1 [%] (test)	Top3 [%] (test)	Top5 [%] (test)
4 klíčové snímky	36.7	60.0	75.0
8 klíčových snímků	40.0	61.7	68.3
16 klíčových snímků	46.7	70.0	81.7
32 klíčových snímků	53.3	68.3	85.0

Tabulka 5

Závislost přesnosti rozpoznávání a počtu klíčových snímků

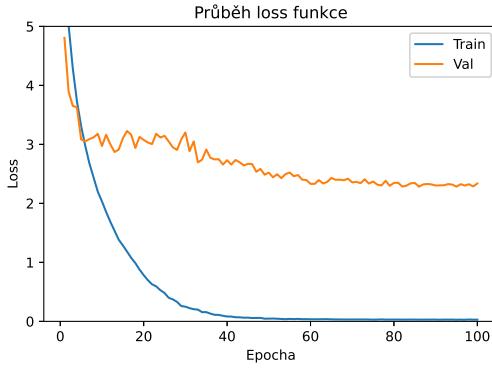
Z Tabulky 5 je patrné, že pro systém pracující s více klíčovými snímky roste také přesnost klasifikace testovaných videí. Díky této znalosti bude v následujících experimentech pracováno s modely s 32 klíčovými snímkami, jediná jejich nevýhoda je větší výpočetní složitost a tím také delší doba trénování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

7.3 Vliv využití předtrénovaných vah na datasetu *Kinetics-400*

Při tomto experimentu využijeme kompletní sadu dat *HAA500* (8000 trénovacích, 500 validačních, 1500 testovacích) s klasickými RGB videi. Otestujeme trénování modelu I3D ResNet50 vycházejícího z náhodně inicializovaných vah a z vah natrénovaných na datasetu *Kinetics-400* a následně porovnáme jejich klasifikační přesnosti. Využitím předtrénovaného modelu se myslí takzvaný *fine-tuning*. Při tomto procesu načteme stejné váhy modelu a odstraníme pouze 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. Pro oba případy budou voleny stejné parametry trénování sítě a to:

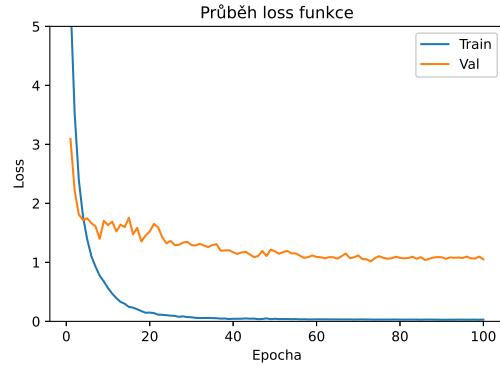
- *optimizer: SGD*,
- *scheduler: cosine*,
- počáteční *lr*: 0.001,
- počet epoch: 100.

Průběh změny *Loss* funkce při trénování na trénovací a validačním množině pro náhodně inicializované váhy modelu je na Obrázku 23 a trénování s hodnotami z *Kinetics-400* na Obrázku 24.



Obrázek 23

Průběh loss funkce při trénování,
náhodně inicializované váhy



Obrázek 24

Průběh loss funkce při trénování,
váhy z *Kinetics-400*

Při porovnávání grafů z Obrázků 23 a 24 je na první pohled patrné, že v obou případech *Loss* funkce konvergovala k nulové hodnotě na trénovacích datech. Hlavní rozdíl je v *Loss* funkci na validačních datech, kde model z náhodně inicializovaných vah dosáhl nejnižší hodnoty $Loss_{min} = 2.2826$ (86. epocha) a model vycházející z modelu *Kinetics-400* dosáhl $Loss_{min} = 1.0725$ (73. epocha). Vzali jsme modely z těchto okamžiků kdy dosáhli svá minima ve smyslu *Loss* funkce a byl proveden test na testovacích datech (Tabulka 6).

Výchozí váhy	Top1 [%] (test)	Top3 [%] (test)	Top5 [%] (test)
Náhodné vahy	50.2	68.8	76.0
<i>Kinetics-400</i>	72.2	86.4	90.3

Tabulka 6

Porovnání rozpoznávací přesnosti modelů s předtrénovanými a náhodně inicializovanými vahami

Testovací výsledky potvrdily lepší kvalitu klasifikování při použití předtrénovaných vah na datasetu *Kinetics-400* o více než 20% *Top1* oproti náhodně inicializovaným vah.

7.4 Vliv normalizace dat

Pro trénování konvolučních neuronových sítí na RGB obrázcích se doporučuje normalizovat kanály pixelů z hodnot $\langle 0, 255 \rangle$ na hodnoty $\langle 0, 1 \rangle$. To je možné jednoduchým vydělením kanálů číslem 255. Dále je ještě doporučené normalizovat tyto hodnoty vzhledem k průměru (rovnice (6)) a směrodatné odchylce(rovnice (7)) dat a to způsobem(rovnice (8)):

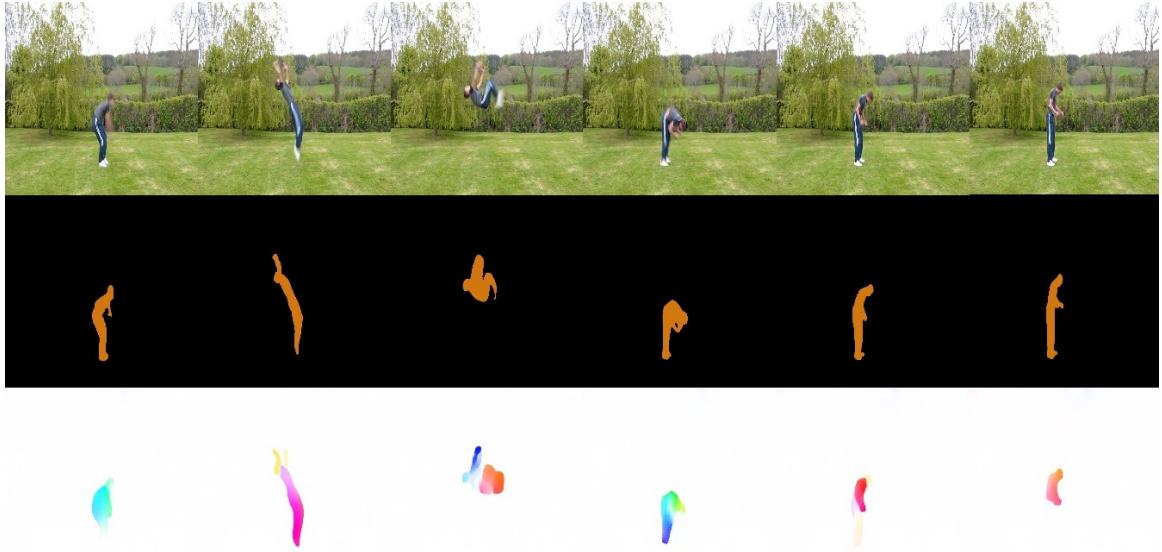
$$\mu = \frac{1}{n} \cdot \sum_1^n input_image_i, \quad (6)$$

$$\sigma^2 = \frac{1}{n} \cdot \sum_1^n input_image_i^2 - \mu^2, \quad (7)$$

$$norm_image = \frac{input_image - \mu}{\sigma}. \quad (8)$$

Tímto způsobem bychom měli získat hodnoty, které budou rozprostřeny okolo nuly. Tyto normalizace pomáhají získat konzistentní výsledky při použití natrénovaných modelů na nová data. Standardně se pro normalizaci RGB obrázků využívají hodnoty μ a σ spočítané z datasetu ImageNet. My ale budeme používat datasetsy, které mají netradiční barvy oproti klasickým RGB obrázkům.

Na Obrázku 25 můžeme vidět ukázku činnosti salto dozadu. První video reprezentuje standardní RGB video, druhé video je segmentované a třetí video znázorňuje optický tok. Na první pohled je patrné že segmentační data a data reprezentující optický tok budou mít odlišné hodnoty μ a σ oproti standardnímu RGB videu.



Obrázek 25
Ukázka činnosti salto vzad pro tři druhy zpracování videa

Experimentálně proto byly zjištěny hodnoty středních hodnot a směrodatných odchylek pro tři různé typy dat. Tyto hodnoty jsou zaznamenány v Tabulce 7. Hodnoty μ a σ mezi RGB (**HAA500**) videi a obrázky z ImageNetu se příliš neliší. Rozdíl je u segmentačních dat a optického toku. Segmentační data mají velkou část plochy černě zbarvenou a proto střední hodnota celého segmentačního datasetu má nízké hodnoty, přesný opak je u optického toku, kde nepohyblivé předměty jsou znázorněny bílou barvou a ve většině videí tato barva dominuje, proto jsou střední hodnoty kanálů vysoké.

Zpracování dat	Segmentace	μ	σ
RGB	ImageNet	[0.485, 0.456, 0.406]	[0.229, 0.224, 0.225]
RGB	HAA500	[0.471, 0.449, 0.428]	[0.282, 0.279, 0.283]
Segmentace	HAA500	[0.131, 0.084, 0.012]	[0.294, 0.192, 0.031]
Optický tok	HAA500	[0.912, 0.895, 0.910]	[0.178, 0.191, 0.190]

Tabulka 7
Experimentálně zjištěné hodnoty μ a σ

Bylo testováno, jak se při trénování různě normalizovaných dat změní přesnost *Top1* rozpoznávání na testovací množině. Trénování proběhlo na předtrénovaných váhách z *Kinetics-400* a nastavením trénovacích parametrů:

- *optimizer*: *SGD*,
- *scheduler*: *cosine*,
- počáteční *lr*: 0.001,
- počet epoch: 100.

Výsledek experimentu je zaznamenán v Tabulce 8.

Zpracování dat	Top1 [%] μ a σ z ImageNetu	Top1 [%] Experimentálně zjištěné μ a σ
RGB	72.2	71.9
Segmentace	44.1	42.0
Optický tok	44.0	43.7

Tabulka 8
Hodnoty Top1 při různých normalizacích dat

Výsledky z tabulky 8 vyšly hodně podobně, ale lepších výsledky dosáhly modely, které normalizují data podle hodnot z ImageNetu. Je to pravděpodobně zapříčiněné tím, že předtrénovaný model *Kinetics-400* používal právě tyto hodnoty k normalizaci. V následujícím experimentu se budou normalizovat všechny tři typy datasetů podle μ a σ z ImageNetu, tedy $\mu = [0.485, 0.456, 0.406]$ a $\sigma = [0.229, 0.224, 0.225]$.

7.5 Trénování na RBG, segmentačních datech a datech po použití optického toku

Doposud jsme zjistili závislost počtu snímku na kvalitě klasifikování, vliv předtrénovaných vah modelu a jak ovlivňuje celý model normalizace dat. Získané poznatky zúročíme v tomto trénovacím experimentu. Ten má za účel natrénovat co nejlepší modely ve smyslu *Top1* klasifikační přesnosti na neviděných testovacích datech. Abychom dosáhli co nejlepších výsledků, otestujeme různé kombinace *schedulery* a *lr*. *Schedulery* budou jmenovitě využívány:

- *step*,
- *cosine*,
- *cyclic*,
- *plateau*,
- *restart*.

V následujících 3 tabulkách (9, 10, 11) jsou zaznamenané trénovací parametry a jejich validační a testovací hodnoty přesnosti. Modely byly vybírány expertně na základě *Loss* funkce

a hodnoty $Top1$ na validačních datech. Nejlepší výsledky přesnosti na validačních datech jsou v tabulkách označeny žlutou barvou a nejlepší přesnosti na testovacích datech oranžovými barvami.

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

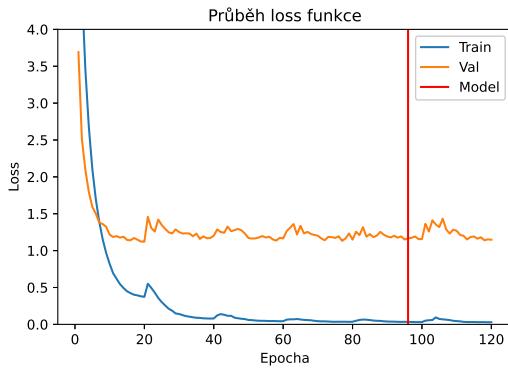
Tabulka 9
Trénování I3D ResNet50 na rgb datech

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
2.0.	SGD cosine	0.01	eta_min=0	37	34.733	51.067	58.000
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

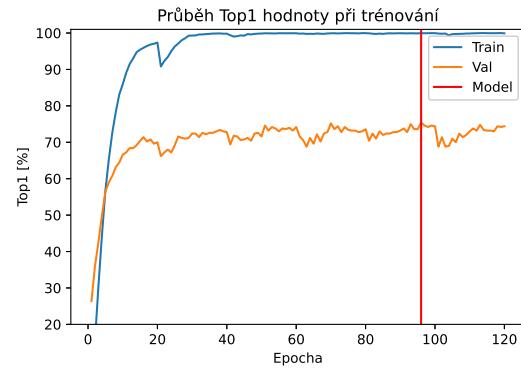
Tabulka 10
Trénování I3D ResNet50 na segmentačních datech

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
5.1	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

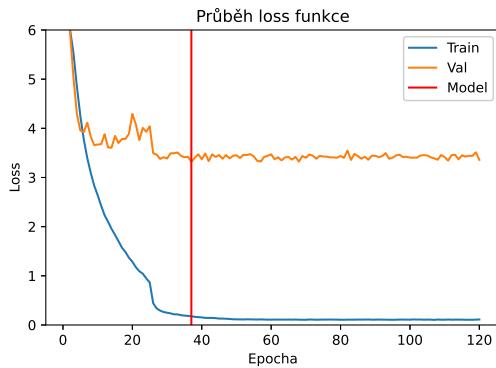
Tabulka 11
Trénování I3D ResNet50 na datech optického toku



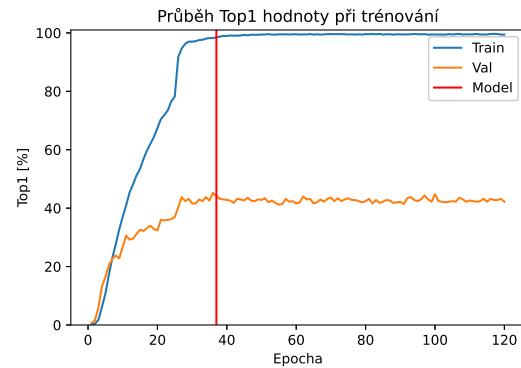
Obrázek 26
Průběh loss funkce při trénování,
(rgb data, pokus **5.2**)



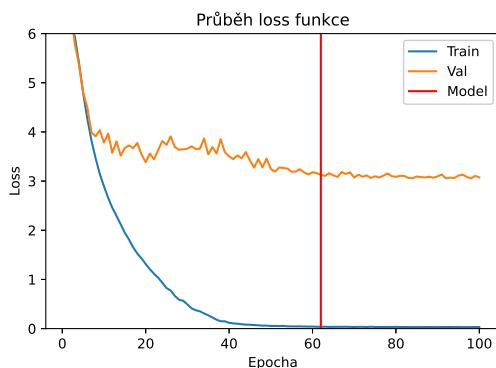
Obrázek 27
Průběh Top1 hodnoty při trénování,
(rgb data, pokus **5.2**)



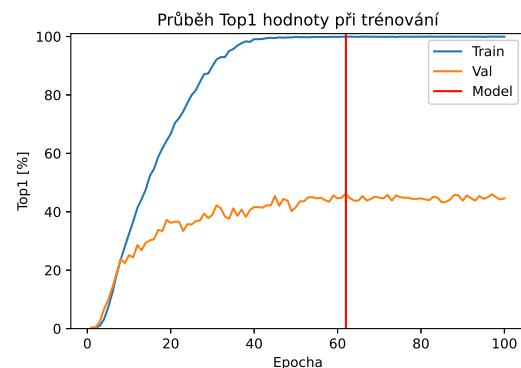
Obrázek 28
Průběh loss funkce při trénování,
(segmentační data, pokus **4.1**)



Obrázek 29
Průběh Top1 hodnoty při trénování,
(segmentační data, pokus **4.1**)



Obrázek 30
Průběh loss funkce při trénování - data
optického toku,
(data optického toku, pokus **2.1**)



Obrázek 31
Průběh Top1 hodnoty při trénování - data
optického toku,
(data optického toku, pokus **2.1**)

7.6 Porovnání modelů

Díky experimentu z kapitoly 7.5 byly získány natrénované modely. Jako nejlepší modely jsme určili ty, které získaly největší *Top1* rozpoznávací přesnost na validačních datech. Simulujieme tím to, že testovací data jsou pro nás neviděná náhodná data, podle kterých nemůžeme model nastavovat. Testovací set nám slouží pouze na porovnávání modelů. Shrnutí nejlepších modelů je v Tabulce 12.

Zpracování dat	Top1 [%] (val)	Top5 [%] (val)	Top1 [%] (test)	Top5 [%] (test)
RGB	75.4	91.2	72.4	91.2
Segmentace	44.2	66.2	42.2	66.6
Optický tok	46.2	68.8	44.0	69.1

Tabulka 12
Tabulka nejlepších výsledků

Z tabulky 12 je patrné, že nejlepších výsledků dosáhl model natrénovaný na klasických RGB videích s Top1 přesností 72.4% na testovacích datech. Data reprezentující optický tok a segmentované data měly zhruba o 30% menší *Top1* přesnost na testovacích datech. Pro hlubší analýzu byly činnosti z testovacího množiny dat rozděleny do 4 tříd:

- sportovní činnosti: 651 videí,
- denní aktivity: 456 videí,
- hraní na hudební nástroj: 156 videí
- koníčky a volnočasová aktivita: 237 videí.

Na základě tohoto rozdělení testovací množiny byl proveden test klasifikační přesnosti jednotlivých tříd činností. Záznam testu je v Tabulce 13.

Zpracování dat	Top1 [%] sportovní činnosti	Top1 [%] denní aktivity	Top1 [%] hra na hudební nástroj	Top1 [%] volnočasová aktivita
RGB	75.42	70.61	67.30	70.89
Segmentace	54.07	37.06	21.79	32.91
Optický tok	49.77	39.69	38.46	40.08

Tabulka 13
Porovnání modelů na základě typů činností

Z tabulky je vidět, že všechny tři modely nejlépe rozpoznávají sportovní činnosti. Nejhůře si modely vedou u rozpoznávání hry na hudební nástroje, v této kategorii segmentované data hodně zaostávají. Je to způsobené právě segmentací, kdy je z videa odstraněn důležitý objekt, hudební nástroj (Obrázek 32). To je možné pozorovat i u ostatních činností, kdy



Obrázek 32

Činnost hraní na klarient pro tři druhy zpracování videa,
vlevo RGB, uprostřed segmentace, vpravo optický tok.

člověk interaguje s nějakým předmětem. Tím že jsme provedli sémantickou segmentaci a zjednodušili obrázek, jsme přišli o důležité pixely, které s danou činností souvisí.

Optický tok dobře rozpoznává sportovní činnosti, protože se zde předpokládají velké a rychlé pohyby, ostatní aktivity klasifikuje z tohoto důvodu hůře. Optický tok odstraňuje z videa stacionární objekty, ty můžou s činností úzce souviset a tím potenciálně snižuje správnou klasifikaci. Problém nastává také u videí, které mají velmi pohyblivé pozadí. Na Obrázku 33 je jedno z testovaných videí činnosti trojskok. U tohoto videa se kvůli pohybující se kameře pohybovalo více prostředí než atlet skákající do píska. Kvůli tomu bylo nejvíce zbarveno pozadí jako nejvíce se pohybující objekt. Optický tok toto video nedokázal správně klasifikovat ani v *Top5*, naopak RGB a segmentační model správně rozpoznali činnost videa v *Top1*.



Obrázek 33

Činnost trojskok pro tři druhy zpracování videa,
vlevo RGB, uprostřed segmentace, vpravo optický tok.

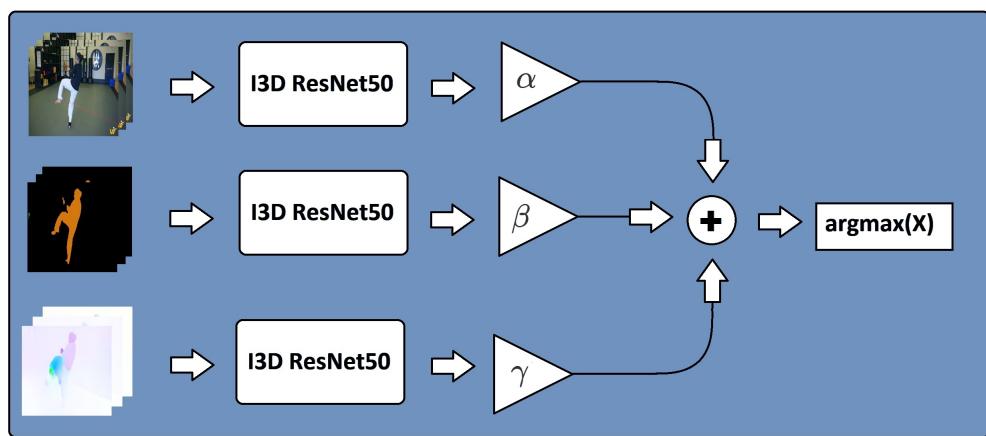
Ukázalo se, že RGB model má nejlepší výsledky a je poměrně konzistentní napříč všemi činnostmi. Zjednodušováním videa segmentací, nebo optickým tokem přicházíme o důležité informace uložené v pixelech původního videa.

7.7 Spojení modelů

Myšlenka spojení modelů vychází z představy, že každý model dokáže správně klasifikovat určité činnosti. Systémy se budou spojovat pomocí součtu *softmax* matic, které získáme na konci sítě. K *softmax* maticím byly ještě přidány váhy. Princip takto spojených modelů pojmenujeme **spojení modelů A**. Celé toto spojení lze matematicky zapsat pomocí (rovnice (9)):

$$X = \alpha X_{rgb} + \beta X_{segmentace} + \gamma X_{opticky_tok}, \quad (9)$$

kde X_{rgb} , $X_{segmentace}$ a $X_{opticky_tok}$ jsou *softmax* matice příslušných modelů. Jsou reprezentovány vektorem o velikosti 500. Každá buňka *softmax* matice nese informaci o pravděpodobnosti příslušné třídy. X je pak jejich součet. Tento vztah je ještě blokové znázorněn na Obrázku 34.



Obrázek 34
Blokové schéma spojení modelů A

Byly otestovány základní konfigurace hodnot α , β a γ . Bylo zaznamenáno, jak se při těchto konfiguracích budou měnit klasifikační přesnosti na testovacím datasetu, to je možné vidět v Tabulce 14.

α	β	γ	Top1 [%] (test)	Top3 [%] (test)	Top5 [%] (test)
1.0	1.0	1.0	72.73	85.53	89.80
1.0	0	1.0	72.47	86.20	90.27
1.0	1.0	0	71.73	85.07	89.33
0	1.0	1.0	49.60	68.20	75.47

Tabulka 14
Tabulka přesnosti rozpoznavání pro různé parametry α, β, γ
spojení modelů A

Z tabulky 14 je patrné, že při parametrech $\alpha = 1$, $\beta = 1$, $\gamma = 1$ a po sečtení *softmax* matic všech třech modelů jsme dosáhli téměř stejného výsledku jako při klasifikování samotným

RGB modelem ($Top1_{RGB} = 72.4\%$). Ve snaze nalézt nejlepší nastavení α , β a γ , které by co nejvíce zvýšilo klasifikační přesnosti, byl proveden test na validačních datech. Pomocí 3 cyklů a s krokem parametrů 0.01 byly nalezeny nejlepší parametry α , β a γ ve smyslu $Top1$ přesnosti na validačních datech.

	α	β	γ	Top1 [%]	Top3 [%]	Top5 [%]
Test na validační setu	0.94	0.61	0.39	77.40	87.60	91.00
Test na testovacím setu	0.94	0.61	0.39	75.00	86.60	90.27

Tabulka 15

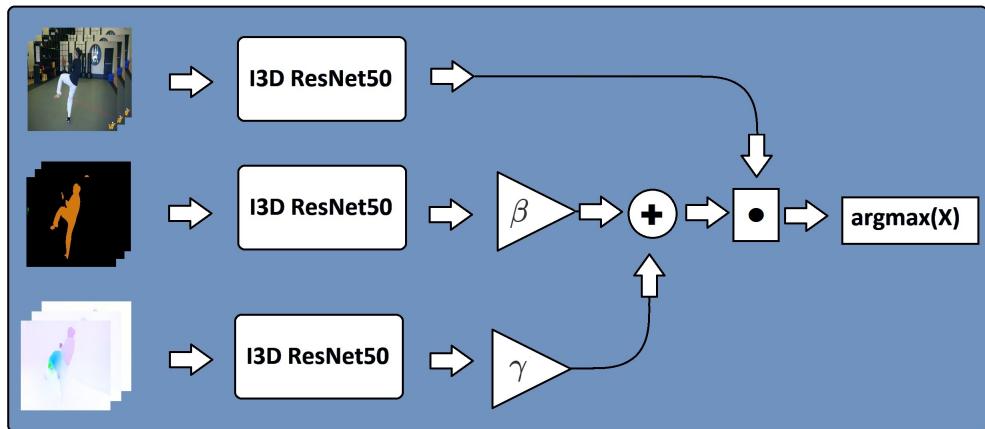
Tabulka přesnosti rozpoznávání pro různé parametry α , β a γ
spojení modelů A

Z tabulky 15 je vidět, že nalezením nejlepší konfigurace parametrů jsme dokázali zvýšit rozpoznávání na testovacím setu zhruba o 2.5% $Top1$ oproti základnímu nastavení $\alpha, \beta, \gamma = 1$.

Byla vyzkoušená další konfigurace, jak je možné výstupy z neuronových sítí spojit. Tuto konfiguraci pojmenujme například **spojení modelů B**. Je založená na základě součtu *softmax* matic modelu segmentačního a optického toku. Pak je tento součet vynásobený se *softmax* maticí RGB modelu, matematicky jako (rovnice 10):

$$X = X_{rgb} \cdot (\beta X_{segmentace} + \gamma X_{opticky_tok}), \quad (10)$$

tento postup je blokově znázorněn na Obrázku 35.



Obrázek 35
Blokové schéma spojení modelů B

Opět byl proveden test na validačním setu pomocí 2 cyklů a krokem 0.01 parametrů β a γ . Byly zaznamenány parametry při nejlepší $Top1$ klasifikaci a poté byl s nimi proveden test na testovací množině dat, to je možné vidět v Tabulce 16.

	β	γ	Top1 [%]	Top3 [%]	Top5 [%]
Test na validační setu	0.35	0.86	79.60	90.80	93.40
Test na testovacím setu	0.35	0.86	77.40	90.20	92.87

Tabulka 16

Tabulka přesnosti rozpoznávání pro různé parametry β a γ spojení modelů B

Touto konfigurací se nám podařilo zvýšit o zhruba 5% *Top1* klasifikaci na testovacích datech oproti samotnému RGB modelu ($Top1_{RGB} = 72.4\%$). Abychom získali informace v jaké třídě proběhlo zlepšení nově vzniklých spojených modelů, bude provedena hlubší analýza na testovacím datasetu s rozdelenými daty do skupin sportovní činnosti, denní aktivity, hraní na hudební nástroje a volnočasová aktivita. Výsledek hlubší analýzy je zaznamenán v Tabulce 17.

Zpracování dat	Top1 [%] sportovní činnosti	Top1 [%] denní aktivity	Top1 [%] hra na hudební nástroj	Top1 [%] volnočasová aktivita
RGB	75.42	70.61	67.30	70.89
Segmentace	54.07	37.06	21.79	32.91
Optický tok	49.77	39.69	38.46	40.08
Spojení modelů 1	78.34	72.80	71.15	72.57
Spojení modelů 2	79.57	75.00	82.05	73.00

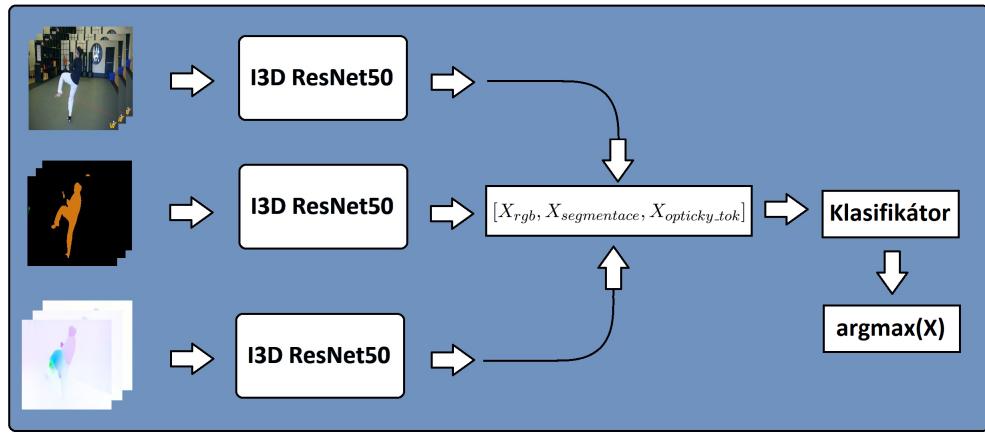
Tabulka 17

Porovnání modelů na základě typů činností po přidání spojení modelů A a B

Z Tabulky 17 je vidět, že se pro spojení modelů A a B zvýšila *Top1* u všech skupin činností oproti samotnému RGB modelu. U spojení modelů B byl zaznamenán velký nárůst *Top1* u hry na hudební nástroje o zhruba 15%. Ze všech modelů si nejlépe ve všech skupinách činností vedl právě spojení modelů B.

7.8 Využití klasifikátorů

Další způsob, kterým se pokusíme navýšit *Top1* klasifikaci je s využitím klasifikátorů. Předpokládá se, že *softmax* matici příslušných modelů nesou informace o podobných třídách. Například pokud rozpoznáváme činnost střela míčem při fotbale, očekáváme že model přiřadí malé věrohodnosti i podobným činnostem jako je třeba dribbling s míčem, nebo příhrávka. Sloučením *softmax* matic z RGB, optického toku a segmentačního modelu můžeme získat jeden vektor o velikosti 1500, který bude představovat příznakový vektor pro klasifikátor. Tento vektor bude vstupovat do klasifikátoru a klasifikátor na základě jeho hodnot přiřadí pravděpodobnost natrénovaným třídám. Tento proces popisuje blokové schéma na Obrázku 36.



Obrázek 36
Blokové schéma klasifikátoru

Problémem je otázka, z jakých dat budeme klasifikátor trénovat. Můžeme využít data z validační množiny. Těchto dat je ale příliš málo, ve validační množině je přesně 500 videí, pro každé video tedy připadá 1 video. I přes malý počet trénovacích dat jsme zkusili natrénovat *SVM* (*Support Vector Machine*) klasifikátor a *KNN* (*K-Nearest Neighbors*). Natrénovaný klasifikátor byl pak otestován na testovacím datasetu. Výsledek je zaznamenán v Tabulce 18.

Klasifikátor	Top1 [%] (test)	Top3 [%] (test)	Top5 [%] (test)
LinearSVC(C=1)	54.8	71.4	76.6
LinearSVC(C=0.5)	54.0	71.4	76.5
LinearSVC(C=0.1)	52.4	71.3	76.7
KNN(n=1)	43.9	44.0	44.3

Tabulka 18
Výsledek testu klasifikátorů trénované na validační množině

Malý počet trénovacích dat se projevil na klasifikaci, kde nejlepší klasifikátor dosáhl hodnot 54.8% Top1. Při snaze o navýšení trénovacích dat klasifikátorů byly zváženy i data z nejčetnější trénovací množiny. Zde nastává problém ten, že modely sítě I3D ResNet50 se na těchto datech učila a modely se při klasifikování trénovacích dat často přiblížily 100% *Top1* přesnosti (viz Obrázky 27, 29, 31 z kapitoly 7.5). Přesto byla trénovací množina klasifikátorů navýšena o trénovací data *HAA500* a tím se klasifikátory trénovaly na 8500 datech, pro každou třídu 17 videí. Po natrénování jsme klasifikátory opět otestovaly na testovacích datech, výsledek pokusu je v Tabulce 19.

Klasifikátor	Top1 [%] (test)	Top3 [%] (test)	Top5 [%] (test)
LinearSVC(C=1)	71.5	83.4	86.5
LinearSVC(C=0.5)	70.5	83.8	87.5
LinearSVC(C=0.1)	69.9	84.3	88.1
KNN(n=1)	60.1	60.2	60.5
KNN(n=3)	56.3	65.0	65.0

Tabulka 19

Výsledek testu klasifikátorů trénované na trénovací a validační množině

Z Tabulky 19 je vidět, že po navýšení trénovací množiny jsme nedosáhli lepších výsledku než které dosáhl samotný RGB model ($Top1_{RGB} = 72.4\%$). Je to pravděpodobně zapříčiněné trénováním klasifikátoru na stejných datech, na kterých se učila neuronová síť. Kdybychom měli velké množství validačních dat, možná bychom dosáhli lepších výsledků.

8 Závěr

Diplomová práce se zabývala úlohou rozpoznávání činností z videa. Pro tuto úlohu byla zvolena architektura *I3D ResNet50*, které transformuje konvoluční jádra modelu *ResNet50* do trojrozměrné podoby. Počáteční hodnoty sítě *I3D ResNet50* byly využity předtrénované váhy na datastu *Kinetics400*. V práci bylo testováno jak ovlivně kvalitu trénování předtrénované váhy oproti náhodně inicializovaným. Výsledek tohoto experimentu ukázal, že při použití předtrénovaných vahách byl natrénován kvalitnější model, který klasifikoval testovací videa s 22% větší *Top1* přesnosti. V práci se také testovalo, jak model ovlivní počet zvolených klíčových snímků. Při experimentu bylo dokázáno, že natrénovaný model s větším počtem klíčových snímků produkuje lepší klasifikační výsledky na testovacím setu.

Hlavní dataset se kterým byly prováděny pokusy byl zvolen *HAA500*, který má 500 odlišných tříd činností s celkovým počtem 10000 videí. Na tyto data byla aplikována segmentace a optický tok. Bylo zkoumáno, jestli zjednodušením videa nezískáme lepší výsledky. Byly ale získány horší výsledky oproti neupraveným videím. Zjednodušením videa přicházíme o důležité informace uložené v pixelech a tím se snižuje potenciál modelu I3D ResNet50. Výsledky tohoto experimentu jsou zaznamenány v Tabulce 20.

Model	Top1 [%] (test)	Top5 [%] (test)
RGB	72.4	91.2
Segmentace	42.2	66.6
Optický tok	44.0	69.1
Spojení A	75.0	90.27
Spojení B	77.4	92.87

Tabulka 20
Tabulka nejlepších výsledků

Zjistilo se, že modely natrénované na odlišných datech rozpoznávají videa odlišně. Toho bylo využito při spojení výstupních *softmax* matic modelů. Bylo navrženo spojení modelů pomocí součtu a každému modelu byla přiřazena váha. Ideální nastavení vah modelů bylo zjištěno na validační setu a provedl se klasifikační test na testovacích datech. V Tabulce 20 byl pojmenován **spojení A**. Byla navržena ještě jedna metoda spojení *softmax* matic pomocí součtu segmentovaného a modelu optického toku a následném vynásobením RGB modelu, ten je v Tabulce 20 pojmenován jako **spojení B**. Byl proveden pokus o natrénování klasifikátorů, který by ze tří modelů zvýšily klasifikační přesnosti. Zde bylo naraženo na problém malého počtu trénovacích dat.

Nejlepší výsledky ve smyslu *Top1* přesnosti na testovacích datech má spojení modelů A. Oproti samostatnému modelu pracujícího s klasickými RGB videi má o 5% vyšší přesnost. K jeho použití je ale potřeba mít k dispozici segmentované data a data reprezentující optický tok a zisk těchto dat je výpočetně náročný. Nejlepší model ve smyslu rychlost výpočtu a přesnost se tedy jeví samostatný model, který zpracovává standardní videa.

Experimenty s modelem *I3D ResNet50* byly prováděny v programovacím jazyce *Python*. Programovací kód je poskytnutý prostřednictvím služby *GitHub* a je možné jej nalézt na odkaze: https://github.com/kleckaa/action_recognition.

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*. 2017. DOI: 10.48550/ARXIV.1705.06950. URL: <https://arxiv.org/abs/1705.06950>.
- [13] 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>.
- [14] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. zpr. 2009.
- [15] Y. LeCun et al. “Backpropagation Applied to Handwritten Zip Code Recognition”. In: *Neural Computation* 1.4 (1989), s. 541–551. DOI: 10.1162/neco.1989.1.4.541.
- [16] 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>.
- [17] 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>.

- [18] 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>.
- [19] 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>.
- [20] 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>.
- [21] Mrinal Tyagi. *Image segmentation nbsp;; Part 1*. 2021. URL: <https://towardsdatascience.com/image-segmentation-part-1-9f3db1ac1c50>.

10 Seznam obrázků a tabulek

Seznam obrázků

1	CIFRA-10 trénovací a testovací error původní obrázek: https://arxiv.org/pdf/1512.03385.pdf [10] (strana 1)	7
2	Residuální blok	7
3	Bottleneck blok	8
4	Blok A	10
5	Blok B	10
6	Blokově znázorněná architektura I3D ResNet 50	11
7	Kroková strategie <i>lr</i>	13
8	Kosínová strategie <i>lr</i>	13
9	Cyklická strategie <i>lr</i> (jeden cyklus)	14
10	Strategie restartováním <i>lr</i>	14
11	Distribuce tříd	16
12	<i>Kinetics-400</i> snowkitting	16
13	<i>Kinetics-400</i> sfoukávání svíček	16
14	<i>Kinetics-400</i> bobování	17
15	<i>HAA500</i> fotbalový dribbling	17
16	<i>HAA500</i> fotbalová hlavička	17
17	<i>HAA500</i> fotbalový zákrok brankáře	17
18	Ukázka segmentace - hod diskem	19
19	Ukázka segmentace - pád na kole	19
20	Ukázka segmentace - hod diskem	20
21	Ukázka segmentace - baseball odpal míčku	21
22	Ukázka augmentace (pád na kole)	22
23	Průběh loss funkce při trénování, náhodně inicializované váhy	26
24	Průběh loss funkce při trénování, váhy z <i>Kinetics-400</i>	26
25	Ukázka činnosti salto vzad pro tři druhy zpracování videa	27
26	Průběh loss funkce při trénování, (rgb data, pokus 5.2)	33
27	Průběh Top1 hodnoty při trénování, (rgb data, pokus 5.2)	33
28	Průběh loss funkce při trénování, (segmentační data, pokus 4.1)	33
29	Průběh Top1 hodnoty při trénování, (segmentační data, pokus 4.1)	33
30	Průběh loss funkce při trénování - data optického toku, (data optického toku, pokus 2.1)	33
31	Průběh Top1 hodnoty při trénování - data optického toku, (data optického toku, pokus 2.1)	33
32	Činnost hraní na klaint pro tři druhy zpracování videa, vlevo RGB, uprostřed segmentace, vpravo optický tok.	35

33	Činnost trojskok pro tři druhy zpracování videa, vlevo RGB, uprostřed segmentace, vpravo optický tok.	35
34	Blokové schéma spojení modelů A	36
35	Blokové schéma spojení modelů B	37
36	Blokové schéma klasifikátoru	39

Seznam tabulek

1	Počet dat	15
2	Počet dat	18
3	Příklad testování úspěšnosti klasifikace videí	23
4	Počet dat subsetu <i>HAA500</i>	24
5	Závislost přesnosti rozpoznávání a počtu klíčových snímků	25
6	Porovnání rozpoznávací přesnosti modelů s předtrénovanými a náhodně iniciálizovanými vahami	26
7	Experimentálně zjištěné hodnoty μ a σ	27
8	Hodnoty Top1 při různých normalizacích dat	28
9	Trénování I3D ResNet50 na rgb datech	30
10	Trénování I3D ResNet50 na segmentačních datech	31
11	Trénování I3D ResNet50 na datech optického toku	32
12	Tabulka nejlepších výsledků	34
13	Porovnání modelů na základě typů činností	34
14	Tabulka přesnosti rozpoznávání pro různé parametry α, β, γ spojení modelů A	36
15	Tabulka přesnosti rozpoznávání pro různé parametry α, β, γ spojení modelů A	37
16	Tabulka přesnosti rozpoznávání pro různé parametry β a γ spojení modelů B	38
17	Porovnání modelů na základě typů činností po přidání spojení modelů A a B	38
18	Výsledek testu klasifikátorů trénované na validační množině	39
19	Výsledek testu klasifikátorů trénované na trénovací a validační množině	40
20	Tabulka nejlepších výsledků	41