FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA
KYBERNETIKY

Bakalářská práce

Automatická detekce typu události ve videozáznamu fotbalového utkání

Václav Kučera

PLZEŇ 2024

FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA
KYBERNETIKY

Bakalářská práce

Automatická detekce typu události ve videozáznamu fotbalového utkání

Václav Kučera

Vedoucí práce
Ing. Zdeněk Krňoul, Ph.D.

PLZEŇ 2024

Citace v seznamu literatury:
KUČERA, Václav. *Automatická detekce typu události ve videozáznamu fotbalového utkání*. Plzeň, 2024. Bakalářská práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra kybernetiky. Vedoucí práce Ing. Zdeněk Krňoul, Ph.D.

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2023/2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Václav KUČERA**
Osobní číslo:	**A21B0428P**
Studijní program:	**B0714A150005 Kybernetika a řídicí technika**
Specializace:	**Umělá inteligence a automatizace**
Téma práce:	**Automatická detekce typu události ve videozáznamu fotbalového utkání**
Zadávající katedra:	**Katedra kybernetiky**

Zásady pro vypracování

1. Prostudujte stávající metody počítačového vidění vhodné pro analýzu videozáznamu fotbalových zápasů.
2. Zaměřte se na problematiku strojového učení pro detekci a sledování a na metody hlubokého učení.
3. Navrhněte vlastní postup řešení a implementujte metodu v programovacím jazyce Python.
4. Zvolte vhodná data pro vypracování a vyhodnocení experimentu.
5. Definujte experiment a kritérium, které ohodnocuje správnost navrženého řešení, a vyhodnoťte výsledky.

Rozsah bakalářské práce: **30-40 stránek A4**
Rozsah grafických prací:
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

Sonka M., Hlavac V., Boyle R.: Image Processing, Analysis, and Machine Vision, Third Edition, Thomson, 2008

Vedoucí bakalářské práce: **Ing. Zdeněk Krňoul, Ph.D.**
Katedra kybernetiky

Datum zadání bakalářské práce: **17. října 2023**
Termín odevzdání bakalářské práce: **20. května 2024**

Západočeská univerzita v Plzni
L.S.
Fakulta aplikovaných věd

Doc. Ing. Miloš Železný, Ph.D.
děkan

Doc. Dr. Ing. Vlasta Radová
vedoucí katedry

V Plzni dne 17. října 2023

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného akademického titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Západočeská univerzita v Plzni má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Plzni dne 20. května 2024

. . . . . .
Václav Kučera

V textu jsou použity názvy produktů, technologií, služeb, aplikací, společností apod., které mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

Abstrakt

Tato bakalářská práce se zabývá automatickou detekcí událostí ve videozáznamech fotbalových utkání. Jejím hlavním cílem je analyzovat a zpracovat videozáznamy za účelem nalezení klíčových událostí, jako jsou góly, rohové kopy, fauly a další významné momenty zápasu. V úvodu práce je představena motivace a využití této úlohy, včetně přehledu současných metod a technologií, které se pro detekci událostí ve fotbalu využívají. Následně se práce věnuje metodám strojového a hlubokého učení, které jsou klíčové pro analýzu videozáznamů. V praktické části je popsána aplikace použitého modelu na videozáznamy zápasů nižších českých fotbalových soutěží, včetně přípravy vlastních dat, trénování modelu, provedení experimentů a vyhodnocení jejich výsledků. Závěrem jsou diskutována možná vylepšení, nedostatky a další směry výzkumu v této oblasti.

Abstract

This bachelor thesis deals with automatic event detection in football match videos. Its main objective is to analyze and process video recordings in order to find key events such as goals, corners, fouls and other significant moments of the match. The motivation and applications of this task are presented in the introduction of the paper, including an overview of current methods and technologies used for event detection in football. Subsequently, the thesis discusses machine and deep learning methods that are key to video analysis. The practical part describes the application of the used model to video recordings of matches of lower Czech football competitions, including preparation of own data, training of the model, conducting experiments and evaluation of their results. Finally, possible improvements, shortcomings and further research directions in this area are discussed.

Klíčová slova

detekce událostí • analýza fotbalového zápasu • neuronová síť • CALF • SoccerNet

Poděkování

Na tomto místě bych rád poděkoval panu Ing. Zdeňku Krňoulovi, Ph.D. za odborné vedení práce, cenné rady, zájem o danou oblast a trpělivost při konzultacích a během celého procesu tvorby této práce. Dále bych chtěl poděkovat své rodině a přátelům za jejich podporu a povzbuzení při psaní této práce.

Výpočetní zdroje byly poskytnuty v rámci projektu e-INFRA CZ (ID:90254), podpořeného Ministerstvem školství, mládeže a tělovýchovy České republiky.

Obsah

Úvod 1

Současná doba je spojena s využíváním informačních technologií prakticky ve všech možných odvětvích. Zjednodušují lidem práci a značně ulehčují některé úlohy. Nejinak je tomu ve světě sportu. Informační technologie mají ve sportu obrovský potenciál a na jejich využití se můžeme podívat více pohledy.

Jedním z nich je neustálá potřeba zpracování nepřeberného množství videozáznamů s cílem extrahovat důležité údaje a statistiky pro trenéry, samotné hráče a fanoušky. Nic, co by nemohl udělat sám člověk, avšak s rostoucím počtem takových záznamů a uvážením, že lidská práce něco stojí a je mnohem pomalejší než počítač, je automatizace tohoto procesu nevyhnutelná.

Spojení sportu s informačními technologiemi je velkým přínosem také pro analýzu různých taktik a strategií, a to jak v týmových sportech, tak i v těch individuálních. Sportovci tak mají na dosah veškeré informace a data týkající se jejich výkonu a mohou je použít pro své vlastní zdokonalení nebo jako taktickou přípravu na své budoucí soupeře. Stále však platí, že jediná správná cesta ke zlepšení individuálního výkonu vede přes přirozený talent a tvrdý trénink.

Sportovní analýza získává v posledních letech stále větší důležitost a v souvislosti s tím se v této práci budeme zabývat nejpopulárnějším týmovým sportem, jímž je fotbal.

1.1 Analýza fotbalového zápasu

Fotbal je označován jako nejvíce populární a nejvíce rozšířený kolektivní sport na světě. Zásluhu na tom má hlavně jeho jednoduchost a možnost hrát v podstatě kdekoliv s kýmkoliv, protože jediná věc, která je potřeba ke hraní, je míč. Analýza fotbalových zápasů má ale smysl až na vyšších úrovních, kdy samotní hráči jsou již natolik herně vyspělí, že jsou schopni poznatky získané z minulých odehraných zápasů využít pro zlepšení svého i týmového herního projevu. Nemluvíme teď pouze o profesionálních soutěžích. Rozbor utkání lze provádět i na amatérské úrovni, ale nenabývá takové důležitosti jako v případě profesionálních zápasů. s rostoucí úrovní zápasů tedy roste i potřeba pořizovat jejich videozáznam právě pro účely analýzy.

Fotbalový zápas je dynamickým a komplexním prostředím, ve kterém se odehrává velké množství událostí a dochází k interakcím jak mezi hráči a míčem (střely, vhazování, výkopy), tak i mezi samotnými hráči (fauly). Porozumění těmto událostem a jejich kontextu je klíčové pro vylepšení taktiky týmu a individuálního výkonu hráčů. Získané statistiky, jako držení míče, střely, karty, fauly a především góly slouží jako podklad pro měření výkonosti daného hráče nebo týmu a umožňují vytvořit si obrázek o průběhu utkání, aniž by ho fanoušek musel vidět. Zkušení trenéři jsou schopní v záznamu najít důvody neúspěchu svého týmu i způsoby jak výkon mužstva vylepšit. Umí přečíst taktické vzorce a styl hry svého soupeře a připravit tak nejlepší strategii pro svůj tým na další utkání. s tím souvisí i výběr hráčů, kteří se nejlépe hodí pro zvolenou taktiku a které trenér do zápasu nasadí. Každý z hráčů může mít pak ještě individuální taktické pokyny, které si má plnit. Obecně je pro trenéra těžké všechny informace a poznatky zachytit od postranní čáry. Navíc postavení kamery ve větší výšce nad hřištěm poskytuje úplně novou perspektivu a umožňuje vidět situace jinak, než se na první pohled zdá.

Je tedy zřejmé, že fotbal je po taktické stránce poměrně složitý sport a zapojování moderních technologií je pro jeho analýzu vhodné. Zhlédnutí záznamu na velké obrazovce s možností zastavit ho a popsat v klidu danou situaci je pro hráče mnohem jednodušší k pochopení, než stejnou situaci vysvětlit pomocí diagramu na tabuli.

Další faktor je, že nejenom trenéři a hráči, ale i fanoušci většinou nechtějí sledovat celé 90 minut dlouhé utkání a rádi by se podívali pouze na ty události, které jsou zajímavé a především významné pro průběh utkání a ke kterým by se mohli vracet a znovu si je přehrávat. Je proto třeba tyto situace ve videozáznamu vyhledat, uložit jejich pozici a zařadit je do příslušné kategorie jako jsou gól, vhazování, rohový kop, faul, penalta apod. Takové zpracování záznamu utkání lze dělat ručně, avšak zabere to spoustu času, a proto se nabízí využít metod počítačového vidění pro detekci těchto událostí s cílem tento proces značně zrychlit a automatizovat.

Byla by škoda nevyužít znalosti o nalezených událostech v zápasu pro další úlohy. Na automatickou detekci událostí tak může navazovat automatická generace krátkého sestřihu, který v pár minutách obsáhne všechny klíčové události zápasu, především góly. Spojením s jazykovým modelem můžeme nechat vygenerovat krátký report o zápasu. Získané statistické údaje o zápasu můžeme porovnávat mezi jednotlivými týmy a je možné tak odhadovat výsledky zápasů na základě formy, počtu vytvořených šancí, vstřelených gólů atd. Důležitým předpokladem pro tyto další úlohy je však přesnost samotné automatické detekce.

1.2 Struktura a náplň práce

V této práci se budeme zabývat automatickou detekcí událostí ve videozáznamech fotbalových utkání. Na začátku si řekneme něco o motivaci a využití úlohy detekce

událostí ve fotbalu a zmíníme se o současných technologiích a přístupech, které se na tuto úlohu zaměřují. Dále se zaměříme na problematiku strojového učení a na metody hlubokého učení pro detekci a sledování objektů a představíme si stávající metody počítačového vidění, které jsou vhodné pro analýzu videozáznamů fotbalových zápasů. V poslední části se budeme věnovat praktické aplikaci detekce událostí. Popíšeme si fungování použitého programového kódu a jeho modifikací a úpravou trénovacích parametrů se pokusíme aplikovat model natrénovaný na záznamech z televizních vysílání na záznamy zápasů týmů hrajících převážně krajský přebor. Nakonec provedeme několik experimentů, vyhodnotíme jejich výsledky a prodiskutujeme možná vylepšení nebo nedostatky.

Detekce událostí ve fotbalu — 2

V této kapitole si detailně popíšeme úlohu detekce událostí ve fotbalu. Řekneme si něco více o využití a motivaci této činnosti. Seznámíme se se dvěma sportovními analytickým nástroji, jejichž součástí je kamera pro zachycení záznamu a software pro automatické zpracování a analýzu. Dále si detailně popíšeme charakter vstupních a výstupních dat vhodných pro úlohu automatické detekce. V neposlední řadě si představíme nejlepší současné přístupy k řešení automatické detekce ve fotbalu.

2.1 Motivace a využití

Hlavní motivací pro úlohu automatické detekce je urychlení procesu hledání událostí a zaznamenávání důležitých statistik zápasu. Automatizace těchto úloh může výrazně snížit časovou náročnost a náklady spojené s manuálním označováním a sběrem dat. Cílem takového automatického zpracování záznamu by pak bylo rozpoznat vybrané události v zápasu, označit jejich pozici ve videu a rozdělit je do příslušných kategorií.

V úvodu jsme krátce zmínili, jak lze s takto anotovanými událostmi dále nakládat. Ve spojení se softwarem, který umožňuje dynamicky vybírat nalezené události a přepínat mezi nimi, se můžeme bavit o silném nástroji pro analýzu taktiky týmu. Hráči a trenéři neztrácí čas, který by mohli věnovat fyzické přípravě, hledáním jedné konkrétní situace. Takový software často poskytuje i další možnosti pro pohodlný rozbor utkání. Jsou to funkce pro uložení oblíbených událostí, přidání popisků k jednotlivým událostem nebo zobrazení statistik. Součástí podobného softwaru velmi často bývá i nástroj pro prezentaci, díky kterému lze do záznamu kreslit různé tvary, označovat hráče, či jiným způsobem graficky doprovodit danou situaci pro její lepší pochopení.

S dalším využitím automatické detekce událostí se můžeme setkat v úloze vytváření sestřihů důležitých a pro diváka atraktivních událostí v zápasu. V angličtině se pro takové sestřihy používá označení highlights. Délka takového sestřihu se pohybuje v řádech jednotek minut a nejčastěji se v něm objevují góly a jiné podobně

významné momenty zápasu (červená karta, neproměněná penalta). Názorným příkladem může být sestřih zápasu čtvrtfinále Ligy mistrů mezi týmy Real Madrid a Manchester City [1]. Sestřihy jsou v dnešní době velmi populární, neboť pro běžného fanouška není reálné, aby sledoval všechny 90 minut dlouhé zápasy. Často se objevují pod internetovými sportovními články informujícími o výsledku a průběhu daného utkání. V tomto spojení si může fanoušek udělat docela ucelený obrázek o průběhu daného zápasu, aniž by ho musel vidět na vlastní oči.

V současné době do fotbalu nemalou měrou zasahuje videorozhodčí (VAR - angl. Video Assistant Referee), který má za úkol zkoumat ofsajdová postavení hráčů, velmi tvrdé zákroky a také možné fauly a hraní rukou v pokutovém území, po kterých by následovala penalta. Často komunikace hlavního rozhodčího s videorozhodčím trvá několik desítek sekund. V takovém případě by se spolehlivý systém pro detekci mohl využít jako podpora při rozhodování a celý proces by se tak urychlil. Největší potenciál má asi ofsajdová technologie [2], která byla úspěšně testována na mistrovství světa v Kataru v roce 2022.

Systém, který by umožňoval automatickou detekci v reálném čase, by bylo možné využít v aplikacích informujících o průběhu utkání, jako je například Livesport [3]. Česká společnost stojící za touto aplikací patří mezi největší světové poskytovatele výsledků, statistik a dalších informací z více než 35 sportů. V současné době se však nedá ještě plně spolehnout na detekci v reálném čase a údaje o zápasu tak stále zadávají zaměstnanci.

Detekce ve fotbalu nabízí velké množství možných využití. Je proto důležité systémy automatické detekce nadále zdokonalovat, aby dosahovaly takové úrovně, že bychom se na ně mohli plně spolehnout.

2.2 Současné technologie

V této sekci si představíme 2 systémy zaměřující se na automatické zaznamenávání, zpracování a analýzu fotbalových utkání. Oba dva jsou v současné době velmi populární a jsou využívány mnoha týmy napříč různými fotbalovými soutěžemi po celém světě.

2.2.1 Systém Veo

Veo je sportovním analytickým nástrojem od dánské společnosti zabývající se technologiemi pro sport [4]. Tato společnost byla založena roku 2015 v Kodani a po pár letech se stala velmi úspěšnou na poli sportovní analýzy. V dnešní době tento systém využívá více než 15000 sportovních týmů ve více než 80 zemích světa a v roce 2022 přesáhl počet veškerých záznamů získaných s tímto systémem hranici jednoho milionu. Veo používají pro nahrávání a analýzu fotbalových zápasů nejenom pro-

fesionální kluby hrající v nejlepších ligách světa, ale také se s ním můžeme setkat v amatérských soutěžích. Mezi zástupce profesionálních týmů můžeme zařadit například Wolverhampton a Burnley působící v anglické Premier League, či LOSC Lille z francouzské Ligue 1. Mezi plzeňské amatérské kluby využívající tento systém patří například TJ Košutka Plzeň nebo Petřín Plzeň.

2.2.1.1 Veo Cam

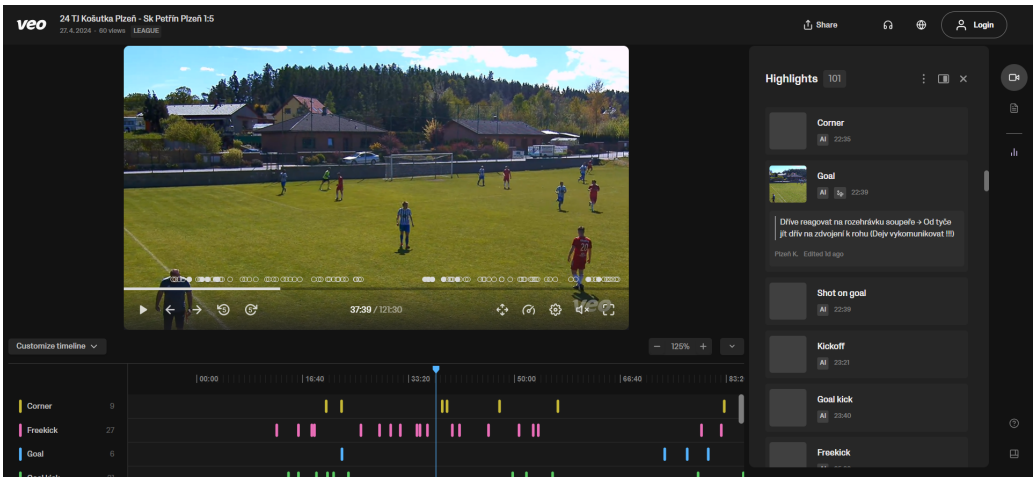
Systém této společnosti je ideálním prostředkem pro pořizování záznamu a jeho následné zpracování díky své jednoduchosti, přesnosti a cenové dostupnosti. Základem je kamerový systém Veo Cam. Na obrázku 2.1 můžeme vidět, že se skládá ze dvou kamer, které dohromady zajišťují úhel záběru 180°, tedy pohled na celé hřiště. Umělá inteligence od společnosti Veo podporující automatické sledování míče na hrací ploše pak z panoramatického záběru vytvoří záznam podobný živému přenosu. Veo Cam tak poskytuje plně automatické nahrávání fotbalového zápasu bez přítomnosti kameramana. Po zápasu se videozáznam jednoduše nahraje na platformu Veo, kde je možné s ním dále pracovat. Nové modernější verze kamery (Veo Cam 2 a Veo Cam 3) podporují i živé vysílání sledovaného zápasu.

Obrázek 2.1: Kamerový systém Veo Cam [4]

2.2.1.2 Veo Editor

Po nahrání zápasu na platformu můžeme ve webovém nástroji Veo Editor začít videozáznam organizovat a analyzovat. Základní balíček umožňuje automatickou detekci událostí v zápasu pomocí vlastní umělé inteligence. Její fungování bohužel není popsáno. Mezi těmito detekovanými událostmi může uživatel libovolně přepínat. Kromě této hlavní funkce má editor i řadu dalších. Mezi ně patří například možnost kreslit do videozáznamu, vytvářet vlastní sestřihy, označovat hráče, psát

komentáře k událostem nebo vybrat vlastní záběr kamery, který je možný právě díky panoramatickému záběru. Pomocí odkazu pak lze nahraný záznam sdílet mezi hráče a trenéry. Obrázek 2.2 ilustruje strukturu webové aplikace, konkrétně výběr nalezených událostí a k nim přidané komentáře.



Obrázek 2.2: Webová aplikace Veo Editor

2.2.1.3 Veo Analytics

Umělá inteligence hraje hlavní roli také v doplňku Veo Analytics. Tento doplněk umožňuje extrahovat pokročilé statistiky a další užitečné údaje o dění v zápasu. Mezi sledované vlastnosti patří například místa střeleckých pokusů, heatmap zobrazující místa, kde probíhalo nejvíce akce v zápasu, 2D vizualizace hrací plochy nebo analýza přihrávek.

Nutno podotknout, že i přes velké množství dat, které má Veo k dispozici ze všech nahraných záznamů, není model umělé inteligence natolik přesný, aby se dalo na něj plně spolehnout. Při pořizování záznamu nastávají občas při hře situace, se kterými si systém při vytváření záznamu neumí poradit. To samé platí i pro nástroje Veo Analytics, kde jsou nepřesnosti mnohem častější.

2.2.2 Systém Panoris

Dalším zařízením pro nahrávání videozáznamu utkání a jeho následnou analýzu je systém Panoris od české společnosti Camvision s.r.o. sídlící v Brně [5]. Dvěma kamerami zachycuje celou plochu hřiště do panoramatického záběru a funguje tak na stejném principu jako Veo. Algoritmy specializující se na sledování míče a hráčů pak určují, jaká oblast v panoramatickém záznamu se má sledovat. Součástí tohoto systému je kromě automatického nahrávání zápasu také nástroj pro analýzu a sdílení

výsledného záznamu, stejně jako v případě jeho konkurence. Systém je poměrně rozšířený ve světě a používají ho jak české týmy (FC Viktoria Plzeň, AC Sparta Praha), tak i zahraniční (SK Rapid Wien, 1. FC Nürnberg).

2.3 Vstupní a výstupní data

Pro automatickou detekci událostí ve videozáznamu fotbalového zápasu jsou jednoznačně nejdůležitější vstupní data. Pro úlohu tohoto typu máme možnost využít dvou různých typů vstupních dat - prostoročasová data nebo videozáznamy. Při použití prostoročasových dat máme k dispozici informace o všech objektech (hráči, míč) na hřišti, konkrétně o jejich pozici v čase. K získání těchto prostoročasových dat se v dnešní době používají bezdrátový senzor v míči a senzory, které mají hráči připevněné při hře na sobě. O využití a zpracování prostoročasových dat v úloze automatické detekce pojednává článek [6].

V naší práci se však budeme zabývat detekcí z videozáznamu. V případě, že se jedná o záznam z televizního vysílání, lze pro detekci využít režii přímého přenosu. To může být výhodou, protože obsah záběrů vybraných režisérem může být použit k rozpoznání událostí (přiblížení, zopakování situace). Dále by se mohla využít grafika zobrazující stav utkání, o čemž pojednává článek [7], nebo také zvukové stopy komentátora a ruchu na hřišti (píšťalka rozhodčího, radování fanoušků po gólu), pokud je videozáznam obsahuje. Takové vylepšení modelu přidáním zvukové stopy do architektury založené pouze na získávání znalostí z videa popisuje článek [8], článek [9] představuje využití multimodálního přístupu pro tvorbu sestřihů v tenise a golfu. Na druhou stranu nám některé události mohou uniknout, protože do záběru kamery se nevejde celá hrací plocha a nemáme tak přehled o veškerém dění na hřišti. Pro získání relativně slušných výsledků automatické detekce jsou však videozáznamy dostačující.

Výsledkem automatické detekce událostí z videozáznamů je v pochopitelně seznam rozpoznaných událostí. U každé takové události potřebujeme mít zaznamenán přesný čas ve videozáznamu a její typ.

2.4 Současné přístupy

Analýza sportovních utkání je aktivní oblastí výzkumu. Existuje mnoho prací, které se zaměřují na získávání informací o objektech na hřišti, výkony hráčů, týmovou taktiku nebo na detekci událostí v zápasech. Následující odstavce představují nejlepší současné přístupy k detekci událostí ve fotbalu.

Morra et al. [10] ve své práci popisují algoritmus pro rozpoznání událostí založený na časových a logických pravidlech s využitím prostoročasových dat, který je schopen ze záznamu extrahovat složité události (např. přihrávky) na základě mnoha

atomických událostí. Jejich výsledky dosahují F-score 0,89 pro přihrávky a 0,81 pro střely. Stejný přístup použili také Khaustov a Mozgovoy [11]. Na základě dat z 5 zápasů dosáhli F-score 0.93 pro úspěšné přihrávky, 0.86 pro neúspěšné přihrávky a 0.95 v případě střel na bránu.

Jia [12] se ve svojí práci zabýval segmentací záběrů z videozáznamů fotbalových zápasů za účelem extrakce důležitých snímků. Po segmentaci provedl sémantickou anotaci těchto záběrů ve formě textu. Klíčové události potom nalezl s využitím kombinace pravidel, skrytého markovského modelu (HMM) a genetického algoritmu. Zaměřil se konkrétně na 3 typy událostí - rohový kop, faul a karta. Výsledný model vyhodnotil s přesností 96,62 % a precizností 98,81 %.

Automatickou detekci událostí pouze z obrazové informace videozáznamu s využitím konvolučních neuronových sítí popsali ve své práci Rongved et al. [13]. Jejich algoritmus používá metodu pohyblivého okénka k prohledání videa a detekuje 3 události - góly, karty a střídání. Model natrénovaný na 3 různých datasetech je schopen detekovat události s nízkou latencí a přesným časem. Přesnost klasifikace dosahuje 88,4 %.

Přístup, se kterým přišli Cioppa et al. [14] ve své práci, je v současné době považován za nejpokročilejší pro úlohu detekce událostí na datasetu SoccerNet [15]. Stejně jako Rongved et al. využívají pro trénování konvoluční neuronové sítě a pro přesnější detekování událostí ve videu uvažují jejich okolní časový kontext. Jejich model dosahuje přesnosti 62,5 % na datasetu SoccerNet podle metriky Average-mAP [15]. Kromě toho v práci také popisují, jak výsledky využívají pro generování sestřihů.

Dataset SoccerNet využívají i Xarles et al. [16]. Ve své současné práci představili model ASTRA založený na architektuře transformátoru, který navrhli tak, aby současně řešili problémy spojené s přesnou lokalizaci událostí, nerovnoměrnou distribucí dat, nepřítomností některých událostí v záběru a nekonzistentním anotováním událostí. Přesnost tohoto modelu na testovacím setu dosahuje 66,82 % Average-mAP.

Strojové učení a neuronové sítě

3

V dnešní době se velmi často skloňuje pojem umělá inteligence (AI - angl. Artificial Intelligence). Pro někoho může být takový pojem zavádějící a může si pod ním představit stroj schopný plně vlastního uvažování. Taková myšlenka nemá daleko od definic. Některé z nich jsou popsány v knize [17]. Všechny se shodují na tom, že umělá inteligence představuje schopnost počítačového systému simulovat lidskou inteligenci a řešit problémy, které by jinak vyžadovali přítomnost člověka. Tyto systémy jsou schopné napodobit lidské kognitivní funkce jako je schopnost vidět, vnímat, rozumět, reagovat na mluvený jazyk, analyzovat nebo se učit. Vyspělá umělá inteligence umí zpracovávat získané informace s okolního prostředí rychle a přesně a díky tomu je užitečná pro různé komplexní úlohy, jako jsou autonomní řízení vozidla, programy pro rozpoznávání obrazu nebo virtuální asistenti.

Jako umělá inteligence se zároveň označuje i obor, který se zabývá tvorbou takových systémů. Strojové učení jako podoblast umělé inteligence je jedním ze způsobů, jak počítač naučit napodobovat myšlení podobné člověku. Cílem této kapitoly je zaměřit se na problematiku strojového učení, neuronových sítí, hlubokého učení a počítačového vidění. Představíme si základní koncepty a jejich využití v praxi.

3.1 Strojové učení

Hlavní disciplínou umělé inteligence je strojové učení (ML - anlg. Machine Learning). S myšlenkou, že počítač lze naprogramovat tak, aby se naučil hrát lepší partii dámy, než jakou dokáže hrát ten, kdo program sám napsal, přišel ve svém článku [18] Artur L. Samuel už v roce 1959. Moderní definice říkají, že strojové učení je proces používání matematických modelů dat, pomocí kterých se počítač bez přímých instrukcí učí a adaptuje. Tato oblast se zaměřuje na vývoj počítačových algoritmů k identifikaci vzorů v datech a tyto vzory se pak používají k vytvoření datového modelu, který dokáže formulovat předpovědi a samostatně se bez přítomnosti člověka rozhodovat. Stejně jako se lidé zlepšují díky větší praxi, tak i strojové učení je postupem času s větším množstvím dat a informací přesnější a efektivnější. Zjed-

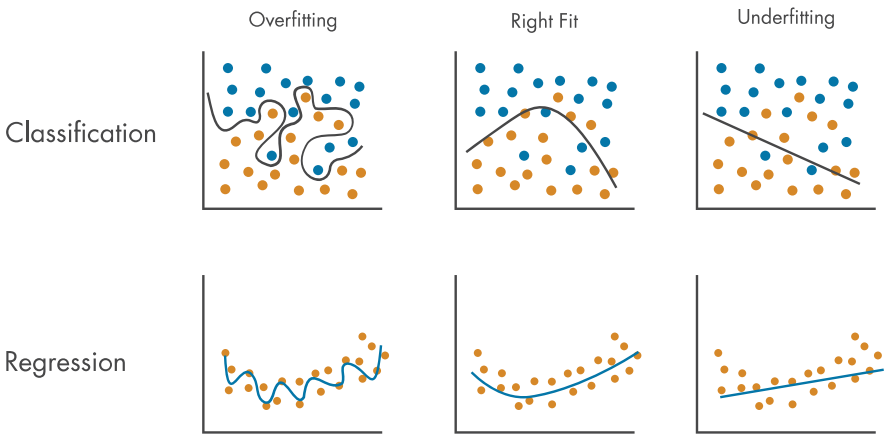
nodušeně řečeno, strojové učení umožňuje počítačům získávat znalosti z dat, učit se z nich a na základě toho dělat vlastní rozhodnutí, aniž by k tomu byly explicitně naprogramovány.

Algoritmy strojového učení lze podle způsobu učení a dostupných dat rozdělit do 3 kategorií - učení s učitelem, učení bez učitele, zpětnovazebné (posilované) učení [19]. Kromě těchto existují ještě způsoby učení zvané semi-supervised a self-supervised. Všechny tyto algoritmy se používají při procesu zvaném trénování. Výsledkem trénování je model. S každým z těchto přístupů se seznámíme v následujících podsekcích.

3.1.1 Učení s učitelem

Učení s učitelem (angl. Supervised Learning) je nejčastějším a nejefektivnějším přístupem v oblasti strojového učení. V praxi to znamená, že pro množinu vstupních dat je definován správný výstup. Tato data pak můžeme označit jako trénovací. Hlavním cílem trénovacího procesu je najít takové parametry, které nejlépe mapují vstup na požadovaný výstup. Po natrénování je model schopný správně predikovat výstupy pro nové neznámé vstupy, které nebyly součástí trénovacího datasetu.

Nevýhodou tohoto přístupu strojového učení jsou nízká flexibilita a vysoké náklady spojené s potřebou získávání velkého množství označených trénovacích dat. Modely natrénované tímto způsobem jsou náchylné k přeučení (angl. overfitting) a mohou se tak příliš přizpůsobit trénovacím datům. Tím ztrácí schopnost zobecnit své znalosti na nová data. Opakem přeučení je nedoučení (angl. underfitting). V tomto případě je model příliš jednoduchý, aby přesně popsal vztah mezi vstupy a výstupy. Během trénovacího procesu nebyl schopen se dostatečně naučit. Výsledkem je velká nepřesnost a nevěrohodná predikce. Obrázek 3.1 graficky ilustruje rozdíl mezi přeučením a nedoučením.



Obrázek 3.1: Overfitting a underfitting [20]

3.1.2 Učení bez učitele

Učení bez učitele (angl. Unsupervised Learning) vyžaduje stejně jako učení s učitelem početnou množinu trénovacích dat, avšak obejde se už bez definovaných výstupů. Algoritmy učení bez učitelem tak musí najít společné vzory a vztahy v trénovacím datasetu samy bez jakékoliv kontroly. Model natrénovaný tímto způsobem pak může objevit spojitosti v trénovacích datech, které nejsou explicitně určené k nalezení. Díky této schopnosti může učení bez učitele produkovat poznatky a údaje, které by jinak mohly zůstat neobjevené. Takový způsob získávání skrytých a potenciálně důležitých informací nazýváme pojmem data mining [21].

3.1.3 Semi-supervised learning

Velmi často se využívá kombinace obou výše popsaných metod nazývaná Semi-supervised Learning. Malá část vstupních trénovacích dat má k dispozici i známý výstup, ale zbylá velká část dat takový výstup nemají. Výhodou tohoto přístupu je možnost rozšířit označená trénovací data velkým množstvím neoznačených dat. Díky tomu se předchází časově náročnému označování trénovacích dat, zvětší se jejich rozmanitost a trénování tak může být efektivnější. Využití tohoto přístupu pro vylepšení detekce hráčů a míče popisuje článek [22].

3.1.4 Zpětnovazebné učení

Zpětnovazebné učení (angl. Reinforcement Learning) je proces učení založený na zpětné vazbě. Základním konceptem je učení se z vlastních chyb a získaných zkušeností. Občas se můžeme setkat s označením posilované učení a největší využití má v agentových technologiích. Agent je autonomní entita, která je schopná interagovat s okolním prostředí. Za své akce je agent buď odměňován nebo penalizován. Cílem agenta je naučit se chovat v daném prostředí tak, aby maximalizoval užitek. Na rozdíl od výše popsaných typů učení, zpětnovazebné učení nevyžaduje trénovací data a učení tak probíhá pouze na základě získaných zkušeností. Zpětnovazebné učení je tak vhodné pro problémy, kdy rozhodnutí v každém kroku mohou ovlivnit budoucí výsledek. S jeho využitím se můžeme setkat například při hraní her nebo v robotice.

3.1.5 Self-supervised learning

Self-supervised learning je způsob učení, kdy je model trénován na neoznačených trénovacích datech. Cílem tohoto přístupu je tak využít struktury nebo vztahy ve vstupních datech k vytvoření smysluplných anotací. Mezi těmito vygenerovanými anotacemi vybere ty s vysokou věrohodností. Takto jsou v prvním kroku připravena označená trénovací data, která se pak využívají jako vstupy v následujících iteracích,

podobně jako při učení s učitelem. Jediný rozdíl je v tom, že po každé iteraci jsou anotace generovány znovu. O tom, co je to iterace, se zmíníme dále v sekci 4.6. Tento způsob učení se využívá zejména v oblastech, které vyžadují velké množství označených dat pro trénování.

3.1.6 Algoritmy strojového učení

Jak již bylo zmíněno výše, účelem strojového učení je naučit počítač nebo stroj dělat vlastní rozhodnutí na základě poskytnutých dat a bez přítomnosti člověka. K tomu se používá celá řada algoritmů, které jsou za učení zodpovědné. Můžeme je rozdělit podle použité metody učení a úlohy, kterou daný algoritmus plní. O takovém rozdělení a popisu jednotlivých typů algoritmů s příklady pojednávají následující podsekce.

3.1.6.1 Úloha klasifikace

Klasifikační algoritmy jsou zástupci strojového učení s učitelem. Hlavní úlohou klasifikace je rozdělit vstupní data do jedné nebo více kategorií na základě trénovací množiny, která obsahuje data se známým zařazením do kategorie. Algoritmus, který implementuje klasifikaci, označujeme jako klasifikátor. Stejně můžeme nazvat i funkci mapující vstupy na výsledné třídy. Článek [23] popisuje 3 typy klasifikace. Prvním z nich je binární klasifikace - data jsou rozdělována do dvou tříd. Třídy mají často povahu "ano a ne", např. určení zda e-mail je nebo není spam. Druhým typem je klasifikace do více tříd. Posledním typem je klasifikace, kdy jednomu vstupu je přiřazeno více tříd. Pro různé druhy klasifikace se používají různé algoritmy, například Random Forest, Naive Bayes, K-Nearest Neighbors, Linear Discriminant Analysis, Decision Tree nebo Support Vector Machine (SVM).

3.1.6.2 Úloha regrese

Regresní algoritmy řadíme také mezi algoritmy učení s učitelem. Regresní analýza se využívá k popisu vztahu mezi závislou a nezávislou proměnnou. Její úlohou je předpovědět hodnotu výstupní spojité proměnné na základě hodnot jedné nebo více vstupních proměnných. Cílem regrese je proložit vstupní data křivkou, která nejlépe odpovídá jejich trendu. Pokud je křivkou přímka (lineární funkce), mluvíme o lineární regresi. Pokud vstupní data prokládáme polynomem, mluvíme o polynomické regresi. Nejčastější metodou pro získání koeficientů přímky v lineární regresi je metoda nejmenších čtverců [24]. Obrázek 3.1 zobrazuje rozdíl mezi klasifikací a regresí.

3.1.6.3 Shlukování

Shlukování, nebo též shluková analýza, je ve strojovém učení metoda využívající algoritmy učení bez učitele. Cílem shlukování je identifikovat a seskupit vzájemně si podobné objekty do shluků takovým způsobem, že objekty ve stejném shluku jsou si sobě v určitém směru podobnější než objekty v ostatních shlucích. Jedná se o iterativní proces objevování informací ve vstupních datech a vyžaduje odborné znalosti a lidský úsudek. Často je totiž třeba vhodně předzpracovat data nebo upravit parametry modelu tak, aby bylo dosaženo požadovaného výsledku. Mezi používané algoritmy patří K-Means, Mean-Shift nebo DBSCAN.

3.1.6.4 Úloha asociace

Dalším typem algoritmů učení bez učitele je asociace. Úlohou asociace je hledání pravidel, vztahů a souvislostí mezi jednotlivými proměnnými ve velmi často rozsáhlém souboru dat. Asociační pravidla se dnes nejvíce využívají pro analýzu uživatelského chování - informace o využívání webu a aplikací, nákupní vzorce zákazníků, systémy doporučování. Práce [23] blíže popisuje nejvíce používané algoritmy - Apriori, Eclat nebo FP-Growth.

3.1.6.5 Redukce dimenze a extrakce příznaků

Velmi často pracujeme s daty, která mají velké množství příznaků nebo dimenzí. To může negativně ovlivnit výkonnost algoritmů a vizualizaci samotných dat. Značně se pak komplikuje jejich zpracování a analýza. Redukce dimenze představuje metody a techniky, které jsou schopny předzpracovat data tak, aby snížili počet příznaků na zvládnutelnou velikost při zachování informace v původních datech. Z datové sady jsou vybírány důležité příznaky a počet méně důležitých se snižuje.

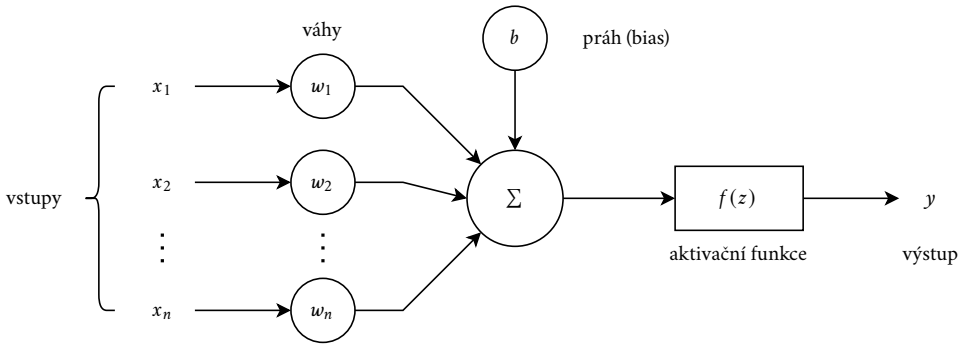
Pro redukci dimenze můžeme využít jak přístupu výběru příznaků, tak přístupu extrakce příznaků. Výběr příznaků je proces, při kterém jsou ve vstupních datech vybírány podmnožiny jedinečných příznaků, které se následně využijí pro natrénování modelu. Méně důležité příznaky jsou eliminovány a snižuje se tak složitost modelu. Cílem extrakce příznaků je snížit počet příznaků ve vstupním souboru dat takovým způsobem, že ze stávajících příznaků se vytvoří nové redukované, které shrnují většinu informací nalezených v původní sadě příznaků. Jednou z technik extrakce příznaků je například analýza hlavních komponent (PCA - angl. Principal Component Analysis) [23], jejíž princip spočívá v lineární transformaci dat do nového souřadnicového systému tak, aby bylo možné snadno identifikovat směry (hlavní komponenty), které zachycují největší variabilitu dat.

Základním rozdílem mezi těmito přístupy je tedy ten, že v případě výběru příznaků je zachovávána podmnožina původních příznaků, zatímco při extrakci příznaků se vytváří příznaky nové.

3.2 Neuronová síť

Umělé neuronové sítě (NN - angl. Neural Network) jsou specifickou třídou algoritmů strojového učení. Cílem neuronových sítí je napodobit způsob, jakým lidský mozek zpracovává informace. Právě samotný lidský mozek byl inspirací pro jejich vytvoření. Základem struktury neuronové sítě jsou malé výpočetní jednotky zvané umělé neurony. Stejně jako mozek obsahuje miliony neuronů a synaptických spojení, tak i neuronové sítě se skládají ze vzájemně propojených umělých neuronů uspořádaných nejčastěji do několika vrstev. Každý z těchto neuronů zpracovává vstupy a vytváří výstup, který může být opět vstupem pro další vrstvy neuronů. V takovém případě hovoříme o dopředné vícevrstvé neuronové síti. Skládá se ze vstupní vrstvy, jedné nebo více skrytých vrstev a výstupní vrstvy. Existují i sítě se zpětnou vazbou. V architekturách těchto sítí je výstup jednoho neuronu přiveden na vstup toho samého (zavádí zpětnou vazbu).

Jak jsme již zmínili, neuronové sítě jsou založeny na vzájemném propojení neuronů. V praxi je každému takovému propojení je přiřazena určitá vlastní váha a práh. Výstup neuronu se aktivuje pokud je suma vyšší než prahová hodnota. Každý neuron může mít libovolný počet vstupů, ale výstup pouze jeden. Propojené neurony si navzájem předávají signály a pomocí aktivačních funkcí je převádějí na výstup. Takové fungování neuronové sítě si představíme na perceptronu - nejjednodušším modelu dopředné neuronové sítě, který v roce 1958 popsal Frank Rosenblatt [25]. Model perceptronu můžeme vidět na obrázku 3.2 a sestává právě z jednoho neuronu.



Obrázek 3.2: Model perceptronu

Vstupy neuronu označíme x_i a tvoří vstupní vektor $\mathbf{x} = [x_1, x_2, ..., x_n]$. Váhy jednotlivých vstupů označíme w_i, vektorově zapsáno $\mathbf{w} = [w_1, w_2, ..., w_n]$. Vstupní

práh označíme b a $f(z)$ je aktivační funkce. V případě perceptronu se jako aktivační funkce často volí bipolární funkce signum nebo Heavisidova funkce. Výstup neuronu je aktivní pokud platí ($\sum_{i=1}^{n} w_i x_i \geq b$) a neaktivní pokud platí ($\sum_{i=1}^{n} w_i x_i < b$). Pro výstup perceptronu pak platí:

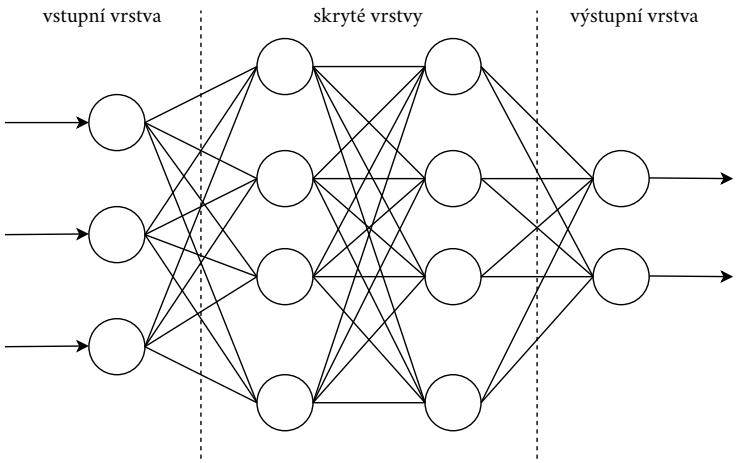
$$y = f(z) = f\left(\mathbf{w}^T \mathbf{x} + b\right) = f\left(\sum_{i=1}^{n} w_i x_i + b\right). \tag{3.1}$$

Model neuronové sítě je charakterizován konkrétním nastavením vah jednotlivých propojení a říká nám, jakým způsobem se mají vstupní vektory mapovat na výstup. Modifikací vah můžeme vytvořit mnoho způsobů, jakým lze vstup transformovat na výstup. Trénování sítě spočívá v nastavení vah propojení takovým způsobem, abychom minimalizovali danou chybovou funkci, která nám určuje velikost rozdílu mezi výstupem sítě a požadovaným výstupem definovaným v trénovací množině. Nejčastěji je trénovací množina předkládána síti opakovaně, natrénování jedné kompletní množiny označujeme jako epocha. Existují dva typy přístupů v předkládání trénovacích dat. Sekvenční přístup (SGD) je založen na okamžité změně vah po předložení jedné trénovací dvojice. V dávkovém přístupu (Batch GD) se změna nastavení provede až po předložení celé jedné trénovací sady.

3.3 Hluboké učení

Hluboké učení (angl. Deep Learning) je oblast strojového učení, která k simulaci složitého fungování a rozhodování lidského mozku využívá hluboké neuronové sítě. Cílem hlubokého učení je natrénovat počítač takovým způsobem, aby zpracovával informace jako člověk. Sítě hlubokého učení se nazývají hluboké neuronové sítě (DNN - angl. Deep Neural Network), protože jejich architektury obsahují mezi vstupní a výstupní vrstvou mnoho skrytých vrstev. Obrázek 3.3 představuje architekturu plně propojené hluboké neuronové sítě se 2 skrytými vrstvami. Každá další vrstva v síti může mít jinou funkci, jejich hlavním cílem je obecně síť optimalizovat a zvýšit její přesnost. Natrénované modely dokáží rozpoznat ve vstupních datech (obrázek, text, zvuk) složité vzorce a na základě nich vytváří přesné predikce a rozhodnutí.

Metody hlubokého učení se uplatňují hlavně při automatizaci úloh, které za normálních okolností vyžadují lidskou inteligenci. Mezi takové úlohy můžeme zařadit například rozpoznávání řeči (angl. Speech Recognition), zpracování přirozeného jazyka (NLP - angl. Natural Language Processing) nebo různé doporučovací systémy. Ve velkém se s hlubokým učením setkáme v oblasti počítačového vidění, kterou si v práci podrobněji představíme později.



Obrázek 3.3: Architektura hluboké neuronové sítě

V rámci této sekce si představíme 4 architektury hlubokých neuronových sítí - vícevrstvý perceptron, konvoluční neuronová síť, rekurentní neuronová síť a časová konvoluční neuronová síť.

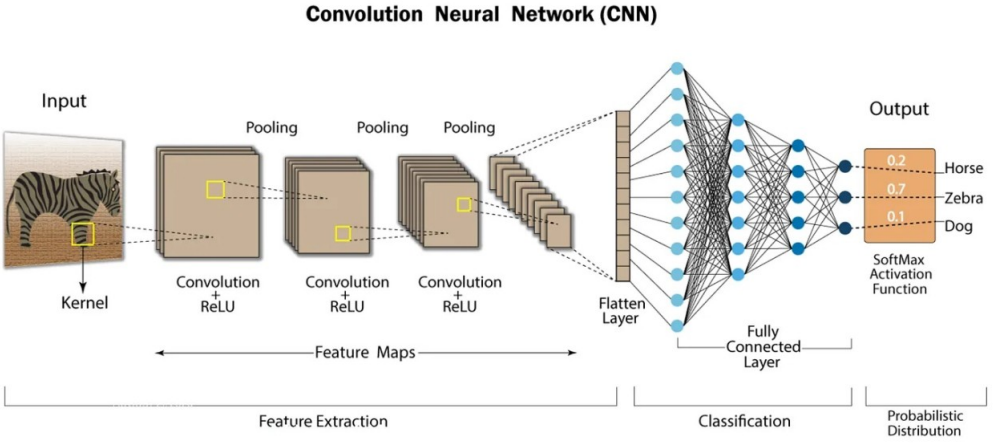
3.3.1 Vícevrstvý perceptron

Vícevrstvý perceptron (MLP - angl. Multi-Layer Perceptron) je základním typem dopředné umělé neuronové sítě. Skládá se ze 3 nebo více vrstev plně propojených neuronů. Výstupní hodnoty neuronů jedné vrstvy jsou posílány na vstupy vrstvy následující. Jako aktivační funkce se nejčastěji volí nelineární sigmoidální a ReLU funkce. Signál se šíří pouze ze vstupu sítě na její výstup. Takovému procesu říkáme dopředné šíření. Nejenom MLP, ale i ostatní neuronové sítě, využívají ke trénování algoritmus zpětného šíření chyby (angl. backpropagation). Tento algoritmus je založen na metodě gradientního sestupu. Pro každý prvek z trénovací množiny se nejprve spočítají výstupní hodnoty sítě. Následně se určí hodnota ztrátové funkce (angl. loss function) jako rozdíl mezi získaným a požadovaným výstupem. Poté se spočítá gradient ztrátové funkce a ten se pak propaguje zpět přes všechny předchozí vrstvy až do vstupní vrstvy. Tyto gradienty představují rychlost změny ztrátové funkce. Na základě toho se upraví váhy propojení tak, aby se hodnota ztrátové funkce minimalizovala. Změna vah jednotlivých propojení se mění v závislosti na velikosti příspěvku neuronu k celkové chybě na výstupní vrstvě. Rychlost jejich změny, tedy velikosti kroku ve směru gradientu, určuje parametr rychlost učení (angl. learning rate). Takový postup se opakuje dokud není dosaženo minimální chyby.

MLP se na rozdíl od perceptronu vyznačují schopností rozdělit nelineárně separovatelná data a jejich využití najdeme při klasifikaci, rozpoznání nebo regresi.

3.3.2 Konvoluční neuronová síť

Konvoluční neuronová síť (CNN - angl. Convolutional Neural Network) je založena na architektuře vícevrstvého perceptronu a díky své schopnosti zpracovávat obrazová data se využívá zejména pro úlohy počítačového vidění. Příkladem takových sítí jsou AlexNet, VGGNet, GoogleNet nebo ResNet. Ve struktuře konvoluční neuronové sítě můžeme mezi vstupní a výstupní vrstvou najít v různém počtu další 3 typy vrstev [26] - konvoluční vrstva, pooling vrstva a plně propojené vrstvy. Takovou strukturu ilustruje obrázek 3.4. Úkolem prvních dvou je extrahovat ze vstupního obrázku příznaky a redukovat jejich dimenzi. Tyto příznaky jsou předány nejčastěji několika plně propojeným vrstvám, jejichž úkolem je provést klasifikaci. Všechny vrstvy si nyní podrobněji popíšeme.

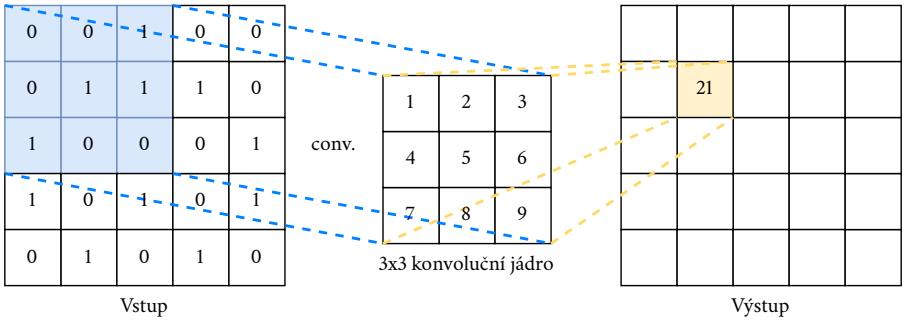


Obrázek 3.4: Architektura konvoluční neuronové sítě [27]

3.3.2.1 Konvoluční vrstva

Konvoluční vrstva je základem CNN a jejím úkolem je extrahovat příznaky ze vstupního obrázku reprezentovaného maticí pixelů. Hlavní operací prováděnou pří extrakci je konvoluce. Extrakce příznaků ze vstupní matice obrázku spočívá v posouvání okénka konvolučního jádra (kernelu) přes celý vstupní obrázek a aplikace skalárního součinu vah konvolučního jádra se vzorem na obrázku po každém posunutí okénka. Obrázek 3.5 ilustruje tento proces konvoluce. Délka posunutí okénka se označuje jako stride a typicky se při tom dva sousední kroky překrývají. Kernely jsou obvykle malé matice, jejichž velikost je menší, než rozměry původní matice, avšak jejich hloubka zůstává stejná. Nejčastěji kernely tvoří matice o rozměru 3x3, 7x7 apod. a jsou trénovány k rozpoznání určitého vzoru z obrazu (čára, zakřivení, hrana). To umožňuje různé nastavení vah v kernelu, ty jsou při posouvání kernelu přes celý vstupní obrázek sdílené a nedochází tak k jejich změně. Jako aktivační

funkce se po každé operaci konvoluce často používá funkce ReLU. Výstupem konvoluční vrstvy jsou tzv. mapy příznaků (angl. feature maps). Jejich rozměry závisí na velikosti posunu okénka a počtu kanálů na vstupu. Čím větší je krok, o který je kernel posunut, tím menší rozměry má výsledná mapa příznaků. Konvolučních vrstev, které se aplikují na vstupní obrázek, je obvykle několik. Struktura CNN je poté hierarchická, jelikož každá další vrstva má jako vstup mapy příznaků vytvořené předchozí vrstvou. Taková architektura umožňuje v obrázcích rozpoznávat složité vzory.



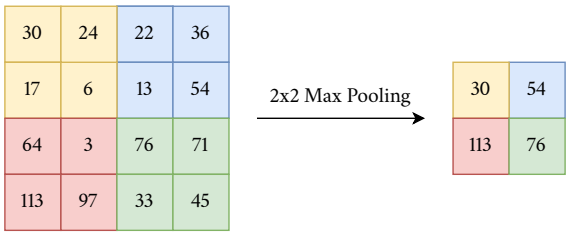
Obrázek 3.5: Proces konvoluce

3.3.2.2 Pooling vrstva

Pooling vrstva se často nachází mezi konvolučními vrstvami. Úkolem této vrstvy je vybrat pouze důležité příznaky získané v konvoluční vrstvě s cílem zachovat původní informaci. Redukuje dimenzi a výrazně snižuje počet příznaků. Pooling se podobně jako v konvoluční vrstvě provádí pomocí tzv. agregačních filtrů, které jsou posouvány přes výslednou mapu příznaků. Obvykle mají filtry rozměry 2x2 a při jejich posouvání typicky nedochází k překrývání. Existují 2 základní typy filtrů. Max pooling filtr vybere při každém posunu okénka filtru nejvyšší hodnotu z vybrané oblasti vstupních dat. Obrázek 3.6 názorně ilustruje tento proces. Average pooling filtr pracuje na podobném principu, akorát místo maximální hodnoty spočítá na výstup jejich průměr. Ačkoliv je v této vrstvě mnoho informace vypuštěno, pomáhá snížit složitost, zvýšit efektivitu, omezit riziko přeučení a zlepšit zobecnění sítě na nová data.

3.3.2.3 Plně propojené vrstvy

Plně propojené vrstvy jsou poslední částí struktury konvoluční neuronové sítě a dohromady tvoří vícevrstvý perceptron (viz 3.3.1). To znamená, že výstup každého neuronu v jedné vrstvě je napojen jako vstup na každý neuron ve vrstvě následující. Cílem vrstev MLP je provést úlohu klasifikace na základě příznaků extrahovaných



Obrázek 3.6: Aplikace max pooling filtru

předchozími konvolučními a pooling vrstvami. Aby tyto často vícerozměrné příznaky mohli být na vstupu plně propojené vrstvy, je třeba je transformovat do podoby jednorozměrného vektoru. V úplně poslední plně propojených vrstvě se jako aktivační funkce používá funkce softmax, která generuje pravděpodobnost každého možného výstupu v hodnotách od 0 do 1.

3.3.3 Rekurentní neuronová síť

Rekurentní neuronová síť (RNN - angl. Recurent Neural Network) je příkladem umělé neuronové sítě se zpětnou vazbou. Výstupy některých neuronů mohou být přivedeny jako vstupy do stejné nebo jiné vrstvy. RNN mají schopnost pamatovat si předchozí vstupy a vnitřní stav a tyto informace využívají ke zpracování sekvenčních dat (slova, věty, časové řady). Jednou z dalších vlastností je sdílení vah napříč vrstvami. Pro všechny neurony v jedné vrstvě je tak nastavena stejná váha. Při trénování využívají RNN algoritmus zpětného šíření v čase (BPTT - angl. Backpropagation Through Time), který je specifický pro sekvenční data. Princip je stejný jako u tradiční zpětné propagace, BPTT je ale typický tím, že sčítá chyby v každém časovém kroku. Článek [28] podrobněji popisuje rekurentní neuronové sítě a mimo jiné představuje dvě architektury - obousměrná RNN a Long Short Term Memory.

3.3.3.1 Obousměrná RNN

Obousměrná RNN (BRNN - angl. Bidirectional RNN) zpracovává data pomocí dopředných a zpětných vrstev neuronů. Dopředné vrstvy pracují na základním principu RNN. Ukládají si předchozí vstup do skrytého stavu a využívají ho pro předpovědi následujícího výstupu. Zpětná vrstva zpracovává data v opačném směru a pro predikci využívá i budoucí skrytý stav. BRNN tak bere v úvahu minulé i budoucí souvislosti a zvyšuje přesnost predikce.

3.3.3.2 Síť s dlouhou krátkodobou pamětí

Síť s dlouhou krátkodobou pamětí (LSTM - angl. Long Short Term Memory) je nejznámější architekturou RNN, která umožňuje rozšířit kapacitu paměti. Za nor-

málních okolností si RNN pamatují pouze bezprostředně minulé vstupy a přichází tak o kontext, který mohl být uložen ve vstupech předchozích. Díky tomu nemusí predikovat výstup správně. LSTM řeší tento problém pomocí tzv. buněk, které mají tři brány - vstupní, zapomínací a výstupní. Tyto brány umožňují zapamatovat si informace, které jsou důležité pro správnou predikci výstupu.

3.3.4 Časová konvoluční síť

V roce 2016 popsal Lea et al. [29] ve své práci koncept časové konvoluční neuronové sítě (TCN - angl. Temporal Convolutional Network) pro segmentaci akcí ve videu. Za normálních okolností by tento přístup vyžadoval dva kroky. Prvním je získání nízkoúrovňových příznaků pomocí CNN. Druhým krokem je předání těchto příznaků klasifikátoru (obvykle RNN), který je schopen zachytit vysokoúrovňovou časovou informaci o dění ve videu. Nevýhodou tohoto přístupu je nutnost dvou samostatných modelů. Tento problém řeší právě TCN, která poskytuje jednotný přístup k hierarchickému zachycení obou úrovní informací a zjednodušuje tak analýzu videa.

3.4 Počítačové vidění

Počítačové vidění (angl. Computer Vision) je jednou z oblastí umělé inteligence, která ve velkém využívá výše popsané koncepty strojového učení. Počítačové vidění se zabývá zpracováním a získáváním smysluplných informací o objektech ze zachycených digitálních obrazových dat. Cílem počítačového vidění je tedy naučit stroj vidět prostředí, detailně mu porozumět a na základě získaných informací činit rozhodnutí, stejně jako by to udělal člověk. Základními technikami využívanými v tomto oboru jsou sítě hlubokého učení, hlavně konvoluční a rekurentní neuronové sítě. V poslední době do oblasti počítačového vidění pronikají i tzv. transformátory [30], které jsou vhodné pro sekvenční úlohy a používají se tak zejména v oblasti NLP. Počítačové vidění je tedy více než jakákoliv jiná oblast závislá na množství obrazových trénovacích dat, na kterých jsou neuronové sítě trénovány. Konvoluční neuronová síť se používá k pochopení jednotlivých obrázků, rekurentní neuronová síť umí z videa určit souvislosti několika po sobě jdoucích snímků. Článek [31] představuje 3 hlavní úlohy na poli počítačového vidění. Jsou to klasifikace obrazu, detekce objektů a segmentace obrazu.

3.4.1 Klasifikace obrazu

Klasifikace obrazu (angl. Image Classification) je základní úlohou v počítačovém vidění. Cílem je rozpoznat obsah na obrázku a zařadit ho jako celek do jedné z předem definovaných tříd. K této úloze jsou nejvhodnější konvoluční neuronové sítě.

Jejich trénování probíhá na velkých datasetech, obsahujících obrázky spolu se známým zařazením do třídy. Natrénovaný model je pak schopný určit třídu pro nové neviděné obrázky.

3.4.2 Detekce objektů

Detekce objektů (angl. Object Detection) je technika v počítačovém vidění pro vyhledávání objektů v obraze. Jejím cílem je označit polohu nalezeného objektu, nejčastěji pomocí ohraničujících rámečku, a určit jeho kategorii. Pro nalezení objektu se nabízí použít techniku posuvného okna, kterým bychom postupně skenovali celý obrázek a v každém kroku prováděli klasifikaci pomocí CNN. Hlavní nevýhodou je nutnost vytvoření velkého počtu různých oříznutí obrázku a s tím rostoucí výpočetní náročnost. Lepším přístupem je obrázek rozdělit do menších regionů podle pravděpodobného výskytu objektu. Na tyto regiony pak aplikujeme CNN. Takový model se nazývá Region-based CNN (R-CNN). Článek [32] představuje rychlejší a vylepšenou variantu této neuronové sítě - Fast R-CNN. Nové přístupy již nevyžadují rozdělení obrázku do regionů, ale jsou schopné detekovat objekty přímo z jednoho pohledu na celý obrázek, a jsou tudíž rychlejší a efektivnější. Lze je poté použít pro detekci v reálném čase. Mezi takové algoritmy patří především YOLO (You Only Look Once) [33], který je v současné době považován za nejlepší řešení v oblasti detekce objektů.

3.4.3 Segmentace obrazu

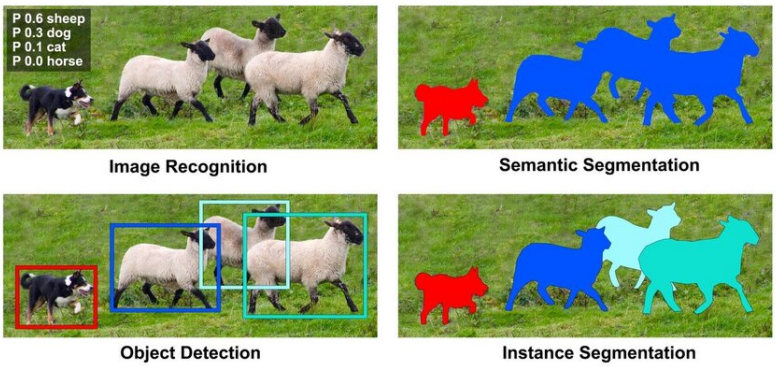
Segmentace obrazu (angl. Image Segmentation) představuje techniku rozdělení celého obrázku na oblasti se společnými vlastnostmi. Jedná se o metodu zpracování obrazových dat na úrovni pixelů. Pomocí různých technik anotuje jednotlivé pixely podle toho, do jaké třídy náleží. Výstupem této anotace jsou tzv. segmentační masky, které reprezentují hranice a tvary jednotlivých tříd. Segmentaci obrazu můžeme rozdělit na úlohy sémantické segmentace a segmentace instancí.

Sémantická segmentace je nejjednodušším typem segmentace. Jejím úkolem je sémanticky pochopit roli každého pixelu v obrázku a každému z nich přiřadit jednu třídu. Zde úloha končí. Tato metoda však neumí rozlišit objekty stejné třídy a další kontext obrázku nebo informace o objektech nám tak nejsou známy.

Segmentace instancí představuje přístup, který zahrnuje nalezení a oddělení jednotlivých objektů v obrázku tím způsobem, že každému pixelu v obrázku je přiřazena konkrétní instance objektu. Segmentace instancí tak vymezuje přesný tvar a hranice každého samostatného objektu a je tedy přesnější než sémantická segmentace.

Tradiční přístupy využívají pro segmentaci barevné vlastnosti jednotlivých pixelů. Mezi takové metody patří prahování, detekce hran či analýza histogramu. Mo-

dernější metody jsou založené na hlubokém učení a využívají výhody konvolučních neuronových sítí. Významnými modely v pro úlohu segmentace obrazu jsou FCN [34] nebo Mask R-CNN [35].



Obrázek 3.7: Porovnání výstupů úloh počítačového vidění [36]

Aplikace detekce událostí

4

V této kapitole je představen postup při aplikaci detekce událostí. Cílem je detekovat události ve videozáznamech fotbalových utkání, které nutně nevyžadují podrobné zpracování nebo získání statistik a dalších údajů. To znamená, že tyto zápasy nejsou například tak divácky atraktivní a důležité, nejsou vypsány kurzové příležitosti, které právě těží informace a údaje o těchto zápasech, nebo existuje pouze malý počet fanoušků, který se o dané týmy opravdu zajímá, a nestojí za tu námahu ručně zápas zpracovávat. Nejčastěji se jedná o utkání v nižších neprofesionálních soutěžích nebo přátelské či tréninkové zápasy profesionálních týmů, kde automatická detekce událostí může vlastní analýzu značně zjednodušit a urychlit. Konkrétně se budeme snažit aplikovat automatickou detekci na videozáznamy zápasů v českých nižších fotbalových soutěží (krajský přebor, divize, ČFL).

Pro vytvoření modelu, který by byl schopen tuto úlohu automatizovat využijeme dataset SoccerNet-v2 [37] a s ním spojený GitHub repozitář SoccerNet-v2 Development Kit [38], na jehož vytvoření se podíleli právě autoři datasetu. Tento development kit slouží jako základ pro další práci s daty a obsahuje programové kódy v jazyce Python implementující metody nejenom pro vytvoření modelu neuronové sítě a jeho natrénování, ale také pro anotaci zápasu, extrakci příznaků a vyhodnocení výsledků. Pro vývoj modelu hlubokého učení a práci s neuronovou sítí je využívána knihovna PyTorch, která se zaměřuje na úlohy strojového učení. Důležitou roli hraje výpočetní platforma a programovací model CUDA od společnosti NVIDIA, která umožňuje výpočty na grafických procesorech (GPU). V neposlední řadě je potřeba balíček SoccerNet [39], který umožňuje stahování zápasů, anotací, příznaků a dalších souborů, které jsou součástí datasetu.

Kromě výše popsaných technologií potřebujeme také dostatečný počet vlastních videozáznamů, na které budeme dané metody aplikovat. Cílem této praktické části je tedy projít si podrobně všechny dostupné programové kódy obsažené v repozitáři, pochopit jejich fungování, porozumět jednotlivým proměnným a tyto kódy následně přizpůsobit a modifikovat pro práci s vlastními zápasy. Nejprve se seznámíme s datasetem SoccerNet-v2 [37]. Poté se budeme zabývat získáním vlastních

videozáznamů a jejich přípravou pro další zpracování a trénování. Podrobněji si popíšeme, jakým způsobem se detekce událostí provádí a nakonec se pokusíme vhodně nastavit parametry modelu pro naší úlohu a provedeme několik experimentů.

4.1 Dataset SoccerNet-v2

Dataset SoccerNet-v2 je jedním z největších datasetů pro úlohu detekce událostí (angl. action spotting) ve fotbalu. Neslouží pouze pro detekci, ale také pro časovou lokalizaci události ve videu. Obsahuje celkem 500 videozáznamů televizních přenosů zápasů ze 3 po sobě jdoucích sezón (2014-2017) z 5 největších evropských lig (Premier League - Anglie, Laliga - Španělsko, Ligue 1 - Francie, Bundesliga - Německo, Serie A - Itálie) a Ligy mistrů UEFA. Tabulka 4.1 zobrazuje distribuci zápasů jednotlivých lig a sezón v datasetu.

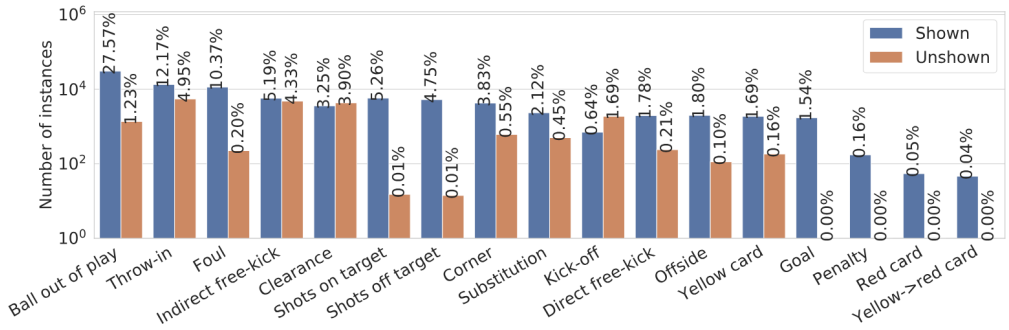
Data zápasů jsou strukturovány do adresářů podle ligy, sezóny a názvu zápasu. Adresář se samotným zápasem obsahuje videozáznamy z prvního a druhého poločasu ve formátu MKV v nízké (224p) a vysoké kvalitě (720p), JSON soubory s anotacemi a předem spočítané příznakové matice (ResNet). Celková délka datasetu činí 764 hodin a bylo v něm ručně anotováno okolo 300 tisíc časových značek. Anotace jsou rozdělené do 3 kategorií - *actions, camera shots, replays*. Nás v této práci budou zajímat jenom *actions*, které reprezentují jednotlivé události.

Liga	14/15	15/16	16/17	Celkem
Premier League	6	49	40	95
LaLiga	18	36	63	117
Ligue 1	1	3	34	38
Bundesliga	8	18	27	53
Serie a	11	9	76	96
UCL	37	45	19	101
Celkem	81	160	259	500

Tabulka 4.1: Distribuce zápasů v datasetu

Celkem dataset rozlišuje 17 typů událostí a je rozdělen na tři části - trénovací set (300 zápasů), validační set (100 zápasů) a testovací set (100 zápasů). Trénovací set slouží pro trénování modelu a je tedy předán trénovacímu algoritmu. Validační set se používá pro odhadování přesnosti modelu při ladění hyperparametrů. Testovací set slouží pro celkové vyhodnocení přesnosti modelu na neznámých datech.

Každá událost je anotována jednou časovou značkou ve chvíli, kdy nastala. Obrázek 4.1 představuje rozložení jednotlivých událostí v rámci datasetu. Celkový počet anotovaných událostí činí 110 458, což je v průměru 221 událostí na zápas.



Obrázek 4.1: Distribuce událostí v datasetu SoccerNet, převzato z [37]

4.2 Příprava vstupních dat

Pro aplikaci detekce událostí je třeba nejprve získat vlastní videozáznamy a předzpracovat je do vhodného formátu. Všechny poskytnuté videozáznamy byly pořízené kamerovým systémem Veo popsaným v podsekci 2.2.1. Hlavním zdrojem dat pro tuto práci byly videozáznamy zápasů fotbalového oddílu TJ Košutka Plzeň, který působí v krajském přeboru mužů. K těmto datům byl poskytnut neomezený přístup. Dalším zdrojem dat byla veřejná tabulka s odkazy na videozáznamy vybraných zápasů ze soutěží Divize a ČFL přístupná na serveru Fotbal.cz pod organizací Řídící komise pro Čechy [40]. Všechny zápasy se odehráli v rámci letošní sezóny 2023/2024.

4.2.1 Stahování a úprava záznamů

Prvním krokem při získání dat bylo stažení videozáznamů nahraných na internetu na platformě Veo. Takto jsme získaly videozáznamy 12 zápasů v rozlišení 1920x1080 ve formátu MP4 o přibližné celkové délce 18 hodin. Zastoupení jednotlivých soutěží můžeme vidět v tabulce 4.2. Vzhledem k tomu, že kamera Veo zachycuje dění na hřišti plně automaticky, nevyžaduje přítomnost kameramana, který by nahrávání zapnul pouze při hře. Realita je taková, že nahrávání se zapne pár minut před začátkem zápasu a běží nepřetržitě dokud se pár minut po skončení zápasu nezastaví. Z tohoto důvodu je třeba z celého dlouhého záznamu vystřihnout zvlášť první a druhý poločas. K tomu jsme využili program LossLessCut [41], který poskytuje grafické rozhraní využívající FFmpeg knihovny, které slouží pro rychlé a bezeztrátové zpracování videa, hlavně stříhaní a ořezávaní.

Abychom zůstali konzistentní s původním datasetem při pojmenovávání adresářů a souborů, vytvořili jsme si vedle adresářů s původními zápasy v datasetu také adresář s názvem `krajsky_prebor`. V tomto adresáři jsme pak vytvořili složky s jednotlivými zápasy, pojmenované podle hrajících týmů. Do těchto složek jsme

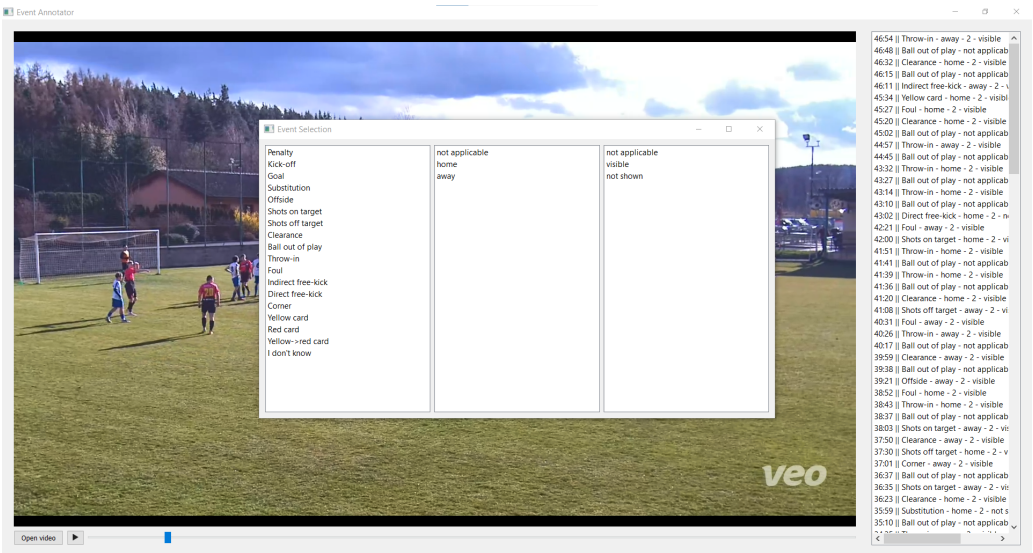
Soutěž	Počet
Krajský přebor	6
Divize	2
ČFL	3
Přátelský	1
Celkem	12

Tabulka 4.2: Získané zápasy

poté ukládali získané záznamy poločasů. První poločas vždy s názvem `1_720p.mp4` a obdobně druhý poločas s názvem `2_720p.mp4`.

4.2.2 Tvorba anotací

Protože se při trénovaní neuronové sítě využívá algoritmu učení s učitelem, nestačí nám pouze předzpracované videozáznamy. Musíme také označit nastalé události ve videozáznamu, abychom tak vytvořili trénovací dvojice vstup-požadovaný výstup. Bylo proto třeba zhlédnout všechny získané videozáznamy a manuálně označit jednotlivé události a zaznamenat jejich časovou značku reprezentující pozici ve videu. Pro vytvoření takových anotací jsme využili nástroj, který je součástí development kitu. Jedná se o jednoduchou aplikaci vytvořenou pomocí frameworku PyQt. Uživatelské rozhraní aplikace můžeme vidět na obrázku 4.2.



Obrázek 4.2: Uživatelské rozhraní aplikace

Kliknutím na tlačítko *Open video* můžeme otevřít vybraný záznam poločasu

a libovolně se v něm pomocí posuvníku nebo klávesových zkratek pohybovat. Nejprve video zastavíme na snímku, na kterém událost nastala, a pak klávesou *Enter* otevřeme menu, ve kterém vybereme jednu ze 17 událostí. Nutno podotknout, že událost je označena pouze jako jeden moment, nikoliv časový úsek. Seznam událostí a jejich popis obsahuje příloha A. Dále můžeme určit, jestli se o událost postaral domácí nebo hostující tým (*home, away*) a jestli byla na videu viděna či nikoliv (*visible, not shown*). Anotované události se v pravé části aplikace zobrazují jako seznam a lze je odtud mazat, nebo mezi nimi libovolně přepínat. Tyto anotované události se ukládají do souboru `Labels-v2.json` společného pro oba poločasy pod klíč *annotations*. Příklad jedné anotované události může vypadat následovně:

```
"UrlLocal": "krajsky_prebor/kosutka_nyrsko/",
"annotations": [
    {
        "gameTime": "1 - 00:02",
        "label": "Kick-off",
        "position": "2483",
        "team": "home",
        "visibility": "visible"
    },
    ...
```

4.2.3 Extrakce příznaků

Abychom mohli s videozáznamem dále pracovat a předložit ho použité neuronové síti pro natrénování, musíme z něj extrahovat příznaky a zredukovat je pak pouze na ty důležité. Základní koncept extrakce příznaků a redukce dimenze je popsán v podsekci 3.1.6.5. Extrakci příznaků si můžeme konkrétně představit jako uložení výstupu určité vrstvy neuronové sítě, obvykle těsně před klasifikační vrstvou. Ze vstupních obrazových dat lze pomocí již natrénované sítě získat vektory příznaků, které můžeme použít jako vstupní data pro další úlohy, v našem případě pro detekci událostí. To urychluje proces trénování, protože není nutné trénovat celou síť od začátku.

Pro úlohu detekce používáme jako extraktor příznaků síť ResNet-152 [42], předtrénovanou na datasetu ImageNet. Všechny zápasy v datasetu SoccerNet-v2 mají tyto příznaky předpočítané a lze je pro účely trénování stáhnout. Jsou získané při 2 snímcích za sekundu a kromě samotné extrakce se provádí i redukce dimenze na velikost 512 pomocí techniky PCA (viz 3.1.6.5).

U vlastních zápasů je ale třeba tyto příznaky extrahovat. K tomu využijeme opět development kit a kód pro extrakci příznaků z ResNet-152. Tento kód používá kni-

hovnu Keras. Pomocí této knihovny je vytvořen základní model ResNet-152 a jeho váhy jsou přednastaveny podle datasetu ImageNet. Tento model je následně upraven tak, aby jeho celkovým výstupem byl výstup avg_pool vrstvy původního ResNet-152 modelu. Nakonec je model nastaven tak, aby jeho váhy zůstaly po celou dobu neměnné a mohl tak sloužit jako extraktor příznaků. Po extrakci následuje redukce dimenze příznaků pomocí normalizace průměrem a následné aplikace předpočítaného modelu PCA.

4.3 Model

Cioppa et al. [14] navrhli pro automatickou detekci událostí model neuronové sítě, jehož cílem je získat přesný čas, ve kterém daná událost ve videozáznamu nastala. Tento model využívá pro detekci časový kontext dění ve videu před a po události v závislosti na jejím typu. Některé události vyžadují delší časový kontext, jiné kratší. Na základě tohoto předpokladu je počítána ztráta při trénování pomocí vlastní navržené funkce - **Context-aware loss function** (CALF). Tato ztrátová funkce respektuje dění okolo události a pro její výpočet jsou události anotované jednou časovou značkou. V následujících sekcích si podrobněji představíme architekturu sítě a způsob, jakým je ztráta počítána.

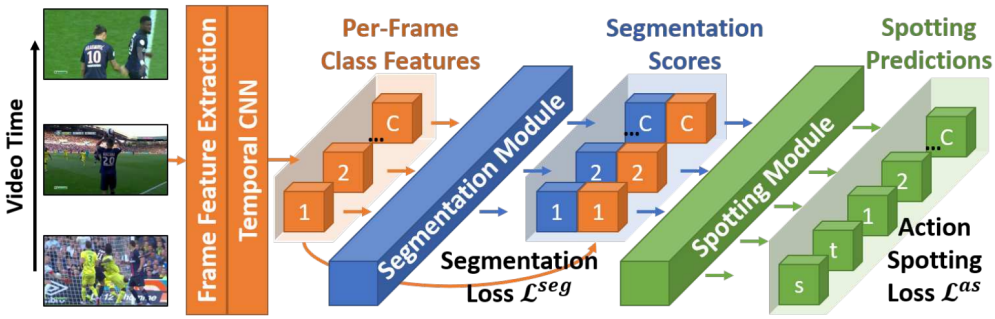
4.4 Architektura neuronové sítě

Architektura použité neuronové sítě je ilustrována na obrázku 4.3, podrobný popis její struktury ve formě grafu je uveden v příloze B. Jak již bylo zmíněno, neuronová síť využívá příznaky reprezentující vlastnosti jednotlivých snímků ve videu, extrahované pomocí sítě ResNet. Tyto příznaky jsou konkrétně vstupem pro časovou konvoluční neuronovou síť, jejíž základní koncept je popsán v podsekci 3.3.4. Tato TCN se skládá z dvouvrstvého MLP, který je následován čtyřmi 3D konvolučními vrstvami. Výstupem této konvoluční sítě jsou vektory příznaků všech tříd C o velikosti f pro každý snímek videa. Získané příznaky jsou vstupem pro segmentační modul.

Segmentační modul využívá Batch normalizaci a sigmoidální aktivační funkce pro určení výstupu. Výstupem je C segmentačních skóre každý snímek videa. Jedná o predikci toho, k jaké události snímek ve videu patří s největší pravděpodobností. Čím je predikce bližší informaci od učitele, tím je vyšší i skóre. Tyto hodnoty jsou spojeny s příznaky, získanými v prvním kroku a jsou předány jako vstup do detekčního modulu.

Detekční modul slouží pro klasifikaci. Skládá se ze 3 po sobě jdoucích max pooling a 3D konvolučních vrstev. Výstupem modulu je N_{pred} vektorů o délce $2 + C$. Počáteční dva prvky jsou výstupem sigmoidální aktivační funkce. Prvním z nich

je skóre reprezentující věrohodnost detekce události v daném snímku, druhý představuje časovou pozici události. Zbývající prvky jsou výstupem aktivační funkce softmax a reprezentují predikce pro jednotlivé třídy událostí. Tyto vektory jsou poskládány do matice $\hat{\mathbf{Y}}$ o rozměrech $N_{pred} \times (2+C)$ představující všechny predikované události.



Obrázek 4.3: Architektura neuronové sítě pro detekci událostí, převzato z [14]

4.5 Ztrátová funkce

Pro výpočet ztráty navrhli Cioppa et al. [14] kombinaci dvou ztrátových funkcí. První z nich se nazývá časová segmentační ztráta a počítá ztrátu pro výstup ze segmentačního modulu. Druhou z nich je detekční ztráta a počítá ztrátu detekčního modulu. Celková ztráta je pak dána součtem těchto dvou. Nejprve si představíme základní koncepty pro výpočet ztrátové funkce, zejména časové segmentační ztráty, která pro výpočet své hodnoty nevyužívá jenom jeden snímek, obsahující anotovanou událost, ale také okolní snímky pro zjištění hlubšího kontextu. Poté si obě ztrátové funkce podrobněji popíšeme a také si představíme, jakým způsobem jsou anotovaná data přepsána do formátu vhodného pro trénování sítě.

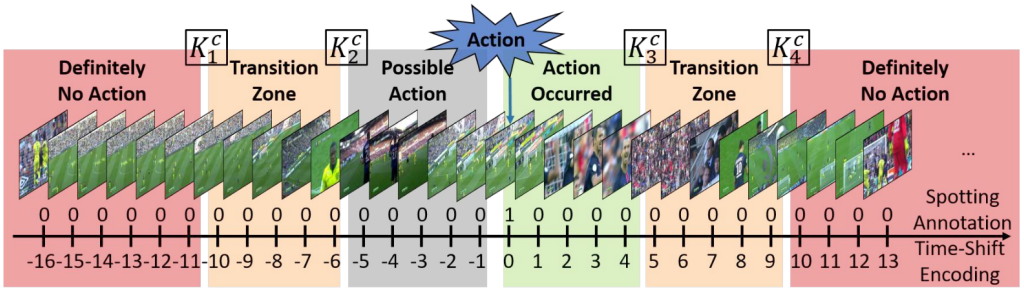
Obrázek 4.4 představuje rozdělení časového kontextu události c do šesti intervalů snímků pomocí 4 parametrů $K_1^c < K_2^c < 0 < K_3^c < K_4^c$ v závislosti na jejich vzdálenosti od události. Nastavení těchto parametrů je závislé na jejím typu, jejich hodnota se zadává v sekundách. Při trénování se se snímky v každém intervalu nakládá při výpočtu ztráty trochu jinak.

Snímky daleko před nebo daleko po události jsou pro danou událost zanedbatelné, protože nelze s jistotou říci, jaká událost nastane nebo nastala. Síť je trénována tak, aby událost nepredikovala.

Ze snímků těsně před událostí nelze předpovědět, že k události skutečně dojde, a proto není síť nijak ovlivněna. Příkladem může být hráč běžící s míčem sám na bránu. Tato situace může nebo nemusí vést ke gólu.

Snímky těsně po události jsou naopak pro síť velmi důležité, protože obsahují mnoho kontextuálních informací o právě proběhnuté události a síť je proto trénována pro predikci události. Příkladem je gólová oslava po vstřelení gólu. Z této informace můžeme s velkou pravděpodobností určit, že gól byl opravdu vstřelen.

Nakonec jsou definovány přechodné zóny, které slouží pro plynulý přechod mezi výše popsanými intervaly a řeší tak případnou nespojitost ztrátové funkce, kterou si podrobně popíšeme později. V těchto oblastech se síť jemně trénuje, aby nepředpovídala události.



Obrázek 4.4: Rozdělení časového kontextu události, převzato z [14]

4.5.1 Kódování vstupních dat

Aby síť mohla být trénována, musí jí být trénovací data předána v definované podobě. Je proto třeba všechny anotace přepsat do vhodného formátu.

Pro výpočet časové segmentační ztráty se provádí přepis pozice ve videu do tvaru, který reprezentuje posun snímku vzhledem k jeho nejbližším minulým nebo budoucím událostem. Každý snímek je tak reprezentován vektorem o délce C rovné počtu všech tříd událostí, kde každý prvek náleží jednomu typu události. Každý prvek v tomto vektoru pak konkrétně představuje počet snímků a může nabývat kladných nebo záporných hodnot podle toho, zda je nejbližší událost dané třídy minulá nebo budoucí. Uvažujme tentokrát 3 třídy událostí c_1, c_2, c_3, kde událost c_1 nastala ve snímku 50, událost c_2 nastala ve snímku 100 a událost c_3 nastala ve snímku 120. Snímek 90 pak bude kódován pomocí vektoru popsaného tabulkou 4.3.

40	-10	-30

Tabulka 4.3: Posun snímku vzhledem k nejbližší události

Pro výpočet detekční ztráty je definován přepis anotací inspirovaný modelem YOLO [33]. Každá anotovaná událost je reprezentována vektorem o délce $2 + C$. První prvek je binární indikátor toho, zda je událost ve videu přítomna (1) či nikoliv

(0). Druhý prvek pak představuje pozici snímku označeného jako událost ve videu. Ostatní prvky pak určují o jakou událost se jedná. Tedy hodnota 1 u indexu anotované události a hodnota 0 u ostatních. Tyto vektory jsou poté poskládaný do matice **Y** o rozměrech $N_{GT} \times (2+C)$, kde N_{GT} je počet všech anotovaných událostí. Tabulka 4.4 ilustruje podobu takové matice pro 5 tříd událostí a 4 anotované události.

1	0.1	1	0	0	0	0
1	0.35	0	0	1	0	0
1	0.6	1	0	0	0	0
1	0.85	0	0	0	1	0

Tabulka 4.4: Přepis anotovaných událostí do matice

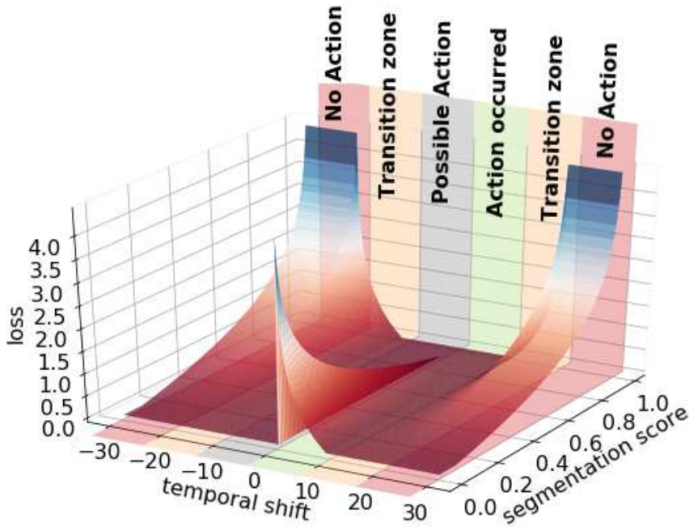
4.5.2 Časová segmentační ztráta

Pro výpočet časové segmentační ztráty je třeba výstupní skóre ze segmentačního modulu, reprezentující náležitost do třídy c. Toto skóre je označeno jako p. Označení s představuje informaci o posunu snímku vůči nejbližší události. V závislosti na pozici snímku x ve videu a předem definovaných K intervalů popsaných výše se vypočítá hodnota ztrátové funkce CALF $\widetilde{L}(p,s)$ generovaná výstupním skóre p ze segmentačního modulu a posunem s za pomoci následujících vztahů:

$$L(p,s) = \begin{cases} -\ln(1-p) & s \leq K_1^c \\ -\ln\left(1 - \dfrac{K_2^c - s}{K_2^c - K_1^c} p\right) & K_1^c < s \leq K_2^c \\ 0 & K_2^c < s < 0 \\ -\ln\left(\dfrac{s}{K_3^c} + \dfrac{K_3^c - s}{K_3^c} p\right) & 0 \leq s < K_3^c \\ -\ln\left(1 - \dfrac{s - K_3^c}{K_4^c - K_3^c} p\right) & K_3^c \leq s < K_4^c \\ -\ln(1-p) & s \geq K_4^c. \end{cases} \tag{4.1}$$

Obrázek 4.5 názorně ilustruje ztrátovou funkci CALF $\widetilde{L}(p,s)$, jejíž přesný výpočet je definován v rovnici 4.3. Můžeme vidět, jak tato funkce rozdílně penalizuje výstupy neuronové sítě (segmentačního modulu) v závislosti na tom, jaký snímek videa se právě zpracovává. Snímky daleko před a daleko po události jsou penalizovány, protože z nich nelze určit, zda akce nastane nebo nastala. Snímky těsně před událostí nejsou nijak penalizovány, a ztráta je tak rovna 0. Ze snímků těsně po události by neuronová síť měla tuto události spolehlivě rozpoznat. Nesprávné predikce jsou proto penalizovány. Výpočet ztráty v tomto intervalu je navržena tak,

že s postupem času jde hladce od hodnoty $-\ln(p)$ pro anotovaný snímek k hodnotě 0. V přechodných intervalech je ztrátová funkce definována tak, aby byla mezi intervaly spojitá a aby intervaly hladce mezi sebou přecházely.



Obrázek 4.5: Context-aware loss function, převzato z [14]

Výsledná segmentační ztráta pro video o N_F snímcích se pak vypočítá pomocí vztahu

$$\mathcal{L}^{seg} = \frac{1}{CN_F} \sum_{i=1}^{N_F} \sum_{c=1}^{C} \widetilde{L}\left(p^c\left(x_i\right), s^c\left(x_i\right)\right), \tag{4.2}$$

kde $\widetilde{L}(p,s)$ je počítáno jako

$$\widetilde{L}(p,s) = \begin{cases} \max(0, L(p,s) + \ln(\tau_{max}) & 0 \leq s < K_3^c \\ \max(0, L(p,s) + \ln(1-\tau_{min}) & \text{jinak.} \end{cases} \tag{4.3}$$

Smyslem rovnice 4.3 je pomoct síti zaměřit se na zlepšení nejhoršího segmentačního skóre vynulováním ztráty pro skóre, která jsou již dostatečně uspokojivá. Toho dosahuje nastavením parametrů τ_{min} a τ_{max}. V našem případě jsme nechali $\tau_{min} = 0.1$ a $\tau_{max} = 0.9$.

4.5.3 Detekční ztráta

Výstupem detekčního modulu jsou predikce jednotlivých událostí, které jsou reprezentovány maticí $\hat{\mathbf{Y}}$ ve formátu $N_{pred} \times (2 + C)$, kde N_{pred} je počet predikovaných událostí. N_{GT} představuje počet všech anotovaných událostí v trénovacích datech. Při výpočtu ztráty se využívá iterativní one-to-one algoritmus [14], který spojí každou událost z trénovacího setu s predikovanou událostí, která je jí časově nejblíže,

a obráceně, tedy predikovanou událost s nejbližší anotovanou událostí z trénovacího setu. Predikce a anotované události, které jsou vzájemně spojené vytvoří pár. Takto se postupuje, dokud všechny anotované události nejsou spojené s některou z predikcí. Jako $\hat{\mathbf{Y}}^M$ je označena matice, kde prvních N_{GT} predikcí má své páry mezi anotovanými událostmi a ostatní predikce jsou nespárované. Detekční ztráta je poté definována rovnicí 4.4.

$$\mathcal{L}^{as} = \sum_{i=1}^{N_{GT}} \sum_{j=1}^{2+C} \alpha_j \left(\mathbf{Y}_{i,j} - \hat{\mathbf{Y}}^M_{i,j} \right)^2 + \beta \sum_{i=N_{GT}+1}^{N_{pred}} \left(\hat{\mathbf{Y}}^M_{i,1} \right)^2 \tag{4.4}$$

4.5.4 Celková ztráta

Rovnice 4.5 představuje výpočet celkové ztráty $\mathcal{L}$, která je počítána jako vážený součet segmentační ztráty $\mathcal{L}^{seg}$ a detekční ztráty $\mathcal{L}^{as}$.

$$\mathcal{L} = \mathcal{L}^{as} + \lambda^{seg} \mathcal{L}^{seg} \tag{4.5}$$

4.6 Trénování

Pro trénování sítě se využívá dávkový přístup. Změna vah se tedy provede až po předložení celé jedné dávky (angl. batch). Zásadním hyperparametrem ovlivňující rychlost učení a změnu vah je *learning_rate* a je třeba ho nastavit tak, aby model konvergoval k optimálnímu natavení vah. Každá dávka pak obsahuje několik chunků. Takový chunk je definován jako sada N_F vektorových příznaků jednotlivých sousedících snímků. Pokud je video vzorkováno na 2 snímky za sekundu, velikost chunku 240 reprezentuje 2 minuty původního videozáznamu. Tyto chunky jsou získávány kolem každé anotované události. Každá dávka obsahuje náhodně vybrané chunky tak, aby všechny události byly při trénování zastoupeny stejnou měrou. To znamená, že událost s menším počtem výskytů bude vybírána stejně pravděpodobně, jako jiné velmi časté události. Počet chunků v jedné dávce závisí na parametru *batch_size*. Počet chunků, které mají projít neuronovou sítí za jednu epochu, definuje parametr *chunks_per_epoch*. Tyto dva parametry pak ovlivňují počet iterací. Pokud tedy parametr *chunks_per_epoch* bude roven 16000 a *batch_size* bude 32, celkový počet iterací pro dokončení jedné epochy bude $16000/32 = 500$.

4.7 Experimenty a vyhodnocení

Z výše uvedeného popisu modelu vyplývá, že existuje velké množství parametrů, a prostor pro experimentování je tak opravdu velký. My se budeme snažit parametry modifikovat tak, abychom mohli model úspěšně aplikovat na vlastní dataset.

Zaměříme se na parametry, které jsou pro trénování zásadní, a které nejvíce mohou ovlivnit samotný výstup. Vzhledem k tomu, že náš vlastní dataset obsahuje pouze 12 zápasů (18 hodin), nemáme dostatečné množství trénovacích dat pro provedení experimentů, na jejichž výsledky bychom se mohli zcela spolehnout. I přes to lze při správném nastavení parametrů pozorovat relativně přijatelné výsledky.

Jednou z možností, jak využít získaný model pro vlastní detekci událostí, je natrénovat ho na vlastním datasetu od začátku. Protože máme k dispozici ale i model předtrénovaný na datasetu, jehož charakter záznamů je velmi podobný záznamům v našem datasetu, nabízí se i druhá možnost, a to použít tento model pro dotrénování tak, aby se adaptoval i pro naše data. Oba tyto přístupy vysvětlíme později v sekci 4.7.

v obou případech je třeba vlastní dataset nejprve rozdělit na trénovací, validační a testovací množinu. V ideálním případě, by každá z množin měla obsahovat různá data a tedy i různé zápasy. V případě našeho malého datasetu jsme však dospěli k takovému rozdělení, kdy testovací množina obsahuje stejné zápasy jako validační. Na výsledek to však nebude mít vliv, zásadní je, aby se testovací zápasy lišily od těch trénovacích, což náš dataset splňuje. Zastoupení zápasů v jednotlivých množinách a počet zápasů v nich ukazuje tabulka 4.5.

Soutěž	Trénovací	Validační	Testovací
Krajský přebor	4	2	2
Divize	2	0	0
ČFL	3	0	0
přátelský	1	0	0
Celkem	10	2	2

Tabulka 4.5: Rozdělění zápasů v datasetu

Z popisu modelu a jeho ztrátové funkce můžeme odvodit, že změna parametrů bude dávat smysl právě zde. Kromě toho výsledky mohou také ovlivnit trénovací hypermarametry jako *batch_size*, *chunks_per_epoch* a především *learning_rate* (LR). Pro zjištění nejlepších hyperparametrů jsme provedli několik experimentů, které nám poskytli základní informace o vhodném nastavení. Dobrých výsledků dosahovalo následné nastavení parametrů: $chunks\_per\_epoch = 6000$, $batch\_size = 16$ a $learning\_rate = 0.001$. Kromě toho jsme přidali parametry *patience* a *factor*, které ovlivňují změnu LR. Konkrétně *patience* představuje počet epoch bez vylepšení modelu, po kterém dojde ke změně LR parametrem *factor*. V případě hodnoty $factor = 0.5$ se LR zmenší na polovinu. Tyto parametry jsme nechali neměnné pro všechny experimenty. V každém experimentu jsme síť trénovali po dobu 300 epoch. Tabulka 4.6 obsahuje seznam všech nastavitelných hyperparametrů trénování a parametrů modelu. Tučně označené parametry jsou empiricky odvozené

hodnoty shodné pro všechny experimenty, hvězdičky představují parametry lišící se v jednotlivých experimentech.

Parametr	Hodnota
SoccerNet_path	/path/to/SoccerNet/
features	ResNET_TF2_PCA512.npy
max_epochs	**300**
load_weights	None
model_name	CALF
test_only	False
K_params	None
num_features	512
chunks_per_epoch	**6000**
evaluation_frequency	5
dim_capsule	16
framerate	2
chunk_size	120
receptive_field	40
lambda_coord	5.0
lambda_noobj	*
loss_weight_segmentation	*
loss_weight_detection	1.0
batch_size	**16**
LR	**0.001**
patience	**150**
factor	**0.5**
GPU	-1
max_num_worker	4
loglevel	INFO

Tabulka 4.6: Nastavení hyperparametrů trénování a parametrů modelu

4.7.1 Metriky pro vyhodnocení experimentů

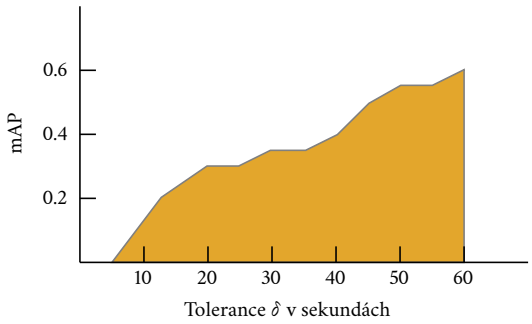
Vyhodnocení experimentů se provádí pomocí metriky pro detekci událostí definované v [15]. Detekce událostí spočívá kromě určení kategorie události také v nalezení jejího přesného času v záznamu. To je ale velmi obtížné, a proto Giancola et. al představují koncept časové tolerance δ, ve které je událost detekována.

Predikovaná událost je označena jako pozitivní, pokud je časový posun mezi ní a anotovanou událostí menší než tolerance δ. Tyto tolerance jsou definovány v rozmezí od 5 do 60 sekund po 5 sekundách. Celkem je tak vytvořeno 12 intervalů tolerancí, pro které se standardním způsobem počítají metriky přesnost (precision),

úplnost (recall), průměrná přesnost (AP) pro každou třídu události, počítána jako obsah pod precision-recall křivkou, a střední průměrná přesnost (mAP) pro všechny třídy. Pro obecné porovnání je definována ještě metrika průměrná mAP (Average-mAP), která je počítána jako obsah pod křivkou mAP v rozsahu δ od 5 do 60 sekund. To je znázorněno na obrázku 4.6. Následující 2 rovnice představují výpočet precision a recall. TP (True Positives) jsou správně detekované události, FP (False Positives) jsou nesprávně detekované události a FN (False Negatives) jsou události, které nebyly detekovány.

$$Precision = \frac{TP}{TP + FP} = \frac{Spravne\ detekce}{Vsechny\ detekce} \tag{4.6}$$

$$Recall = \frac{TP}{TP + FN} = \frac{Spravne\ detekce}{Vsechny\ anotace} \tag{4.7}$$



Obrázek 4.6: Metrika Average-mAP jako obsah pod křivkou mAP

4.7.2 Trénování od začátku

Trénování od začátku (angl. training from scratch) představuje přístup, kdy je neuronová síť trénována na vlastním datasetu (viz tabulka 4.5) bez předtrénovaných vah. Nejprve jsou v síti inicializovány náhodně váhy, které jsou následně po postupném předkládání trénovacích dat aktualizovány tak, aby minimalizovali celkovou ztrátu popsanou v rovnici 4.5. Nastavení jednotlivých hyperparametrů modelu můžeme vidět v tabulce 4.6.

Původní počet 17 tříd událostí jsme pro naše experimenty snížili na 8. Původní nastavení K parametrů pro výpočet CALF jsme pro každý typ události snížili na základě zkušenosti tak, aby lépe odpovídaly našim videozáznamům a zároveň, aby se mezi událostmi objevilo více akce na pozadí, tedy snímků, které jsou daleko od všech událostí.

Celkem jsme provedli 6 experimentů při kterých jsme postupně měnili nastavení parametrů *lambda_noobj*, který v rovnici 4.4 reprezentuje parametr β,

K_i^c	Penalta	Gól	Kop od branky	Míč v zámezí	Vhazování	Faul	Volný přímý kop	Rohový kop
K_1	-40	-8	-37	-39	-12	-30	-38	-30
K_2	-20	-4	-18	-20	-6	-14	-19	-15
K_3	20	20	18	20	6	14	19	15
K_4	40	30	37	39	12	30	38	30

Tabulka 4.7: Nastavení K parametrů pro 8 tříd

a *loss_weight_segmentation*, který představuje parametr λ^{seg} v rovnici 4.5. Výsledky jednotlivých experimentů spolu s Average-mAP jednotlivých událostí a celkovým Average-mAP můžeme vidět porovnané v tabulce 4.8.

exp.	β	λ^{seg}	Penalta	Gól	Kop od branky	Míč v zámezí	Vhazování	Faul	Volný přímý kop	Rohový kop	Average-mAP
1	1.0	0.000367	0.00	0.15	0.21	0.28	**0.31**	0.18	0.10	0.04	0.16
2	2.0	0.000367	0.46	0.13	0.05	0.30	0.29	0.13	0.03	0.01	0.18
3	0.5	0.0005	**0.52**	0.03	0.24	0.28	0.20	0.12	0.14	**0.17**	**0.21**
4	0.5	0.001	0.42	**0.28**	0.25	0.24	0.16	**0.22**	0.04	0.01	0.20
5	2.0	0.0005	0.00	0.25	**0.26**	**0.31**	0.21	0.19	0.19	0.04	0.18
6	1.0	0.0005	0.00	0.05	0.23	0.29	0.21	0.15	**0.21**	0.13	0.16

Tabulka 4.8: Porovnání výsledků (trénování od začátku)

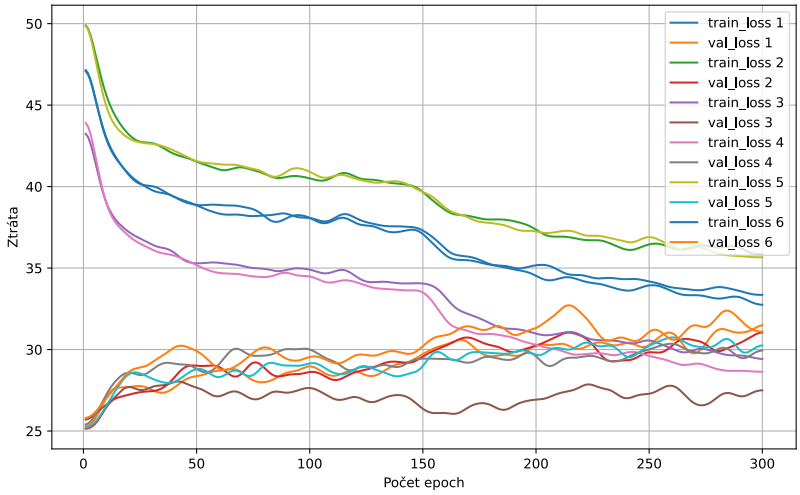
Z výsledků je zřejmé, že nejlepší predikce vyprodukoval model z experimentu č. 3, jehož Average-mAP dosahuje 21 %. Jeho podrobnější výsledky jsou rozepsány v tabulce 4.9. Metrika pojmenovaná Average-mAP visible představuje Average-mAP událostí, které byl anotované jako *visible*, Average-mAP unshown popisuje přesnost detekce událostí, které byly anotované jako *not shown*. Tento model vyčnívá hlavně v úspěšnosti detekce penalty. Naopak nedosahuje dobrých výsledků při detekci gólů, což je poměrně zásadní událost v zápasu, kterou bychom chtěli spolehlivě detekovat. Můžeme si také všimnout relativně vysokých přesností detekce míče v zámezí a vhazování, pokud nebyly zachyceny přímo v záběru. To vykazuje schopnost modelu detekovat událostí na základě okolního kontextu. Nuly u ostatních událostí v pří-

Událost	Avg.-mAP	Avg.-mAP visible	Avg.-mAP unshown
Penalta	0.52	0.52	0.00
Gól	0.03	0.03	0.00
Kop od branky	0.24	0.24	0.00
Míč v zámezí	0.28	0.27	0.42
Vhazování	0.20	0.19	0.44
Faul	0.12	0.12	0.00
Volný přímý kop	0.14	0.14	0.00
Rohový kop	0.17	0.17	0.00
Average-mAP	0.21	0.21	0.14

Tabulka 4.9: Experiment č. 3 (Avg. mAP per class)

padě unshown metriky znamenají, že se v testovací sadě takto označené události nevyskytovali.

Góly nejlépe detekuje model z experimentu č. 4, ale dosahuje nejhoršího skóre v případě rohových kopů, což je opět jedna z často sledovaných statistik, tudíž by nebylo vhodné k jejich detekci tento model použít. Všechny modely fungovali velmi podobně při detekci kopů od branky, míče v zámezí a vhazování a až na dvě výjimky dokázaly predikovat tyto události se skóre vyšším než 20 % Average-mAP. To pravděpodobně souvisí s častým výskytem těchto událostí v trénovacích datech a síť se tak tyto situace mohla dobře naučit.

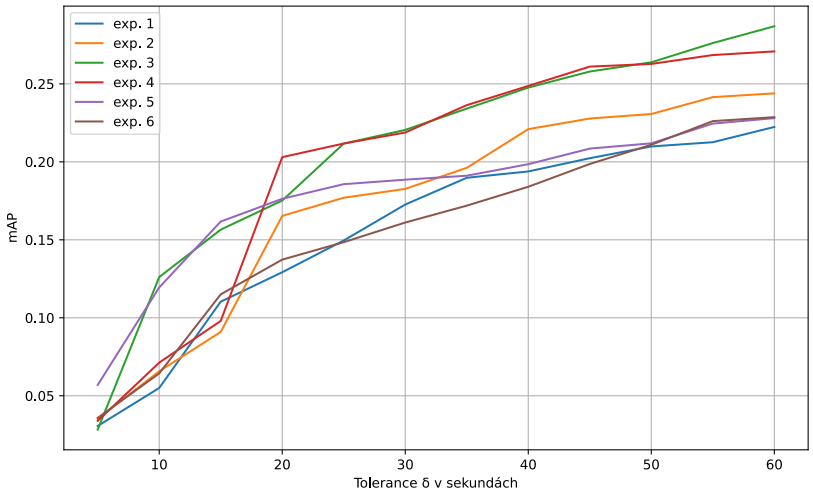


Obrázek 4.7: Průběh trénovací a validační ztráty (train from scratch)

Obrázek 4.7 zobrazuje vývoj trénovací a validační ztráty pro jednotlivé experimenty. Trénovací ztráta popisuje vývoj chyby na trénovací množině. Z klesající ztráty můžeme odvodit, že se neuronová síť v průběhu času učí a snaží se přizpůsobit trénovacím datům. Pokud bychom nechali síť trénovat více epoch, síť by se dále snažila učit a trénovací ztráta by tak pravděpodobně klesala dále.

Validační ztráta popisuje výkonnost modelu na validační množině obsahující data, která model během trénování neviděl. V ideálním případě by její hodnota měla klesat. V grafu ale můžeme vidět, že její hodnota pro všechny experimenty stagnuje nebo dokonce lehce stoupá. To naznačuje, že se model přeučuje na trénovací data a není schopen dobře predikovat výstupy pro data nová. Tento jev je nejspíše způsoben nedostatečným množstvím trénovacích dat a jejich malou variabilitou.

Z grafu je zřejmé, že model z experimentu č. 3 s parametry $\beta = 0.5$ a $\lambda^{seg} = 0.0005$ dosahuje nejnižší validační ztráty. Můžeme tedy předpokládat, že se jedná o nejlepší model v rámci všech provedených experimentů. Tento závěr podporuje i hodnota Average-mAP uvedená v tabulce 4.8, která dosahuje 21 %.



Obrázek 4.8: Závislost mAP na toleranci δ (train from scratch)

Obrázek 4.8 vykresluje hodnotu mAP jako funkci časové tolerance δ. Výsledky lze interpretovat tak, že modely dosahují nejvyšší přesnosti detekce, pokud se jejich predikované události nachází v časové toleranci 60 sekund od odpovídajících anotovaných událostí. Se snižující se tolerancí δ klesá počet přesně detekovaných událostí, což vede ke snížení hodnoty mAP. Můžeme vidět, že v časovém rozmezí 5 sekund od anotované události bylo správně detekováno minimum událostí a přesnost v této toleranci je pro téměř všechny modely menší než 0.05 % mAP.

Nejlepší vlastnosti pro větší časové tolerance vykazují modely z experimentů č. 3 a 4, které při toleranci $\delta = 60\,s$ dosahují přesnosti téměř 30 % mAP. Pro lepší detekci událostí ve smyslu určení přesného času jsou nejvhodnější modely z experimentů č. 3 a 5, kdy při toleranci $\delta = 10\,s$ mají přibližně o 5 % vyšší přesnost než ostatní modely.

4.7.3 Dotrénování

Dotrénování (angl. fine-tuning) je proces učení neuronové sítě, který se používá k přizpůsobení předtrénovaného modelu konkrétní úloze, v našem případě se jedná o doladění poskytnutého modelu tak, aby byl schopen detekovat události v zápasech v našem datasetu. Jako předtrénovaný model nám bude sloužit model `CALF_benchmark`, který je součástí GitHub repozitáře [38]. Tento model byl natrénován na datasetu SoccerNet-v2 [37] na původním počtu 17 tříd událostí a dosahuje hodnoty Average-mAP 40,7 %. Dotrénování tedy musí být provedeno na stejném počtu tříd, což znamená, že se nemůžeme omezit na menší počet tříd, jako v případě trénování od začátku. Model je blíže popsán v sekci 4.1.

Celkem jsme provedli 2 experimenty, ve kterých jsme zachovali stejné hodnoty parametrů β a λ^{seg}, jaké byly použity u nejlepších výsledků při trénování od začátku (experimenty č. 3 a 4). Nastavení K parametrů pro prvních 8 tříd definuje tabulka 4.7, parametry pro zbylých 9 tříd jsou popsány v tabulce 4.10.

K_i^c	Výkop	Střídání	Ofsajd	Střela na bránu	Střela mimo bránu	Nepřímý volný kop	Žlutá karta	Červená karta	Žl. karta -> Č. karta
K_1	-39	-16	-38	-5	-8	-10	-15	-33	-12
K_2	-19	-8	-19	-3	-4	-5	-7	-16	-6
K_3	19	6	19	3	4	5	7	16	6
K_4	39	12	38	5	8	10	15	33	12

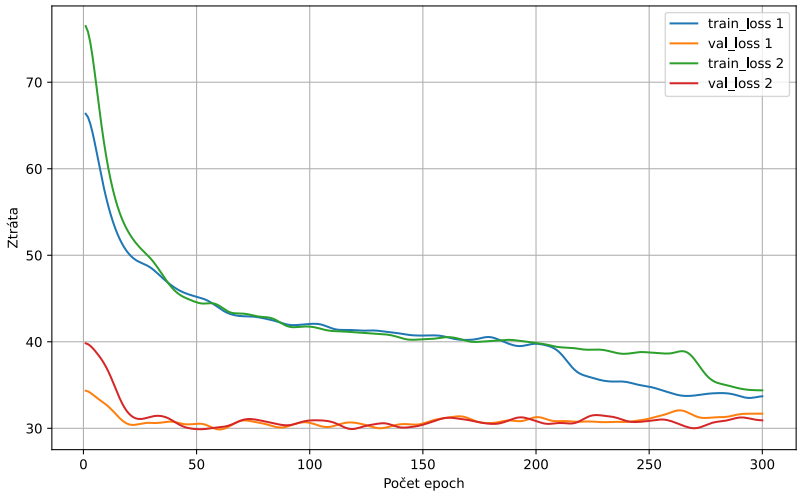
Tabulka 4.10: Nastavení K parametrů zbylých 9 tříd

Výsledky obou experimentů obsahuje tabulka 4.11. Podrobnější vypsání přesností jednotlivých tříd není již tak zajímavé a stejně jako při trénování od začátku mají vysoké hodnoty Average-mAP ty události, které se v datasetu vyskytují v největším počtu.

V obrázku 4.9 můžeme vidět, že trénovací ztráta má klesající charakter, stejně jako v případě trénování od začátku. To znamená, že model se snažil přizpůsobit

exp.	β	λ^{seg}	Avg.-mAP	Avg.-mAP visible	Avg.-mAP unshown
1	0.5	0.0005	0.17	0.20	0.04
2	0.5	0.001	0.17	0.17	0.05

Tabulka 4.11: Porovnání výsledků (dotrénování)



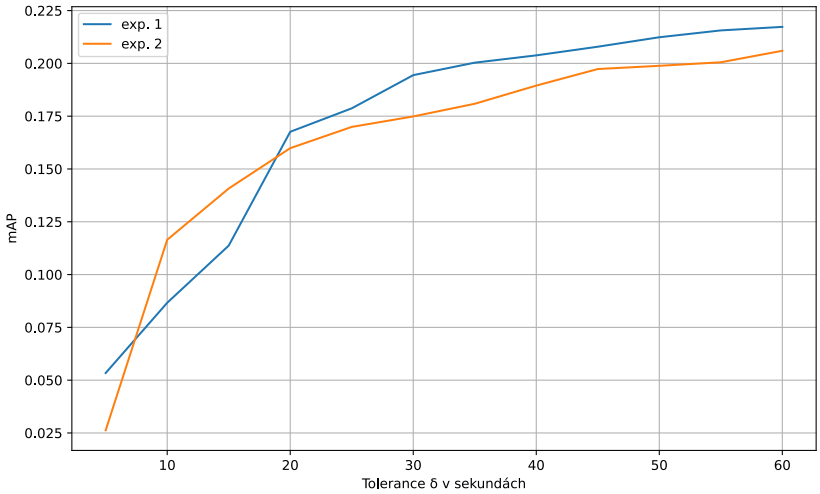
Obrázek 4.9: Průběh trénovací a validační ztráty (fine-tuning)

trénovacím datům. Průběh validační ztráty se však nepatrně liší. Prvních 25 epoch validační ztráta klesala, což je pro dobré výsledky žádoucí, poté však její průběh začal stagnovat. Z takového průběhu lze odvodit, že na začátku se použitý model správně přizpůsoboval zápasům v naší validační sadě, ale později pravděpodobně došlo k přeučení na trénovací sadě.

Průběh závislosti mAP na časové toleranci δ (viz obrázek 4.10) je pro dotrénované modely téměř shodný s výsledky při trénování od začátku. Většího rozdílu si můžeme všimnout při větších časových tolerancích (30 sekund a více), kdy výsledné modely mají přibližně o 5 % nižší hodnoty mAP než modely trénované od začátku.

4.7.4 Shrnutí

Celkem jsme provedli 6 experimentů pro trénování sítě od začátku a 2 experimenty pro dotrénování použitého modelu `CALF_benchmark`. Jak již bylo zmíněno na začátku této kapitoly, nemáme dostatečné množství dat k tomu, abychom naše výsledky mohli považovat za zcela spolehlivé. Počet výskytů některých událostí v našem datasetu dosahuje maximálně řádů jednotek a pro síť je složité se z takového



Obrázek 4.10: Závislost mAP na toleranci δ (fine-tuning)

množství naučit. Přesnost detekce některých událostí však dosahuje již poměrně přijatelných výsledků. Jedná se o události, které se obecně v porovnání s ostatními objevují v zápasech velmi často, a i přes malou velikost našeho datasetu vidíme v síti velký potenciál tyto události úspěšně detekovat. Konkrétně se jedná o události typu míč v zámezí, vhazování a kop od branky.

Lepší výsledky jsme očekávali při dotrénování modelu na našem datasetu. I když je povaha záznamů z televizních vysílání podobná s našimi záznamy, přesnost výsledné detekce není vyšší než při trénování od začátku. Důvodem může být rozdílná výška kamery nad hrací plochou, kvalita záznamu, rozdílné světelné podmínky, nebo vliv režie při televizních přenosech, kdy původní model mohl těžit z opakovaných nebo přiblížených záběrů.

Vzhledem k malému počtu testovacích dat a tedy i událostí obsažených v nich mohou být některé výsledky zkreslené. To můžeme pozorovat například u události penalta. Náš testovací set obsahuje jeden výskyt této události a pokud ji model správně detekuje, automaticky je přesnost její detekce vysoká, v opačném případě je rovna nule. Takový problém by vyřešil dataset s častým výskytem takových událostí.

Nemalý vliv na výsledky experimentů může mít také nastavení K parametrů, které ovlivňují výstup ztrátové funkce. Tyto parametry jsme odvodili na základě zkušenosti a jejich jiné nastavení by mohlo vést k lepším výsledkům.

Je zde velký prostor pro experimentování a získané výsledky nám naznačují, jakým směrem by se ladění parametrů modelu a hyperparametrů trénování mohlo dále ubírat.

Závěr 5

Tato práce se zabývala automatickou detekcí typu události ve videozáznamech fotbalových utkání. Práce obsahuje rozbor problematiky analýzy zápasů a detekce událostí ve fotbalu a popisuje některé současné technologie, které se takovou detekcí zabývají. Dále práce poskytuje rozbor teorie strojového učení, neuronových sítí, hlubokého učení a počítačového vidění, tedy metod a přístupů, které se v oblasti analýzy záznamů sportovních utkání často využívají.

Cílem praktické části bylo pochopit strukturu a fungování použitého modelu neuronové sítě zaměřeného na detekci událostí ve fotbalu a následně tento model aplikovat na zápasy týmů hrajících nižší fotbalové soutěže. Tento proces zahrnoval přípravu vstupních dat, jejich anotaci, extrakci příznaků a přizpůsobení trénovacích hyperparametrů a parametrů modelu pro definovanou úlohu. Na závěr jsme provedli několik experimentů a jejich vyhodnocení. Jejich výsledky nejsou příliš relevantní, vzhledem k nízkému počtu zápasů ve vytvořeném datasetu. I přesto jsme mohli některé výstupy považovat za přijatelné. V použité neuronové síti tak lze pozorovat velký potenciál vybrané události při větším počtu zápasu v datasetu a správném nastavení parametrů spolehlivě detekovat.

Práce navazující na tuto by se měly zaměřit na rozšíření vytvořeného datasetu a nalezení způsobů, jak by přesnost použitého modelu mohla být vylepšena. Tato práce poskytuje dobrý základ pro další výzkum v této oblasti a naznačuje, jakým směrem se dále vydat ve vývoji spolehlivého systému pro automatickou detekci událostí ve fotbale, který by bylo možné využít pro další úlohy, jako je například automatická generace sestřihů.

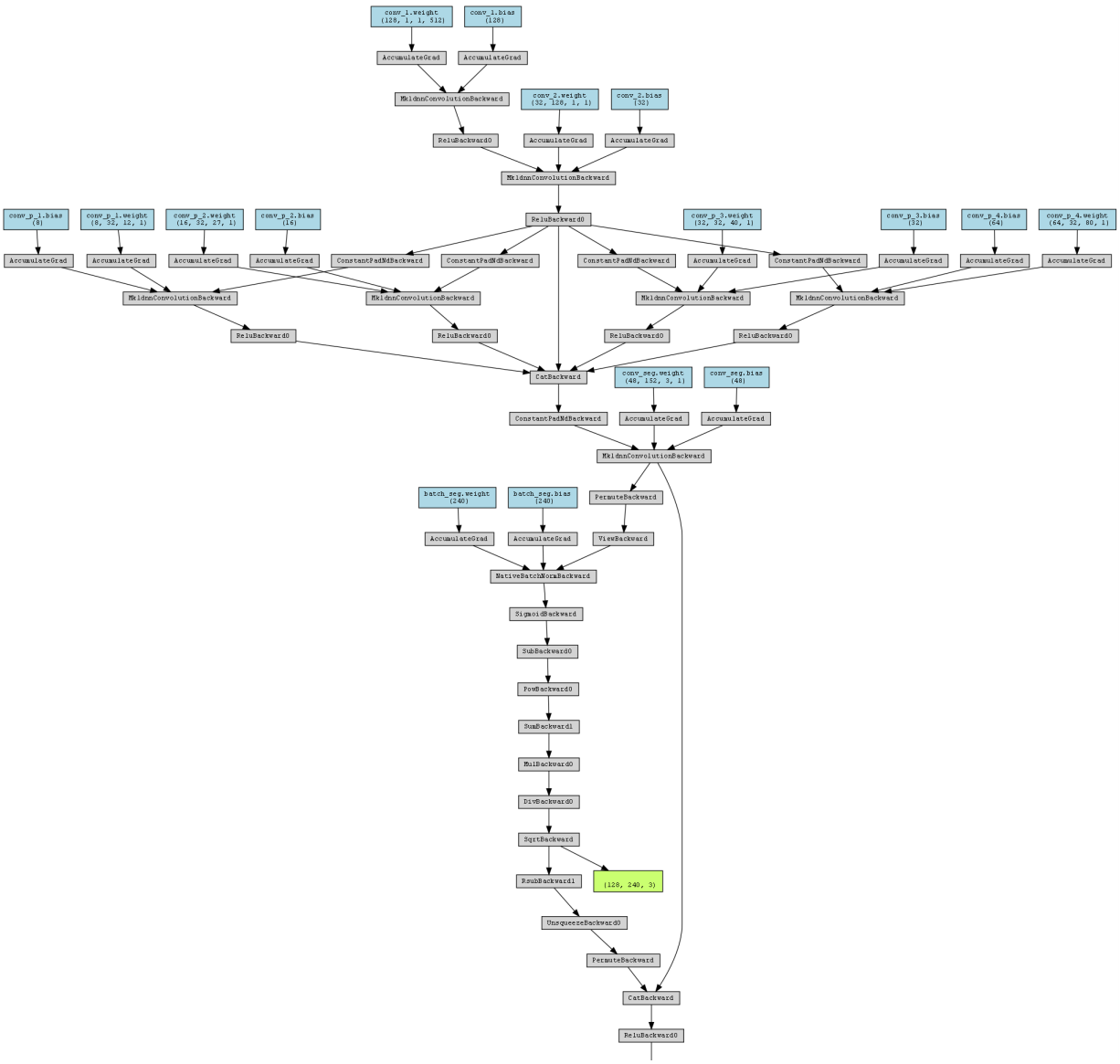
Seznam událostí a jejich popis

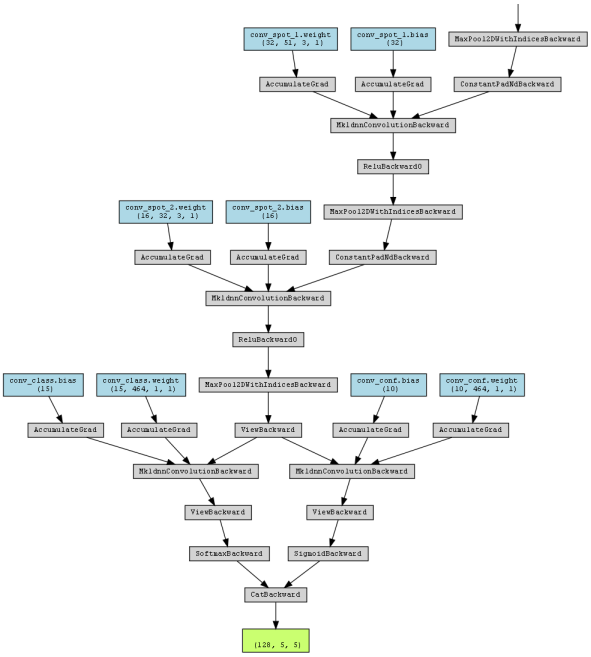
A

Událost	Angl.	Popis
Penalta	Penalty	Hráč střílí z pokutové značky (kope penaltu)
Výkop	Kick-off	Hráč zahajuje hru na začátku utkání nebo po vstřeleném gólu
Gól	Goal	Míč přejde přes brankovou čáru
Střídání	Substitution	Střídající hráč opouští hrací plochu a nastupuje hráč ze střídačky
Ofsajd	Offside	Asistent rozhodčího zvedá praporek pro signalizaci ofsajdu
Střela na bránu	Shots on target	Hráč vystřelí na bránu
Střela mimo bránu	Shots off target	Hráč vystřelí mimo bránu
Kop od branky	Clearance	Hráč nebo brankář vrací míč do hry kopem od branky
Míč v zámezí	Ball out of play	Míč opouští hrací plochu
Vhazování	Throw-in	Míč je vhozen hráčem do hry od autové čáry
Faul	Foul	Hráč se dopustí nedovolené hry
Nepřímý volný kop	Indirect free-kick	Hráč kope volný přímý kop s úmyslem rozehrát míč
Přímý volný kop	Direct free-kick	Hráč kope volný přímý kop s úmyslem dát gól (bránící tým postaví zeď)
Rohový kop	Corner	Hráč rozehrává míč od rohového praporku
Žlutá karta	Yellow card	Rozhodčí uděluje hráči žlutou kartu
Červená karta	Red card	Rozhodčí uděluje hráči červenou kartu
Žlutá karta -> červená karta	Yellow card -> Red card	Rozhodčí uděluje hráči červenou kartu po druhé žluté kartě

Architektura neuronové sítě

B




conv_spot_1.weight
(32, 51, 3, 1)
conv_spot_1.bias
(32)
MaxPool2DWithIndicesBackward
AccumulateGrad
AccumulateGrad
ConstantPadNdBackward
MkldnnConvolutionBackward
ReluBackward0
conv_spot_2.weight
(16, 32, 3, 1)
conv_spot_2.bias
(16)
MaxPool2DWithIndicesBackward
AccumulateGrad
AccumulateGrad
ConstantPadNdBackward
MkldnnConvolutionBackward
ReluBackward0
conv_class.bias
(15)
conv_class.weight
(15, 464, 1, 1)
MaxPool2DWithIndicesBackward
conv_conf.bias
(10)
conv_conf.weight
(10, 464, 1, 1)
AccumulateGrad
AccumulateGrad
ViewBackward
AccumulateGrad
AccumulateGrad
MkldnnConvolutionBackward
MkldnnConvolutionBackward
ViewBackward
ViewBackward
SoftmaxBackward
SigmoidBackward
CatBackward
(128, 5, 5)

Bibliografie

1. *HIGHLIGHTS! CITY & REAL ALL-SQUARE AFTER CHAMPIONS LEAGUE THRILLER | Real Madrid 3-3 Man City* [online]. [cit. 2024-04-15]. Dostupné z: https://www.youtube.com/watch?v=eWvGJ-W7uzg.
2. FIFA. *Semi-automated offside technology* [online]. [cit. 2024-04-15]. Dostupné z: https://inside.fifa.com/technical/football-technology/football-technologies-and-innovations-at-the-fifa-world-cup-2022/semi-automated-offside-technology.
3. *Livesport* [online]. [cit. 2024-04-15]. Dostupné z: https://www.livesport.cz/.
4. *Veo* [online]. [cit. 2024-04-15]. Dostupné z: https://launch.veo.co/.
5. *Panoris* [online]. [cit. 2024-05-13]. Dostupné z: https://www.panoris.com/.
6. VIDAL-CODINA, Ferran; EVANS, Nicolas; EL FAKIR, Bahaeddine; BILLINGHAM, Johsan. Automatic event detection in football using tracking data. *Sports Engineering*. 2022, roč. 25, č. 1, s. 18. ISSN 1460-2687. Dostupné z DOI: 10.1007/s12283-022-00381-6.
7. SOMWONG, Rungroj; HNOOHOM, Narit. Automatic Football Match Event Detection from the Scoreboard using a Single-Shot MultiBox Detector. In: *2019 14th International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP)*. 2019, s. 1–6. Dostupné z DOI: 10.1109/iSAI-NLP48611.2019.9045280.
8. VANDERPLAETSE, Bastien; DUPONT, Stéphane. *Improved Soccer Action Spotting using both Audio and Video Streams*. 2020. Dostupné z arXiv: 2011.04258 [cs.CV].
9. MERLER, Michele et al. Automatic Curation of Sports Highlights Using Multimodal Excitement Features. *IEEE Transactions on Multimedia*. 2019, roč. 21, č. 5, s. 1147–1160. Dostupné z DOI: 10.1109/TMM.2018.2876046.
10. MORRA, Lia et al. Slicing and Dicing Soccer: Automatic Detection of Complex Events from Spatio-Temporal Data. In: *Image Analysis and Recognition*. Springer International Publishing, 2020, s. 107–121. ISBN 9783030503475. ISSN 1611-3349. Dostupné z DOI: 10.1007/978-3-030-50347-5_11.

11. KHAUSTOV, Victor; MOZGOVOY, Maxim. Recognizing Events in Spatio-temporal Soccer Data. *Applied Sciences*. 2020, roč. 10, č. 22. ISSN 2076-3417. Dostupné z DOI: 10.3390/app10228046.

12. JIA, Yunke. Automatic Detection Algorithm of Football Events in Videos. *Computational Intelligence and Neuroscience*. 2022, roč. 2022, s. 2839244. ISSN 1687-5265. Dostupné z DOI: 10.1155/2022/2839244.

13. NORGÅRD RONGVED, Olav A. et al. Real-Time Detection of Events in Soccer Videos using 3D Convolutional Neural Networks. In: *2020 IEEE International Symposium on Multimedia (ISM)*. 2020, s. 135–144. Dostupné z DOI: 10.1109/ISM.2020.00030.

14. CIOPPA, Anthony et al. A Context-Aware Loss Function for Action Spotting in Soccer Videos. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

15. GIANCOLA, Silvio; AMINE, Mohieddine; DGHAILY, Tarek; GHANEM, Bernard. SoccerNet: A Scalable Dataset for Action Spotting in Soccer Videos. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2018. Dostupné z DOI: 10.1109/cvprw.2018.00223.

16. XARLES, Artur; ESCALERA, Sergio; MOESLUND, Thomas B.; CLAPÉS, Albert. *ASTRA: An Action Spotting TRAnsformer for Soccer Videos*. 2024. Dostupné z arXiv: 2404.01891 [cs.CV].

17. SHEIKH, Haroon; PRINS, Corien; SCHRIJVERS, Erik. Artificial Intelligence: Definition and Background. In: *Mission AI: The New System Technology*. Cham: Springer International Publishing, 2023, s. 15–41. ISBN 978-3-031-21448-6. Dostupné z DOI: 10.1007/978-3-031-21448-6_2.

18. SAMUEL, A. L. Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*. 1959, roč. 3, č. 3, s. 210–229. Dostupné z DOI: 10.1147/rd.33.0210.

19. MOHAMMED, Mohssen; KHAN, Muhammad; BASHIER, Eihab. *Machine Learning: Algorithms and Applications*. 2016. ISBN 9781498705387. Dostupné z DOI: 10.1201/9781315371658.

20. *Overfitting* [online]. [cit. 2024-05-05]. Dostupné z: https://www.mathworks.com/discovery/overfitting.html.

21. HAN, Jiawei; KAMBER, Micheline; PEI, Jian. *Data Mining: Concepts and Techniques*. 3rd. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. ISBN 0123814790.

22. VANDEGHEN, Renaud; CIOPPA, Anthony; DROOGENBROECK, Marc Van. *Semi-Supervised Training to Improve Player and Ball Detection in Soccer*. 2022. Dostupné z arXiv: 2204.06859 [cs.CV].

23. SARKER, Iqbal H. Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*. 2021, roč. 2, č. 3, s. 160. ISSN 2661-8907. Dostupné z DOI: 10.1007/s42979-021-00592-x.

24. YAN, X.; SU, X. *Linear Regression Analysis: Theory and Computing*. World Scientific, 2009. G - Reference,Information and Interdisciplinary Subjects Series. ISBN 9789812834102. Dostupné také z: https://books.google.cz/books?id=5pdpDQAAQBAJ.

25. ROSENBLATT, Frank. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*. 1958, roč. 65 6, s. 386–408. Dostupné také z: https://api.semanticscholar.org/CorpusID:12781225.

26. O'SHEA, Keiron; NASH, Ryan. *An Introduction to Convolutional Neural Networks*. 2015. Dostupné z arXiv: 1511.08458 [cs.NE].

27. HAQUE, Kh. Nafizul. *What is Convolutional Neural Network — CNN (Deep Learning)* [online]. 2023. [cit. 2024-05-05]. Dostupné z: https://www.linkedin.com/pulse/what-convolutional-neural-network-cnn-deep-learning-nafiz-shahriar.

28. SCHMIDT, Robin M. *Recurrent Neural Networks (RNNs): A gentle Introduction and Overview*. 2019. Dostupné z arXiv: 1912.05911 [cs.LG].

29. LEA, Colin; VIDAL, Rene; REITER, Austin; HAGER, Gregory D. *Temporal Convolutional Networks: A Unified Approach to Action Segmentation*. 2016. Dostupné z arXiv: 1608.08242 [cs.CV].

30. TURNER, Richard E. *An Introduction to Transformers*. 2024. Dostupné z arXiv: 2304.10557 [cs.LG].

31. QIU, Jin; LIU, Jian; SHEN, Yunyi. Computer Vision Technology Based on Deep Learning. In: *2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA)*. 2021, sv. 2, s. 1126–1130. Dostupné z DOI: 10.1109/ICIBA52610.2021.9687873.

32. GIRSHICK, Ross. *Fast R-CNN*. 2015. Dostupné z arXiv: 1504.08083 [cs.CV].

33. REDMON, Joseph; DIVVALA, Santosh; GIRSHICK, Ross; FARHADI, Ali. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. Dostupné z arXiv: 1506.02640 [cs.CV].

34. LONG, Jonathan; SHELHAMER, Evan; DARRELL, Trevor. *Fully Convolutional Networks for Semantic Segmentation*. 2015. Dostupné z arXiv: 1411.4038 [cs.CV].

35. HE, Kaiming; GKIOXARI, Georgia; DOLLÁR, Piotr; GIRSHICK, Ross. *Mask R-CNN*. 2018. Dostupné z arXiv: 1703.06870 [cs.CV].

36. *instance segmentation.png* [online]. [B.r.]. [cit. 2024-05-14]. Dostupné z: https://wiki.math.uwaterloo.ca/statwiki/index.php?title=File:instance_segmentation.png.

37. DELIÈGE, Adrien et al. SoccerNet-v2 : A Dataset and Benchmarks for Holistic Understanding of Broadcast Soccer Videos. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2021.

38. *SoccerNetv2-DevKit* [online]. [cit. 2024-04-30]. Dostupné z: https://github.com/SilvioGiancola/SoccerNetv2-DevKit/tree/main.

39. *SoccerNet* [online]. [cit. 2024-04-16]. Dostupné z: https://www.soccer-net.org/.

40. *odkazy na videozáznamy z utkání* [online]. [cit. 2024-04-30]. Dostupné z: https://www.fotbal.cz/souteze/subjekty/subjekt/249.

41. *LosslessCut* [online]. [cit. 2024-05-01]. Dostupné z: https://github.com/mifi/lossless-cut.

42. HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. *Deep Residual Learning for Image Recognition*. 2015. Dostupné z arXiv: 1512.03385 [cs.CV].

Seznam obrázků

Seznam tabulek