# Tartalomjegyzék

[Tartalomjegyzék 1](#_Toc149912460)

[1 Neurális hálók 4](#_Toc149912461)

[1.1 Perceptron 4](#_Toc149912462)

[1.2 Definició. (Mesterséges neuron) 4](#_Toc149912463)

[1.3 Aktivációs függvény 5](#_Toc149912464)

[1.3.1 Sigmoid: 5](#_Toc149912465)

[1.3.2 Softmax: 5](#_Toc149912466)

[1.3.3 ReLU: 5](#_Toc149912467)

[1.4 Neuron réteg 6](#_Toc149912468)

[1.4.1 Definíció. (Mesterséges neuronréteg) 6](#_Toc149912469)

[1.5 Előrecsatolt mesterséges neuronháló 6](#_Toc149912470)

[1.5.1 Definíció. (Előrecsatolt mesterséges neuronháló) 6](#_Toc149912471)

[1.6 Veszteségfüggvény 6](#_Toc149912472)

[1.7 A modell tanulása gradiens ereszkedés módszerrel 7](#_Toc149912473)

[1.7.1 Definíció(Gradiens): 7](#_Toc149912474)

[2 Konvolúciós Neurális hálók 9](#_Toc149912475)

[2.1 Biologiai vonatkozat 9](#_Toc149912476)

[2.2 Konvoluciós réteg 10](#_Toc149912477)

[2.3 Pooling 13](#_Toc149912478)

[3 2D póz felismerés 15](#_Toc149912479)

[3.1 Mask R-CNN [3] 15](#_Toc149912480)

[3.1.1 Faster R-CNN [4] 16](#_Toc149912481)

[3.2 Roi Align 17](#_Toc149912482)

[3.3 Fejhálózatok 18](#_Toc149912483)

[3.4 Póz felismerés optikai áramlással [5] 19](#_Toc149912484)

[3.5 Póz hasonlóság felismerése 20](#_Toc149912485)

[4 2 dimenzióból 3 dimenzióba emelés [6] 23](#_Toc149912486)

[4.1 Ideiglenes konvolúciós hálók 24](#_Toc149912487)

[4.1.1 Alkalmi konvolúció 24](#_Toc149912488)

[4.1.2 „Táguló” ideiglenes konvolució 24](#_Toc149912489)

[4.2 Modell architektúrája 25](#_Toc149912490)

[4.3 Félig-felügyelt megközelítés 26](#_Toc149912491)

[4.3.1 Leképzés 27](#_Toc149912492)

[4.3.2 Csont hossz L2 veszteség 28](#_Toc149912493)

[4.3.3 Kamera Paraméterek 28](#_Toc149912494)

[5 Alkalmazás 29](#_Toc149912496)

[6 Irodalomjegyzék 32](#_Toc149912497)

Bevezetés

Szakdolgozatom tárgya az emberi pózfelismerő rendszerek alkalmazása a sportban és ezen belül a rövidpályás gyorskorcsolyában.

Szerettem volna ezekből a rendszerekből a jelenlegi edzésmunkát ténylegesen segítő információkat kinyerni. Sajnos a mai 3D-s emberi pózfelismerők közül a pontossága egyelőre csak a több kamerával több szögből felvett rendszereknek van meg. Illetve a sportban napjainkban főleg a ruhákba varrt GPS és egyéb jeladókkal segített rendszerek dominálnak.

Nekem többszöri próbálkozás ellenére sem sikerült a kamerakalibrálás akadályainak legyőzése ezért egy kevésbé pontos viszont a gyakorlatban jóval egyszerűbb megoldást választottam. Ez a program több már jól működő rendszert egyesít vagy finomít, amiket a későbbiekben ismertetek

1. fejezet

# Neurális hálók

„A madarak inspiráltak minket a repülésre, a bojtorján növények a tépőzárat, és a természetinspirált rengeteg más találmányt is, logikusnak tűnik tehát megvizsgálni az agy architektúráját egy intelligens gép megépítéséhez. Ez a kulcsgondolat, amely inspiráltmesterséges neurális hálózatokat (angolul ANN-ek). Bár a repülőket madarak ihlették, nem kell csapkodniuk a szárnyukkal. Hasonlóképpen, az ANN-ek fokozatosan meglehetősen eltértek biológiai unokatestvéreiktől. Egyes kutatók még amellett is érvelnek, hogy le kell mondanunk a biológiai analógiával teljes egészében, nehogy kreativitásunkat a biológiailag elfogadható rendszerekre korlátozzuk. Az ANN-ek képezik a mélytanulás lényegét. Sokoldalúak, erősek és nagy teljesítményűek, így ideálisak a nagy és rendkívül összetett gépi tanulási feladatok megoldására, mint például több milliárd kép osztályozása (pl. Google Images), beszédfelismerési szolgáltatások (pl. Apple Siri), több millió felhasználónak ajánljanak videókat (pl. YouTube), vagy megtanuljak legyőzni a világbajnokot a Go játéknál több millió múltbeli játék megvizsgálásával.” - idézet a [1] könyvből, a továbbiakban használt definíciók és fogalmak pedig főleg a [2] tartalmaira építenek.

## Perceptron

A Perceptron az egyik legegyszerűbb ANN architektúra, amelyet Frank Rosenblatt talált fel 1957-ben. Ezekből lettek később a „mesterséges neuronok”, amik a mesterséges neuronháló alap építő kövei.

## Definició. (Mesterséges neuron)

Egy leképzést mesterséges neuronnak nevezünk és felírhatjuk

alakban, ahol egy aktivációs függvény bemeneti érték az ehhez tartozó súly es az eltolás (bias).

## ****Aktivációs függvény****

**A mesterséges neurális hálózat neuronjainak aktiválási függvénye egy olyan függvény, amely kiszámítja a csomópont kimenetét (a bemenetei, az egyes bemenetek súlya es az eltolás alapján). A nem triviális problémákat csak nemlineáris aktiválási függvény segítségével lehet megoldani. Ezek közül pár amit napjainkban is használnak.**

### Sigmoid**:**

#### 

**Sajnos nem mindig megfelelő mivel szélsőséges bemeneti értékekre a függvény értékünk 1-hez vagy 0-hoz közelít ahol a derivált majdnem 0 (ezt majd a későbbiekben latjuk hogy miért nem jó).**

### ****Softmax**:**

**ahol es**

**A softmax függvény, más néven softargmax  vagy normalizált exponenciális függvény,  K valós számból álló vektort K lehetséges kimeneteli valószínűség eloszlására konvertál.** **Ez a logisztikus függvény általánosítása több dimenzióra A softmax függvényt gyakran használják egy neurális hálózat utolsó aktiváló függvényeként, hogy a hálózat kimenetét a várható kimeneti osztályok közötti valószínűségi eloszlásra normalizálják.**

### ****ReLU**:**

**A mai neurális halók alap értelmezett függvénye , mert egyszerű vele számolni és az elotte használt sigmoid és hiperbolikus tangens függvényeket nehezebben lehet igazán mély neurális hálókban használni az eltűnő gradiens probléma miatt.**

## ****Neuron réteg****

**A fent említett mesterséges neuronokat, mesterséges rétegekbe rendezzük ezek segítségével definiálható leképezés, ahol n a bemeneti vektor hossza m pedig a neuronok száma.**

### ****Definíció. (Mesterséges neuronréteg)****

**a vektort mesterséges neuronréteg értekéi nevezzük, ahol egy adott aktivációs függvény mátrix a neuronok súlyait tartalmazza, a bemeneti vektor pedig az eltolásvektor. Itt a leképezés eredményé egy -hosszúságú vektor.**

## ****Előrecsatolt mesterséges neuronháló****

**Ezen rétegek egymásutánjából kaphatunk egy neurális hálót melynek legegyszerűbb és leggyakoribb modellje az előre csatolt neurális hálók, ahol a neuronrétegeket függvény kompozícióval egymásután kapcsoljuk.**

### ****Definíció**. (**Előrecsatolt** **mesterséges** **neuronháló**)**

**ahol a neuron rétegek paramétereit tartalmazó vektor pedig a hálózat rétegéit reprezentáló függvény.**

**Egy két extra indexet bevezetve egy mesterséges neuronunk kimenetét a következőképp kapjuk**

**ahol az -edik réteg -edik neuronja.**

## ****Veszteségfüggvény****

**A veszteségfüggvény az a függvény, amely kiszámítja az algoritmus aktuális kimenete és a várt kimenet közötti távolságot. Ez egy módszer annak értékelésére, hogy az algoritmus hogyan modellezi az adatokat. Akkor teljesít jól a modellünk, ha ennek a függvénynek az érteké minél kevesebb. Ezekből az egyik ismertebb az L2 loss függvény**

**ahol, az elvart érték es az utolsó -edik réteg megfelelő indexű kimeneti értéké.**

## ****A modell tanulása gradiens ereszkedés módszerrel****

A gradiens ereszkedés egy nagyon általános optimalizálási algoritmus, amely képes megtalálni az optimális megoldást sokféle problémára. A gradiens ereszkedés általános elképzelése az, hogy a paraméterek iteratív módosítjuk és ezzel minimalizáljuk a költség függvényt.

### Definíció(Gradiens):

A vektorszámításban egy skaláris értékű differenciálható több változós függvény gradiense. Az a vektormező mely egy pontra megadja a leggyorsabb növekedés irányát. Ha egy függvény gradiense egy pontban nem nulla a gradiens nagysága az adott irányú növekedés mértéke.

Ha =  **ahol a paraméterek akkor a gradiens a következő:**

A képen Betűtípus, szöveg, kézírás, diagram látható

Automatikusan generált leírás

**A gradiens megmutatja hogy milyen „irányban” változtassunk a paramétereinken es hogy aranyaiban milyen mértékben ,de magát a lépés nagyságát mi választjuk meg es egy konkrét lépés a következőképpen írható le**

A képen Betűtípus, szöveg, fehér, tipográfia látható

Automatikusan generált leírás

ahol θ a hálózat a paramétereit tartalmazó vektor, λ a lépés mérete, L pedig a veszteségfüggvény.

**Ezekkel a lepesekkel megtalálhatjuk a függvényünk lokális minimumát, ami persze nem feltétlen jelent jó megoldást. Szerencsere erre a gyakorlatban rengetek megoldás létezik** mint pl: a sztochasztikus gradiens ereszkedés vagy a „mini-batch” gradiens ereszkedés, ahol random tanuló elem vagy elem-csoport választásával próbálunk kikerülni a lokális minimumokból.

2.fejezet

# Konvolúciós Neurális hálók

## Biologiai vonatkozat

1996-ban már egy szuperszámítógép megverte az akkori sakk világbajnok Garry Kasparovot még is nagyon sokáig kellett várni rá hogy megbízhatóan tudjon képekről objektumokat azonosítan vagy hang alapján szavakat felismerni. De miért ilyen egyszerű ez számunkra? Ezek a érzékelések számunkra a tudatalattinkban történik és mire eljut az információ a tudatunkig nem tudjuk nem felismerni az autót , kutyát vagy akár egy közlekedési-lámpát.

A konvoluciós neurális hálók ötlete egy tanulmányhoz köthető mely az agyunk vizuális kérgét tanulmányozta és már az 1980-as évektől használták képfelismeréshez. Az utóbbi pár évben köszönhetően az exponenciális növekedésnek a számítási kapacitásban és a különböző tanító adatok mennyiségében. Valamint egyéb a szakmában elterjedt megoldásoknak hála mára már rengeteg helyen használják pl: önvezető autókban, kép-keresésben, automatizált videó kategorizálásban és más területeken is mint hangfelismerés és nyelvi modellek építésében. Mi csak a vizuális felhasználásával fogunk foglalkozni.

David H. Hubel és Torsten Wiesel egy sor kísérletet végzett macskákon 1958 és 1959 (és néhány évvel később a majmokon), amely döntő betekintést nyújt a látókéreg szerkezetébe (a szerzők élettani Nobel-díjat ill. Orvostudomány Nobel-díjat kaptak 1981-ben munkájukért). Különösen azt mutatták ki, hogy sok idegsejtnek a látókéregben van egy kis lokális receptív mezője, vagyis csak a látómező egy korlátozott területén elhelyezkedő ingerekre reagálnak. A különböző neuronok mezői átfedhetik egymást, és együtt a teljes látóteret lefedik.

Ezenkívül a szerzők kimutatták, hogy egyes neuronok csak a vízszintes képekre reagálnak, míg mások csak a különböző orientációjú vonalakra reagálnak (két neuron lehet azonos receptív mezővel rendelkeznek, de eltérő vonalirányokra reagálnak). Ők is észrevették, hogy egyes neuronok nagyobb befogadó mezőkkel rendelkeznek, és jobban reagálnak komplex mintákra, amelyek az alacsonyabb szintű minták kombinációi. Ezek a megfigyelések ahhoz az elképzeléshez vezetettek, hogy a magasabb szintű neuronok a szomszédos alacsonyabb szintű idegsejtek kimenetein alapulnak. Ez a nagy teljesítményű architektúra képes mindent bonyolult mintát észlelni a látómező bármely területén.

A képen szöveg, diagram, képernyőkép, sor látható

Automatikusan generált leírás

Ez a tanulmány ihlette az 1980-ban bevezetett neokognitront, amelyek fokozatosan az általunk ma ismert konvolúciós neurális hálózatokká fejlődtek. Egy fontos mérföldkő volt az 1998-as tanulmány Yann LeCun, Léon Bottou, Yoshua Bengio, és Patrick Haffner által, amely bemutatta a híres LeNet-5 architektúrát, hogy felismerje a kézzel írt csekkszámokat. Ennek az architektúrának van néhány építőeleme amelyeket már ismerünk, mint például a teljesen összekapcsoltneurális háló és a szigmoid aktiválási funkció de két új építőelemet is bevezettek: a konvolúciós rétegeket és az úgynevezett „poolingot”.

## Konvoluciós réteg

A CNN legfontosabb építőeleme a konvolúciós réteg. Az első konvolúciós réteg neuronjai nem kapcsolódik a bemeneti kép minden egyes pixeléhez (mint az előző fejezetben), csak a pixelekhez a befogadó mezőjükben. A második konvolúciós rétegben minden neuron megint csak az előző réteg kis részére koncentrál. Ez az építészet lehetővé teszi a hálózat számára, hogy az első rejtett réteg alacsony szintű részleteit koncentrálja és a következő réteg ezekből építsen komplexebb információt és így tovább. Ez a hierarchikus struktúra gyakori a valós képeken, ez az egyik oka miért működnek olyan jól a CNN-ek a képfelismerésben.

Egy adott réteg i. sorában és j. oszlopában elhelyezkedő neuron az előző réteg } neuronjainak kimenetével van kapcsolatban, ahol a befogadó mező hossza és szélessége. Annak érdekében, hogy egy réteg magassága és szélessége ugyanolyan legyen, mint az előző rétegé, megszokott, hogy nullás értékeket adjunk a bemenet vagy réteg széléhez.

A képen sor, diagram, tervezés, kocka látható

Automatikusan generált leírás

Lehetőség van arra is, hogy egy nagy bemeneti réteget egy sokkal kisebb réteghez kapcsoljunk a befogadó mezők eltolásával. A két egymást követő receptív mező közötti távolságot lépésnek nevezzük. Az ábrán egy 5 × 7-es bemeneti réteg (plusz nulla párnázás) egy 3 × 4-es réteghez van csatlakoztatva, 3 × 3 receptív mezővel és egy 2-es lépéssel (ebben a példában a lépés mindkét irányban azonos, de nem kell, hogy így legyen). A felső réteg i.sorban és j. oszlopában található neuron az előző réteg oszlopaiban található neuronok kimeneteihez kapcsolódik, ahol a függőleges és vízszintes lépések. A képen sor, diagram, tervezés látható

Automatikusan generált leírás

A fentiekben receptív mezőnek nevezett területet nevezik még kernelnek vagy filternek és bár vannak hiperparaméterek amiket a modellező állit be ( kernel mérete, párnázás és lépésköz) a modellünk most a kernelben lévő mezők súlyát próbálja optimalizálni. Ezen paraméterek állitgatásával tudunk különböző információkat kinyerni egy egy képről. Egy egyszerűbb példát nézve ha a kernelünk középső oszlopa csupa egyes míg minden más 0 azzal a vertikális mintákat vagy vonalakat emeljük ki míg ha a középső sor 1 és és minden más 0 akkor a horizontális éleket hangsúlyozzuk

A képen képernyőkép, szöveg, fésű, fekete-fehér látható

Automatikusan generált leírás

Eddig az egyszerűség kedvéért minden konvolúciós réteget 2D-ben ábrázoltunk, de a valóságban több azonos méretű kis rétegből állhatnak, tehát pontosabb, ha 3D-ben ábrázoljuk őket. Egy szinten belül minden neuron ugyanazokat a paramétereket (súlyokat és bias-t) kapják, de szinten belüli rétegek kaphatnak különböző paramétereket tehát különböző kernellel vagy filterrel dolgoznak. Egy neuron receptív tere megegyezik a korábban leírtakkal, de kiterjed az összes előző szint rétegére. Röviden: egy konvolúciós szint egyszerre több szűrőt alkalmaz a bemeneteire, így képes érzékelni több jellegzetességet.

A képen szöveg, diagram, képernyőkép, sor látható

Automatikusan generált leírás

Ehhez tartozó matematikai képlet a következőA képen Betűtípus, szöveg látható

Automatikusan generált leírás

• az *i*. sorban és *j*. oszlopban lévő neuron kimenete a *k*-adik rétegben az *l*-edik szinten

• A korábban már említett a vertikális és horizontális lépésköz a kernel hossza és szélessége és a rétegek száma az *l*-1 edik szinten.

• a kimenete az l-1 edik szinten lévő neuronnak az

rétegben.

• a bias az l-edik szinten a k-adik rétegben ezzel lehet állitani a réteg „fényességét”.

• pedig a kapcsolódó súly a k-.adik rétegben az l-edik szinten és a bemenete az u-adik sorban és v-edik oszlopban van a rétegben.

## Pooling

Csakúgy, mint a konvolúciós rétegekben, a pooling rétegben minden neuron kapcsolódik az előző rétegben lévő neuronokkal, amelyek a kernelben vannak. Itt is meghatározhatjuk a a méretét, a lépéshosszt és a párnázás típusát a kernelnek, csak úgy mint ezelőtt. Az egyesülő neuronoknak azonban nincs súlya, mindössze egy aggregációs függvényt alkalmazunk az elemekre, mint például a max vagy az átlag. A példában egy max függvényt alkalmazunk a gyakorlatban ez a legelterjedtebb. Ez nyilvánvalóan egy nagyon pusztító réteg: még egy apró 2 × 2-es kernellel és egy 2-es lépéssel a kimenet kétszer kisebb lesz mindkét irányban (tehát a területe négyszer kisebb), egyszerűen kiesik a bemeneti értékek 75%-a.

A képen képernyőkép, sor, ég, ház látható

Automatikusan generált leírás

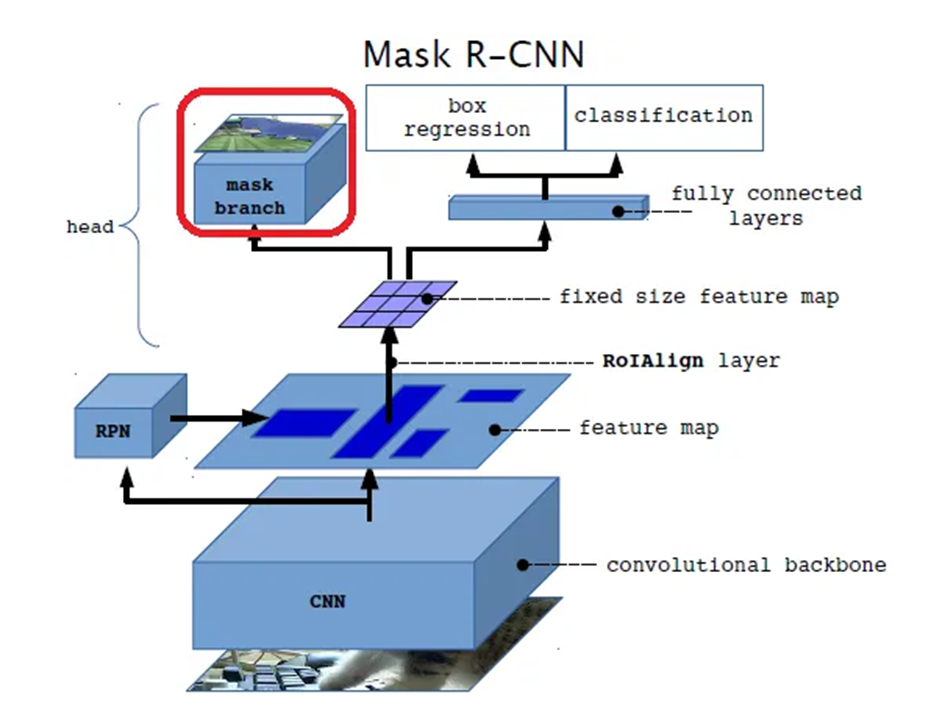
3. fejezet

# 2D póz felismerés

## Mask R-CNN [3]

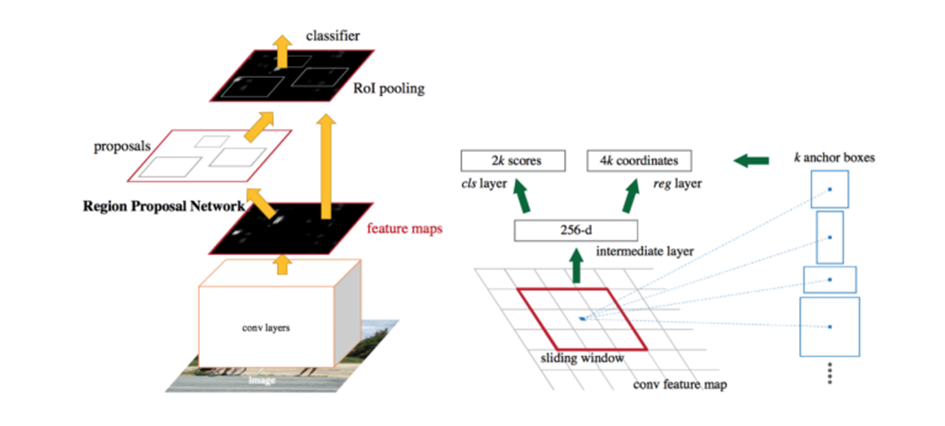
A következő részben már a kísérletben használt modell elemeit mutatom be. Első lépésben meg kell kapjuk a képen elhelyezkedő emberek fontosabb ízületeinek koordinátáját és az emberekhez tartozó „határoló dobozt”(boundingbox vagy BB) ha esetleg több ember is lenne a képen.

Ehhez a mi modellünk a Mask R-CNN architektúrát használja kisebb módosításokkal. A Mask R-CNN megértéséhez elég megérteni a Faster R-CNN működését, ami 2 elődjét (R-CNN, Fast R-CNN) váltotta mivel ezek az elején számításban és memóriahasználatban sem teljesítettek jól.



1. ábra Mask R-CNN architekrúra

### Faster R-CNN [4]

Először egy szabványos főbb jellemzők kivonására alkalmas CNN-t (általában ResNet verziót) alkalmaznak, hogy tömörebb formában kapjuk meg az információt. Ezután ennek a hálózatnak az utolsó még nem teljesen kapcsolódó rétege egy „tereptérkép” kerül felhasználásra a régiójavaslathoz és a végső észleléshez is. A régiójavaslat részben egy neurális hálózat "csúszik végig" a tereptérképeken a térképek n × n méretű ablakát bemenetként véve. Ez a hálózat prediktál mindegyik ablakra egy *cls*(„classification” vagy osztályozó pontszám az objektum osztályához) ∈ R 2k és egy *reg*(regressziós pontszám a határoló dobozokhoz) ∈ R 4k pontszámot. Itt k az ún. horgonyok. A tereptérképeken minden n × n ablakhoz tartozik egy megfelelő terület (látómező) az eredeti képen, amely hatással van a tereptérkép adott régiójára. A horgonyok ezen látómező területeinek transzformációi egy paraméterrel a méretskálázáshoz és egy a magasság/szélesség aránnyal. Így a modell javaslatot tud tenni érdekes régiókra különböző méretben és arányban. Az edzés során a kimeneti *cls* és a *reg* pontszámokat veszteségfüggvényen keresztül hasonlítják össze a kapott címkékkel a következő módon: Minden alap-igazság objektum határolódobozhoz rendelünk egy vagy több horgonyt (olyanokat, amelyeknek elég magas a metszés/unió arányuk - nagyobb mint 0.7). Horgonyok amelyek egy objektumhoz vannak rendelve, 1-es *cls* cimkét kapnak, míg mások 0-t. Hasonlóképpen a *reg* címkék a horgonyokhoz az objektum alapján vannak hozzárendelve (ha van ilyen obkjektum). Ezek a reg címkék lesznek a határoló doboz paraméterei egy horgonyhoz rendelt objektumról. A végső modellben kiválasztják a legjobb N (mi esetünkben 100) *cls* pontszámot és a hozzájuk tartozó *reg* értékek lesznek a határolódoboz koordináta javaslatai. A javasolt határolódobozok alapján RoI-k(Regional of Interest vagy magyarul „érdekes területek”)lesznek kivágva a tereptérképből mint m × m képek. RoI-k és a tereptérképünk általában nincsenek tökéletesen összhangban. 

2. ábra Faster R-CNN

## Roi Align

RoI-k kiválasztása RoI Align módszerrel. A fő különbség az elődök által használt RoI Pooling és a Mask R-CNN által használt RoI Align között a kvantálás. A RoI Align nem használ kvantálást az adatgyűjtéshez. Tudjuk, hogy a Fast R-CNN kétszer alkalmazza a kvantálást. Először a leképezési folyamatban, másodszor pedig a pooling folyamatban. Igy az itt kapott határoló dobozok nem lesznek mindig összhangban nem fogják pontosan határolni az objektumot ,ami nem mindig akkora probléma de ha „maszk”-ot akarunk rajzolni az objektumunk köré akkor igen. Ezt áthidalhatjuk úgy hogy nem szorítkozunk a természetes számokra és az objektumunk határait racionális számokkal írjuk le majd az így kapott nem teljesen illeszkedő hálón bilineáris interpolációt hajtunk végre. És így kapunk pontosabb határoló dobozokat.



3. ábra RoI Align kvantálás nélkül bilineáris interpolációval

## Fejhálózatok

Más fejhálózatokat alkalmazunk a RoI-kon az objektum osztályozásához és a maszk elkészítéséhez (1. ábra) (vagy szemantikus szegmenálásához). Ezt egy újabb kis konvolucios háló végzi el (4. ábra). Ami minden lehetségeshez osztályhoz jósol maszkokat a RoI-k számára (ez a mi esetünkben csak egy objektumot jelentene), de a veszteségfüggvényben csak a megjósolt osztály maszkját vesszük figyelembe.

Ezt az architekturát könnyen ki lehet szélesíteni emberi ízületi koordináták detektálására is. Úgy modellezünk egy izületet mint egy maskot és K izülethez K maskot készitünk. Változtatunk egy kicsit a szegmentációs folyamaton. Minden K ízülethez lesz egy -es tanulási cél ahol csak 1 pixel van megjelölve „előtérnek”. A tanitás során minden látható tényleges kulcspontnak minimalizáljuk a „cross-entropy” veszteségét egy fajtájú szoftmax kimeneten(ami arra sarkal hogy detektáljuk azt az egy pontot), de itt még minden K kulcspont külön van kezelve. Itt egy ResNet-FPN (4. ábra) variánst használunk. A kulcspont detektáláshoz használt fejrendszer egy 3x3-as 512-dimenzios konvolucios réteg egy dekonvolucios réteggel követve és 2x -es bilineáris felskálázással egy 56x56 -os halót hagyva. Mivel ez jobban használható ízületi detektálásra

A képen szöveg, diagram, sor, Betűtípus látható

Automatikusan generált leírás

4. ábra Konvolúciós fejhálózat az ízületetek detektálásához

A képen szöveg, képernyőkép látható

Automatikusan generált leírás

5. ábra Optikai áramlás

## Póz felismerés optikai áramlással [5]

A következőkben ismertetem hogy, a használt modell hogyan biztosít még jobb detektálást speciálisan videofelvételekre. A többszemélyes pózkövetés a videókban először megbecsüli az emberi pózokat a határoló dobozokban, majd követi ezeket a dobozokat vagy „embereket” egy egyedi azonosítóval (*id*) a képkockákon át. Egy embert úgy jelölünk hogy, ahol az test ízületeinek koordinátai az *id* pedig a fent említett azonosító. Amikor feldolgozzuk a -adik képet , már feldolgoztuk az embereket az képen . Az képen feldolgozott emberekhez pedig *id*-ket rendelünk úgy hogy es a peldanyok az es képen. Ha egy az aktuális kép kockán hozza van kötve a példányhoz az képen akkor hozza lesz kötve az -hez, ha nem új *id*-t kap jelezve egy új embert. A mi modellünk kép kockánkként 2 szett határoló dobozt fog kreálni egyet Mask R-CNN segítségével és egyet az előző kép kockák segítségével és egy úgynevezett optikai áramlás segítségével majd ezeken alkalmaz egy párosító algoritmust hogy legjobban észlelje az embereket.

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírásAlkalmazni egy egyetlen kép kockára optimalizált rendszert videók esetén sokszor hiányos vagy téves észleléshez vezethet a videókban tapasztalt elmosodás miatt. A fenti 5-ös ábrán a detektor a gyors mozgás miatt kihagyja a bal oldali fekete ruhás személyt. Az időbeli információkat gyakran használják fel robusztusabb észlelések létrehozására. Egy megoldás erre, hogy a közeli képekből állítsunk elő határoló dobozokat az éppen feldolgozó kép számára optikai áramlással. Adott egy ember ízületi koordinátákkal a képen es az optikai áramlás mező az es képek között , megtudnánk becsülni az ízületi koordinátákat az képen ha vesszük es alkalmazzuk rá az áramlási mezőt. Amikor nehéz embert detektálni egy képkockán akkor is tudnánk hozzá rendelni határoló dobozt ha valamelyik előző kép kockán helyesen volt detektálva. Mint az ábrán látható 2(c), a bal oldali fekete ruhás személyt nem detektálta a rendszer de az optikai áramlással tudunk hozzá határoló dobozt rendelni.

. ábra Pózkövetési algoritmus jelölései

## Póz hasonlóság felismerése

Határoló dobozok IoU (Intersection-over-Union vagy magyarul „metszet/unio”) tulajdonságát használni hasonlósági mérőszámként () problémás lehet ha egy ember túl gyorsan mozog es ezért a dobozok nem feddik egymást eléggé, és a zsúfolt jelenetekben, ahol a dobozok esetleg nem rendelkeznek megfelelő kapcsolattal a tényleges emberrel. Egy finomabb mérőszám lehetne a póz ha

sonlóság (), amely kiszámítja a test ízületeinek távolságát kettő objektum között, a módszer neve (OKS – Object Keypoint Similarity „objektum kulcspont hasonlóság”). Persze a póz hasonlósággal is lehet probléma, ha túl gyorsan változik vagy esetleg rossz érzékelés miatt. Javasoljuk egy áramlásalapú póz hasonlósági mérőszám használatát.

Adott egy példány az -ik képen es egy a az képen a kettejük áramlási hasonlósága a következő

ahol OKS reprezentálja két pozíció közti különbséget , ahol a jósolt ízületek -nak az képről az képre az optikai áramlás mezőt használva

Más emberekkel vagy tárgyakkal való takarás miatt az emberek gyakran eltűnnek és újra megjelenni. Ezért nem mindig elég két egymást követő kép kockát nézni, így megvan az áramlás alapú póz hasonlóság több képkockán át is, amelyet -flow-nak jelölünk, jelentése a propagált több korábbi képkockából származik. Ily módon újra linkelhetjük azokat a példányokat, amelyek a közbülső képeken eltűnnek.

Az optikai áramlást és az áramlás alapú pózhasonlóságot használva kapjuk pózkövető algoritmust, amely ezt a kettőt kombinálja, ahogy a 7.ábrán prezentálva van a 6.ábrai jelölésekkel. Először is megoldjuk a pózbecslési feladatot. Egy képkockán a videóból, egyesítjük a detektor által generált *BB(Bounding Box vagy „határoló doboz”)*-okat és az előző képkockákból az optikai áramlás segítségével generált *BB*-okat ezek a generált *BB*-ok egyfajta kiegészítésként szolgálnak az esetleg a detektor által nem észelt BB-okhoz. Ezután a kivágott és átméretezett képek alapján megbecsüljük az emberi pózt ezekkel a dobozokkal a 2. szakaszban javasolt pózbecslési hálózatunkon keresztül.

Másodjára megoldjuk a követési problémát Eltároljuk a követett példányokat egy dupla végű sorba fix hosszall ahol

ahol a kovetett peldanyt jelenti az kepen es Q- hossza ahány előző képet veszünk figyelembe a párosításhoz.

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

. ábra Pózkövető algoritmus

4.fejezet

# 2 dimenzióból 3 dimenzióba emelés [6]

Ebben a részben egy lehetséges megoldást ismertetünk a fent említett problémára, mely nem feltétlen egyértelmű hiszen egy 2 dimenziós pózhoz többféle 3 dimenziós tényleges pozíció is tartozhat. A korábbi munkák ezt a kétértelműséget az időbeli információk modellezésével kezelték visszatérő neurális hálózatokkal. Másrészt a konvolúciós hálózatok az utóbbi időben nagyon sikeresek az időbeli információ modellezésben olyan feladatokban, amelyeket hagyományosan RNN-kel oldottak meg, mint például a neurális gépi fordítás, nyelvi modellezés, beszédgenerálás és beszéd felismerés. A konvolúciós modellek lehetővé teszik több képkocka párhuzamos feldolgozását, ami visszatérő hálózatok esetén nem lehetséges.

A modell két részre bontható az elsőben egy „Ideiglenes konvolúciós” (Temporal Convolution) hálózattal állítunk elő 2D ízületi koordinátákból és határoló dobozokból 3D koordinátákat és a tanitáshoz egy olyan adathalmazt használunk mely rendelkezik tényleges 3D-s „igazi” koordinátákkal 2D-s képekhez. A modell másik részében felhasználunk előre nem fel címkézett adatokat is és így „félig felügyelt tanitással” is tudjuk segíteni a modell pontosságát. Ez azért előnyös mert nem rendelkezünk még túl sok 3D-s felcímkézett adathalmazzal és ezek előállítása is igen körülményes. Ezt a modellt a ciklus konzisztencia ihlette a nyelvi fordításban használt felügyeletlen tanulásban, ahol egy köztes nyelvre oda-vissza fordítva hasonló eredményt kell hogy kapjunk. Ezt a saját problémánkban úgy értelmezzünk, hogy 2D-s koordinátákhoz jósolunk 3D-seket majd ellenőrzésként visszafordítjuk őket és az eredetihez közeli eredményt várunk.

A képen sor, diagram, vállfa látható

Automatikusan generált leírás

. ábra Ideiglenes konvolúció

## Ideiglenes konvolúciós hálók

A temporális konvolúciós hálókról általánosságban elmondható hogy 1D konvolúciós hálót használnak a bemenet hosszát felfoghatjuk az idő dimenziója ként vagy képkockánkénti információként. Ahhoz hogy a kimenet vagy több réteg esetén a következő réteg egyik elemét megkapjuk vesszük az előző szint elemének skaláris szorzatát ahol a kernel mérete majd a következő elemhez a kernelt egyel arrébb csúsztatjuk. Több bemeneti csatorna esetén a kernelünk \* C ahol C a bemeneti csatornák száma. Így végső soron 2D-s konvolúciót hajtunk végre de továbbra is csak 1 dimenzió mentén haladunk és persze több csatornából több csatornába képzés esetén minden kernelnek külön súlyokat kap.

### Alkalmi konvolúció

Megkülönböztetünk még ezen kívül „alkalmi” (casual) ideiglenes konvolúciót ahol minden -edik elemhez a kimenetben csak a bemenet {} elemeit használhatjuk. Ez végső soron azt jelenti, hogy egy képen a koordináták meghatározásához nem vehetjük számba a jövőbeli képeket ez felvett videó esetén egy felesleges korlátozás de élő videófelvételnél szükséges. Ezt az alkalmi ideiglenes konvolúciót könnyen betarthatjuk, ha nulla-párnázással látjuk el a bemenetünket.

### „Táguló” ideiglenes konvolució

Az egyik jó tulajdonsága a szekvenciákból jósoló modelleknek hogy akár az összes bemeneti adat hozzájárul 1 kimeneti adat értékéhez (vagy alkalmi konvolúció esetén az összes kisebb indexű bemeneti adat). Ezt nevezhetjük teljes történeti fedettségnek. Ahogy ezelőtt láttuk egy konvoluciós réteg ráhatása a következőre vagy a kimenetre a kernel méretétől függ, ha van rétegünk es a kernel mérete akkor a befogadó mező

ha meg akarjuk nézni hogy hány réteg szükséges a teljes lefedettséghez

ez nagyobb bemenet eseten könnyen nagyon mély hálóhoz vezetne rengeteg paraméterrel és hosszú tanulási idővel. Ehelyett bevezethetjük a „tágulást” ami exponenciálisán nagy befogadó mezőt eredményez. A tágított konvolúció a konvolúció egy sajátos formája ritka szerkezettel, amelynek magpontjai egymástól távol helyezkednek el egyenletesen és közöttük nullákkal töltve. Például, egy diszkrét szűrő h = [ 1 2 3 ] (ahol 2 a középpont) d = 2 diletációs faktorral [ 1 0 2 0 3 ] lesz, és [ 1 0 0 2 0 0 3 ], ahol d = 3. Ezt a diletációt rétegenként exponenciálisan növelve kapjuk hogy a látómező

ahol a dilatációs bázis (általában 2 vagy 3)

A képen szöveg, képernyőkép, Betűtípus, sor látható

Automatikusan generált leírás

. ábra Táguló ideiglenes konvolúció és táguló aktuális konvolúció(b = 3)

A képen szöveg, Betűtípus látható

Automatikusan generált leírás

. ábra Bemenetünk 234, képkocka 34 ízületi koordináta (2x17) B = 4 konvoluciós block alkotja a modell törzsét zölddel a bemenetek ahol 2J = 2x17 ízületi koordináta 3d1 = 3-as kernel méret 1 dilatáció és 1024 kimeneti csatorna

## Modell architektúrája

Modellünk 10.ábra egy teljesen konvolúciós architektúra maradék kapcsolatokkal, amelyek 2D-s pózok sorozatát veszik fel bemenetként, és időbeli konvolúciókon keresztül átalakítja őket. A konvolúciós modellek lehetővé teszik a párhuzamosítást mind a köteg és idődimenzión át, míg az RNN-ek időben nem párhuzamosíthatók. A konvolúciós modellekben a gradiens útja kimenet és a bemenet között fix hosszúságú, függetlenül a sorozat hosszától, ami mérsékli az eltűnés és robbanó gradiens problémákat, amelyek befolyásolják az RNN-ket. A konvolúciós architektúra az időbeli receptív mező precíz vezérlését is kínálja, ami előnyös az időbeli modellezés szempontjából a 3D pózbecslés feladatához. Ráadásul, táguló konvolúciókat alkalmazva a hosszú távú függőségek modellezésére, hatékonyak is maradunk. A dilatációs konvolúciójú architektúrák sikeresek voltak a hanggenerálásban, a szemantikai szegmentálásban és a gépi fordításban is. A bemeneti réteg megkapja az összefűzőt (x, y) koordinátáit a izületnek, és időbeli konvolúciót alkalmaz és C kernelméretű kimeneti csatornákkal ( a kernel mérete C a bemeneti csatornák száma. Ezt követik a B ResNet-stílusú blokkok, amelyeket egy átugrási kapcsolat vesz körül. Minden blokk először egy 1D konvolúciót hajt végre kernelmérettel és dilatációs tényezővel, majd ezt követi egy konvolúció 1-es kernelmérettel.. A konvolúciókat (a legutolsó réteg kivételével) követi egy batchnormalizáció , majd egy ReLu aktiváció és végül „lemorzsolódás”. Minden blokk exponenciálisan növeli a befogadó mezőt k-szorzóval, miközben a paraméterek száma csak lineárisan nő. A szűrő hiperparaméterei, és D, úgy vannak beállítva, hogy bármely kimenet befogadómezője olyan fát alkot, amely lefedi az összes bemeneti képet. Végül az utolsó réteg előrejelzést ad a 3D-s pózokról minden képkockára a bemeneti szekvenciában felhasználva múltbeli és jövőbeli adatok egyaránt.

A tanulmányban kísérleteznek élő videók feldolgozásával ,amihez aktuális konvoluciót használnak itt a konvolúciós képmodellek általában nulla kitöltést alkalmaznak, hogy annyi kimenetet kapjanak, ahány bemenetet. A korai kísérletek azonban jobb eredményeket mutattak, ha ezek helyett a szekvencia kezdeti vagy végső adataival töltöttük fel a rétegeket

## Félig-felügyelt megközelítés

Megpróbálunk fel nem címkézett videókból is információt kinyerni annak érdekében ,hogy javítsuk a modellünk veszteságfüggvényén. Megoldunk egy automatikus kódolási problémát címkézetlen adatokkal: a kódoló (pozíció becslés) 3D-s pózbecslést végez a 2D-s ízületi koordinátákon és egy dekódoló visszavetítí ezeket a 3D-s pózokat 2D-s koordinátákra. A tanítás során pedig „büntetjük” ha a dekódoló koordinátái nagyon eltérnek az eredetitől. Az alábbi képen látható, ahogy a 2 modell együtt működik és a félig-felügyelt modell egyfajta szabályozóként funkcionál. A két modellt együttesen optimalizáljuk, ahol a kötegünk(batch) első felét a felcímkézett adatok teszik ki míg a másodikat a felcímkézetlen adatok. A felcímkézett adatokhoz az „Igaz” 3D-s koordinátákat használjuk célként a veszteségfüggvényben míg a felcímkézetlen adatoknál az autoencoder veszteségét használjuk.

A képen szöveg, diagram, képernyőkép, Tervrajz látható

Automatikusan generált leírás

. ábra Félig-felügyelt tanítás

### Leképzés

A perspektív vetítés miatt a 2D-s póz a képernyőn függ a pályától (pl: az emberi alap ízület globális pozíciójától, ami nálunk a két csípő ízület felezőpontja, ami a csípőcsont közepe akar lenni) és a 3D-s pozíciótól (az összes ízület pozíciója relatíve az alap ízülethez).A globális pozíció nélkül az alany mindig a kép közepére lenne visszavetítve egy fix méretben. Ezért szabályozzuk a 3D-s pályáját az embernek, hogy helyesen legyen visszavetítve 2D-be.Erre optimalizálunk egy második hálózatot ami szabályozza a globális pályát a kamera térben. Az utóbbit hozzáadjuk a pozícióhoz mielőtt visszavetítjük. A két architektúra megegyezik ,de nem osztoznak a súlyokon mert ez negatívan hat a tanulás során. Mivel egyre nehezebb a szabályozni a pályát ha egy ember messzebb van a kamerától, ezért optimalizáljuk a súlyozott per ízület pozíció hiba átlagot (WMPJPE) veszteségfüggvényt

vagyis minden minta súlyozva van az igazi mélység inverzével a kameratérben. Szabályozni egy precíz pályát a messzi embereknek relatíve felesleges hísz az ízületek amúgy is egy kis pont köré tömörülnek.

### Csont hossz L2 veszteség

Szeretnénk ösztönözni a hihető 3D-s koordináták prédiktálását ,ahelyett hogy csak visszaadjuk a kimenetet. Ehhez effektívnek találták a bevezetését egy gyenge szabályozásnak, ami nagyjából próbálja az átlag csont-hosszát az alanynak a nemcímkézett adaton és a címkézett adaton egyenlőnek tartani. Ez akkor hoz előnyöket ha a tanítás során kevés a címkézett adat.

### Kamera Paraméterek

### A tanitáshoz és értékeléshez szükségesek még a kamera belső paraméterei (ami a legtöbb ma használatos kameránál adott) mivel 3D-s „Igaz” pózokat a kamera térben rotáljuk és transzformáljuk ezek alapján. Kivéve a félig-felügyelt modellben ahol a globális pályát használjuk. Itt egy egyszerű vetítést használunk (lineáris fokális távolságot és elsődleges pontot feltételezve) úgy mint nem lineáris lencse disztorció együtthatót(érintő, vagy sugár irányú). Úgy találták hogy az adathalmazban használt kamera lencse disztorciója elhanyagolható

# Alkalmazás

Végül bemutatom, hogy az esetemben ezt a technológiát hogyan alkalmaztam rövidpályás gyorskorcsolya videók elemzésére. A modellben szerencsére van implementált funkció mind a kapott koordináták fájlba mentésére és ezek vizualizálására is. A koordinátákat egy .json fájlba mentve könnyen lehet olvasni és képkockánként értékelni az eredményeket. A sportágban a gyors sebesség miatt kulcsfontosságot kap a videóelemzés, és ebben a különböző ízületi szögek mérése. Ezeket pedig a térbeli koordinátákkal egyszerűen ki lehet számolni:

A következő képeken látható, hogy lehet pozíciók mélységét (térdszöget) kanyarban a centrifugális erő elleni bedőlés szögét(dőlés szöget) mérni. Utóbbinál kihasználjuk, hogy a nagy sebesség és gyorsulás miatt fellépő ellenerők stabilizálására a kanyar közepe környékén lerakjuk a kezünket a jégre. Ezzel a boka és a kéz vonala nagyjából azonosítja a képen a vízszintet és erre tudunk mérni egy dőlésszöget. Persze amelyik képen nincs jégen a kéz ott ez az érték figyelmen kívül hagyandó. Mérhető még egyes pozíciókban a lökés teljessége is (eléri-e a láb nyújtás a 180º-ot).

A képen szöveg, képernyőkép látható

Automatikusan generált leírás

12. ábra Jobb térdszög 136º dőlés szög 39º

A képen szöveg, képernyőkép, jégkorcsolyázás látható

Automatikusan generált leírás

13. ábra Térdszög 120º dőlés szög 34º

A képen szöveg, sportfelszerelés, síelés, képernyőkép látható

Automatikusan generált leírás

14. ábra Térdszög 175º

A képen szöveg, sportfelszerelés, síelés, képernyőkép látható

Automatikusan generált leírás

. ábra Lábak összekeverése

Természetesen a gyorskorcsolya rendkívül speciális mozgású sport és lenyűgöző, hogy a modell relatíve mennyire jól kiigazodott ezen a mozgáskultúrán, de néha összekevert pár lábat és természetesen a 3D-s koordináták és a belőlük számolt szögek sem voltak mindig közel a valósághoz. ennek ellenére elképzelhető, hogy a jövőben nem kellenek majd ruhába vart szenzorok és egyéb trükkök arra ,hogy hatékonyan elemezhessük ezen extrém mozgású sportokat is csupán videófelvételekből.

# Irodalomjegyzék

|  |  |
| --- | --- |
| [1] | A. Géron, Hands-On Machine Learning with Scikit-Learn & TensorFlow, O'Reilly Media, Inc, 2019. |
| [2] | I. Godfellow, Y. Bengio és A. Courville, Deep Learning, http://www.deeplearningbook.org: MIT Press, 2016. |
| [3] | G. Gkioxari, P. Dollár and R. Girshick, “Mask R-CNN,” 2017. [Online]. Available: https://openaccess.thecvf.com/content\_ICCV\_2017/papers/He\_Mask\_R-CNN\_ICCV\_2017\_paper.pdf. |
| [4] | R. Shaoqing, H. Kaiming, R. Girshick és J. Sun, „Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” 2016. [Online]. Available: https://arxiv.org/pdf/1506.01497.pdf. |