BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Electrical Engineering
and Communication

MASTER'S THESIS

Brno, 2020 Bc. Lukáš Dobiš

BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF BIOMEDICAL ENGINEERING
ÚSTAV BIOMEDICÍNSKÉHO INŽENÝRSTVÍ

DETECTION OF PERSONS AND EVALUATION OF GENDER AND AGE IN IMAGE DATA
DETEKCE OSOB A HODNOCENÍ JEJICH POHLAVÍ A VĚKU V OBRAZOVÝCH DATECH

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR — **Bc. Lukáš Dobiš**
AUTOR PRÁCE

SUPERVISOR — **doc. Ing. Radim Kolář, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2020

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

Master's Thesis

Master's study program **Biomedical Engineering and Bioinformatics**

Department of Biomedical Engineering

Student: Bc. Lukáš Dobiš ***ID:*** 186654

Year of study: 2 ***Academic year:*** 2019/20

TITLE OF THESIS:

Detection of persons and evaluation of gender and age in image data

INSTRUCTION:

1) Make a review on methods for face detection, gender classification and age assessment in image data with the focus on deep learning approaches. 2) Find suitable image sets on the Internet. 3) Choose and implement one of the face detection approaches and evaluate its precision. 4) Add the age and gender assessment to the algorithm and optimize it in order to achieve the best results. 5) Make a proper quantification of the achieved results, and discuss the results.

RECOMMENDED LITERATURE:

[1] LeCun, Y. et al., “Deep learning”, Nature, 521 (7553), pp. 436–444, 2015.

[2] Levi, G. et al., “Age and Gender Classification Using Convolutional Neural Networks”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp. 34-42, 2015.

Date of project specification: 3.2.2020 ***Deadline for submission:*** 29.5.2020

Supervisor: doc. Ing. Radim Kolář, Ph.D.

prof. Ing. Ivo Provazník, Ph.D.
Chair of study program board

ABSTRACT

This master thesis describes an approach for automated human recognition by using convolutional neural networks (CNN) to perform facial analysis of persons face in image data. The predicted biometric indicators are following: age, gender, facial landmarks and facial expression. CNN architectures with pretrained weights for each task are described. Age estimation CNN has new weights trained and freezed, then has added new LSTM layers into its architecture. New LSTM layers are trained and tested on newly created video data set. Test results indicate improved age prediction accuracy. Solution for human recognition inference with single image and time series variants, in form of script with interconnected CNNs is explained, and its inference speed performance supports further proposed expansion plans for live video inference.

KEYWORDS

deep learning,computer vision, convolutional neural networks, long short-term memory, face detection, age estimation, gender classification, emotion classification, Pytorch

ABSTRAKT

Táto diplomová práca sa venuje automatickému rozpoznávaniu ludí v obrazových dátach s využitím konvolučných neurónových sieti na určenie polohy tváre a následnej analýze získaných dát. Výsledkom analýzy tváre je určenie pohlavia, emócie a veku osoby. Práca obsahuje popis použitých architektúr konvolučných sietí pre každú podúlohu. Sieť na odhad veku má natrénované nové váhy, ktoré sú vzápätí zmrazené a majú do svojej architektúry vložené LSTM vrstvy. Tieto vrstvy sú samostatne dotrénované a testované na novom datasete vytvorenom pre tento účel. Výsledky testov ukazujú zlepšenie predikcie veku. Riešenie pre rýchlu, robustnú a modulárnu detekciu tváre a ďalších ludských rysov z jedného obrazu alebo videa je prezentované ako kombinácia prepojených konvolučných sietí. Tieto sú implementované v podobe skriptu a následne vysvetlené. Ich rýchlosť je dostatočná pre ďalšie dodatočné analýzy tváre na živých obrazových dátach.

KĽÚČOVÉ SLOVÁ

hlboké učenie, počítačové videnie, konvolučné neurónové sieťe, LSTM, detekcia tváre, odhad veku, klasifikácia pohlavia, klasifikácia emócií, Pytorch

DOBIŠ, Lukáš. *Detection of persons and evaluation of gender and age in image data*. Brno, Rok, 100 p. Master's Thesis. Brno University of Technology, Faculty of Electrical Engineering and Communication, Department of Biomedical Engineering. Advised by doc. Ing. Radim Kollář, Ph.D.

ROZŠÍRENÝ ABSTRAKT

Táto diplomová práca sa venuje téme automatického rozpoznávania ľudí v obrazových dátach za použitia konvolučných neurónových sietí. Výsledkom práce je úspešná detekcia ľudskej tváre a jej následná analýza určujúca pohlavie, emóciu a vek osoby. Na každú podúlohu tohto procesu je použitá jedna konvolučná sieť. Celkové riešenie rozpoznávania je kombináciou prepojených konvolučných sieti. Tento prístup umožňuje využiť architektúry, ktoré sú state-of-the-art pre jednotlivé podúlohy v jednom celku tak, že môžu byť ľahko vymenené a modifikované bez zásahu do ostatných sietí. To samozrejme platí iba vtedy, ak sa nemení sieť, na ktorej výstup sú ostatné siete naučené. To znamená, že ak sa zaručí vstup - výstup kompatibilita medzi sieťami, siete budú v totožnej forme ako využívajú súčasne najúspešnejšie architektúry, tak výsledne riešenie je rýchle, presné, modulárne a schopné využívať váhy z najúspešnejších sietí súčasnosti. Každá sieť riešiaca jednu podúlohu má výhodu v možnosti využitia benchmark datasetov na vyhodnotenie ich úspešnosti. Práca pokrýva základy teórie neurónových sietí, od umelého neurónu k neurónovej sieti, spolu s menším úvodom k rekurentným sieťam. Následne sa venuje teórií konvolučných neurónových sietí a základným architektúram, ktoré túto teóriu implementovali, resp. kde bola teoria po prvýkrát aplikovaná. Metodika detekcie objektov za použitia konvolučných sieti je podrobne popísaná a sú predstavené hlavné používané architektúry na detekciu objektov. Jedna z nich s názvom RetinaNet je porovnaná a rozpísaná do hĺbky, vrátane jej chyby, architektúry, trénovania a inferencie, za účelom ukážky aplikácie metodiky detekcie objektov. Na tento rozbor nadväzuje od RetinaNet odvodená architektúra RetinaFace, ktorá je špecializovaná na detekciu tváre a je zodpovedná za detekciu tváre v tejto práci. V skrátenej podobe je spravený prieskum metód na klasifikáciu pohlavia a emócií, na základe obrazu tváre osoby. Pre obe úlohy je vybrané riešenie pomocou konvolučnej klasifikačnej neurónovej siete, založenej na modernej architektúre s názvom Xception. Autori tejto siete využili túto architektúru a zmenšili ju smerom k najlepšiemu pomeru presnosti na počet parametrov architektúry. Výsledkom je veľmi malý model (<1000kB) siete s vyššou rýchlosťou a minimálnou stratou presnosti. Poslednou riešenou podúlohou rozpoznávania je odhad veku. Táto úloha je náročná, s chybou okolo 4 až 12 rokov a ponúka teda priestor na zlepšenie. Metódy odhadu veku majú dva hlavné prístupy. Prvý sa pozerá na vek ako kladné racionálne číslo. Druhý prístup považuje vek za ordinálnu premennú. Tieto metódy majú vyššiu úspešnosť, preto sú predstavené v skrátenom zhrnutí. Jedna z nich je vybraná na odhad veku v tejto práci. Vybraná metóda je založená na konvolučnom neurónovom modeli na riešenie ordinálnych problémov a má názov Konzistentné Rádové Logity. Logity predstavujú v metódach hlbokého učenia nenormalizovaný vektor predikcií klasifikačného modelu, ktorý je prevedený na vektor normalizovaných pravdepodobností tried, v prípade vi-

acerých klasifikačných tried. V použitej metóde je sieť naučená vytvoriť z obrázkov tváre logit, so zoradenými triedami veku, ktorý je následne normalizovaný poslednou vrstvou, do podoby vektoru binárnych klasifikačných úloh. V tejto podobe pozitívna binárna klasifikácia u vekovej triedy neznamená, že osoba má vek danej triedy, ale že osoba má vek vyšší než daná trieda. To znamená, ak je pozitívna klasifikácia pre triedu veku 8 rokov, tak sieť musí mať pozitívne klasifikácie aj pre triedy 1 až 7 rokov. Týmto sa sieť naučí takzvanej triednej konzistencií a nenastane pozitívna klasifikácia pre 2 vzdialené triedy veku napríklad 12 a 30 rokov. Metóda ďalej ponúka váhovanie dôležitosti tried veku v poslednej vrstve, podľa zastúpenia jednotlivých vekov v trénovacom datasete. To znamená sieť sa viac učí z tried s malým počtom obrázkov ako napríklad hraničné vekové skupiny. Teoretická časť práce je ukončená návrhom na vylepšenie presnosti tejto architektúry tak, že sa pridá element rekurzívnych sietí zvaný LSTM medzi vytvorený logit a klasifikačnú vrstvu. Sieť je zmenená na rekurzívnu a vrstva LSTM buniek spracováva časovú sekvenciu logitov vytvorených z tvári do jedného logitu, ktorý vstupuje do klasifikačnej vrstvy. Pre potreby učenia týchto LSTM vrstiev sú zhrnuté požiadavky na trénovací a testovací dataset, ako posledné sú navrhnuté testované spôsoby nastavenia na dotrénovanie týchto vrstiev. Práca prechádza k praktickej časti tým že obsahuje rozbor datasetov pre každú podúlohu od detekcie tváre cez odhad veku až po klasifikáciu emócií. Pohlavie býva často značené ako súčasť anotácie v spomenutých datasetoch. Pred implementáciou sú popísané populárne prostredia pre návrh a učenie neurónových sieti. V implementácií sú hardvérové a softvérové požiadavky použitých metód. Potom je predstavené riešenie automatického rozpoznávania ľudí, za použitia modelov sietí s predtrénovanými váhami a možnosťou výberu váh pre sieť odhadu veku podla ich trénovacieho datasetu. Detekcia tváre využívajúca RetinaFace architektúry má možnosť výberu medzi modelom s architektúrou pre extrakciu príznakov ResNet-50 alebo MobileNet-0.25. Implementované riešenie začína inicializáciou modelov sietí a načítaním ich váh, ktoré môžu byť v niektorých prípadoch vybrané vstupom užívatela. Na propagáciu vstupov cez sieť si užívatel môže zvoliť medzi grafickou kartou alebo procesorom jeho zariadenia. Z priečinku, do ktorého je zvolená cesta, sú načítané obrazy, ktoré sú normalizované a vložené na vstup siete RetinaFace. Tá predikuje polohu tváre v podobe predikcie parametrov obdĺžníka lemujúceho obvod tváre. Ďalšími predikovanými parametrami sú pozície očí, nosu a kútikov úst. Silne prekrývajúce obdĺžniky sú zjednotené. Obdĺžniky majú parameter pravdepodobnosti, že obsahujú tvár. V prípade pravdepodobnosti nižšej než 97% je obdlžnik vyradený. Výstupom detekcie tváre sú parametre nevyradených obdĺžnikov, ktoré značia tvár zadetekovanej osoby. Pre každý obdĺžnik, respektíve tvár v obraze je spravená klasifikácia podúloh zvyšnými neurónovými sieťami. Siete na klasifikáciu pohlavia a emócií majú na vstupe o 66% vyšší a 50% širší obdĺžnik, pre zvýšenie

presnosti. U týchto sietí je obraz ešte normalizovaný a prevedený do šedotónovej podoby. Obe siete majú na výstupe vektor pravdepodobností, kde najvyššia hodnota značí predikovanú triedu. Sieť na odhad veku má tiež predspracovaný obraz tváre ako vstup a výstupom je vektor pravdepodobností značiaci pozície výstupov ordinálnej binárnej klasifikácie vekových tried. Po prahovaní 0.5 prahom obsahuje vektor jednotky a nuly. Jednoduchou sumou jednotiek je získaný výstupný predikovaný vek. K tomuto veku sa v prípade váh, ktoré boli trénované na datasete s vekovým rozsahom nezačínajúcim od 0, pripočítava spodná hranica tohto datasetu k predikovanej hodnote. Predikcie podúloh sú vykreslené s obdĺžnikom pri detekovanej tvári v obraze. Výstupom riešenia je tento obraz uložený spolu s metadátami predikcií do priečinku vybraného vstupom užívatela. Ďalej nasleduje popis tvorby vekového datasetu video sekvencií obrázkov. Z youtube kanálu SoulPancake sú extrahované zo 7 videí scény s jednotlivými osobami, ktorých vek bol značený v ľavom dolnom rohu obrazovky. Zo scén 6 videí, je vytvorený trénovací dataset zložený z sekvencií 20 obrázkov zoradených podľa času. Testovací dataset má scény iba z 1 videa, ktoré je staršie ako ostatné videá. Skladá sa zo sekvencií, vytvorených iba z prvých 20 obrázkov scény, takže väčšina osôb má väzbu iba na jednu sekvenciu, vytvárajúc objektívny dataset pre porovnávanie predikcie veku. Posledná implementovaná metóda je vylepšená sieť na odhad veku. Má popísané trénovanie nových váh ako základ pre testované LSTM vylepšenie. Tieto váhy sú zmrazené a sú skúšané vklady 1 až 3 vrstiev do architektúry na doučenie na trénovacom data sete video sekvencií. Na najlepšom počte vrstiev sú potom testované možnosti odmrazovania vrstiev architektúry iných než LSTM. Najúspešnejšie váhy vylepšenej architektúry sú potom vybrané na porovnanie s váhami pôvodných autorov siete na odhad veku a natrénovanými váhami pred vložením LSTM vrstiev. Porovnanie je spravené na testovacom datasete video sekvencií. Výsledky obsahujú priemernú rýchlosť inference u jednotlivých sieti a riešenia rozpoznávania vlastností vytvoreného datasetu videosekvencií a porovnanie presnosti medzi sieťami na odhad veku. Ukázalo sa, že časovo najnáročnejšia je úloha detekcie tváre. ResNet má špatné hodnoty pri inferencií na procesory, MobileNet je ledva použiteľný pre živú inferenciu na procesory. Pri použití na grafickej karte sa rýchlosť zvýši ,ale stále je značne závislá od počtu osôb v obraze. Vytvorený dataset video sekvencií má dobrú etnickú diverzitu osôb, iba mierne viac osôb v mladých vekových než v starších vekových skupinách. Má značne viac žien než mužov a najpočetnejšiu skupinu kaukazského etnika. Výsledky testovaného vylepšenia mali ako najlepšie trénovacie nastavenie, pridanie 2 LSTM vrstiev a dotrénovania čisto LSTM vrstiev. Z porovnania sietí na odhad veku vyplynulo, že vylepšená sieť dosahuje presnejšie výsledky ako nevylepšená sieť. Riešenie automatického rozpoznávania má druhý variant s vylepšenou sieťou na odhad veku, ktorý ale musí byť použitý na obrazy zoradené v čase.

DECLARATION

I declare that I have written the Master's Thesis titled "Detection of persons and evaluation of gender and age in image data" independently, under the guidance of the advisor and using exclusively the technical references and other sources of information cited in the thesis and listed in the comprehensive bibliography at the end of the thesis.

As the author I furthermore declare that, with respect to the creation of this Master's Thesis, I have not infringed any copyright or violated anyone's personal and/or ownership rights. In this context, I am fully aware of the consequences of breaking Regulation § 11 of the Copyright Act No. 121/2000 Coll. of the Czech Republic, as amended, and of any breach of rights related to intellectual property or introduced within amendments to relevant Acts such as the Intellectual Property Act or the Criminal Code, Act No. 40/2009 Coll., Section 2, Head VI, Part 4.

Brno ....

author's signature

ACKNOWLEDGEMENT

Rád by som sa poďakoval vedúcemu diplomovej práce pánovi doc. Ing. Radim Kollář Ph.D. za odborné a profesionálne vedenie a hlavne trpezlivosť. Ďalej sa chcem poďakovať všetkým osobám vrátane rodiny a známych, ktorý mi vytvorili zázemie potrebné pre vytvorenie tejto práce. A ako dúfam že budúci informatik by som tiež chcel poďakovať všetkým anonymným prispievatelom na fórách založených na riešenie programátorských problémov.

Contents

List of Figures

List of Tables

Introduction

Automated human recognition as a technology is used widely in human-computer interaction, the security and surveillance industry, demographic research, commercial development, mobile application, service robotics and video games [102]. Use of computer vision for human recognition, has greatly increased with introduction of deep learning methods based on CNN.

State-of-art neural networks often focus only on one task, so combining multiple of them using their strengths, leads to composition of networks each responsible for different task. This way used networks can have their architecture or weights changed at will. So when better architecture for one task is discovered it can be used as replacement without need to retrain whole composition of networks. With this approach it is also easier to train, test and evaluate networks, since independent tasks have often their own benchmark data sets.

This paper applies multiple pretrained convolutional neural networks to detect face, its landmarks and to further predict person age, gender and facial expression. These are important descriptors in automated human recognition. Since each human has unique appearance and leads different lifestyle, these descriptors precise prediction creates a challenge for current algorithms. Aim of this work is to create composition of CNNs oriented on human recognition, which will be capable of real-time inference in video, with improvement in prediction by utilizing temporal information.

Age estimation being hardest task is main focus. So in addition to pretrained weights has used age prediction network new weights trained and freezed. Then is modified with additional LSTM layers which are independently trained and tested on new data set created for this purpose. Results show that adding LSTM layers has improved age prediction performance.

Human recognition is therefore solved with distinct CNNs interconnected to create automated human face detection and subsequent age, gender and emotion classification from images. By using several lightweight precise CNNs, the proposed solution is fast, accurate and highly modular for further improvements.

This thesis is divided into 9 chapters. First chapter lays out basics of neural networks, from one neuron to network, gradient learning and problems these networks face. Second chapter provides basic review of convolutional neural network architecture, and provides examples of such networks. Third chapter introduces object detection as a problem of deep learning its basic fundamentals and approaches. Second half of this chapter is dedicated to RetinaNet architecture and its facial detector variant RetinaFace. Fourth chapter opens up age estimation as ordinal regression problem and offers solution in form of Consistent Rank Logits algorithm and follows

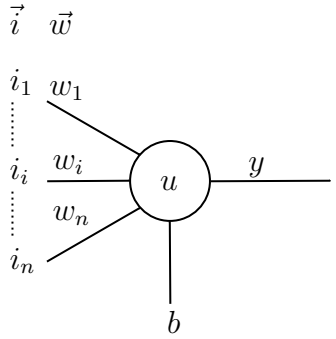
up with proposed enhancement based on LSTM elements. Fifth chapter covers gender and emotion classification tasks with review of methods and description of one such method, that is used to solve both tasks. Sixth chapter is a review of tasks data sets and neural network frameworks. Seventh chapter has implementation of human recognition solution, new video data set creation process and enhancement of age prediction network, which uses mentioned video data set for training and testing. Eight chapter consist of results from evaluations and output examples of human recognition solution, examples and internal structure of new video data set, and lastly enhanced age prediction evaluation. Ninth chapter discusses results in detail and offers possible improvements to amend solution shortcomings. Thesis concludes with a thorough review of all accomplished actions and information deducible from them.

1 Neural networks

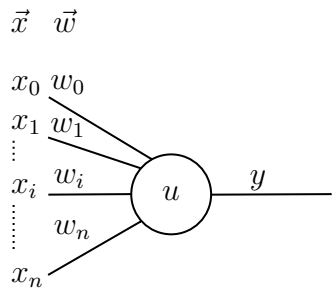
To make computer output human-like, its inner algorithm workings must approximate human information processing. Basic processing element of human brain is neuron. In human biology the visual information processing works sequentially, by decomposing visual input followed with connecting small edge lines into feature patterns which are then used to recognise objects in the input. Recognised objects give insight about information contained within the visual input. Each part of this process is done by unique inline neural networks (NN), sending processed information in the form of electrical impulse. To achieve human level recognition one must design and implement such a set of layered networks.

1.1 Artificial neuron

Artificial neuron (AN) is an approximation of its biological counterpart. Biological neuron can be presented as function weighting multiple inputs, whose sum if reaches firing potential, sends a single electrical impulse as output. This can be approximated by mathematical model (1.1a). Where $\vec{i}$ is vector of input numbers, weighted by weight vector $\vec{w}$ in base function $f(\vec{i}, \vec{w}, b)$. This function outputs equivalent of firing potential u. Neural output y is representation of biological electrical impulse. Mirroring the biological neuron firing potential, model has a threshold potential otherwise known as bias b. For easier mathematical description, computation threshold potential has been embedded into input and weight vectors (1.1b). This is done by making b first value of weight vector and adding first value to input vector equal to one. Newly created vectors $\vec{x}$, $\vec{w}$ of N (n = 0,1,...,N) length are called augmented vectors [89][52].



(a) Artificial neuron model



(b) Augmented artificial neuron model

1.1.1 Base function

There are two main base function types. First is called Linear base function (LBF - 1.1) which linearly sums a vector of weighted inputs into output. Second type is Radial base function (RBF - 1.2), its output is distance between input and weight vector. Making a RBF neuron weight vector a multidimensional hyperplane [52].

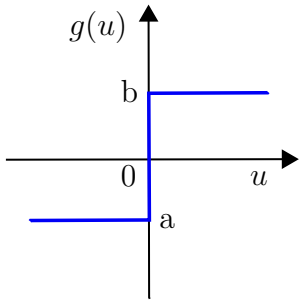
$$u = \sum_{k=0}^{N} x_k \cdot w_k \qquad (1.1) \qquad\qquad u = \sum_{k=1}^{N} \sqrt{(x_k - w_k)^2} \qquad (1.2)$$

Output of base function u is transformed by activation function $g(u)$ into single neuron output y. Linking y as an input to other neurons creates NN [89][21].

1.1.2 Activation function

Activation function (AF) has many various types, depending on nature of problem NN is solving. Because of this, this thesis wont cover AF for RBF neurons. Most common AF for LBF are step discontinuity AF with binary or bipolar outputs (1.3, b = 1 , a ∈ <-1,0>). If continuous output is needed, sigmoid function (1.4) or hyperbolic tangent (1.5) can be used (1.3a, b = 1 , a ∈ <-1,0>) [52].

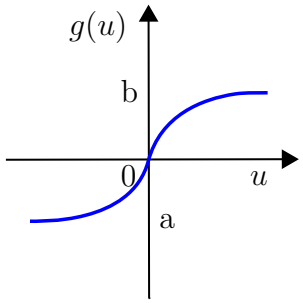
$$y = g(u) = \begin{cases} \text{a} & u < 0 \\ \text{b} & u > 0 \\ y_{old} & u = 0 \end{cases} \qquad (1.3)$$



(a) Jump activation function

$$y = \frac{1}{1 + e^{-u}} \qquad (1.4)$$

$$y = \tanh(u) \qquad (1.5)$$



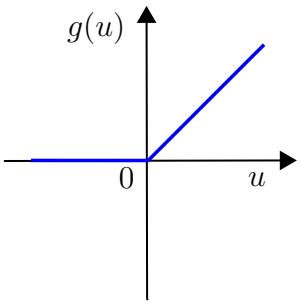
(a) Continuous activation function

For classification are often used Rectified Linear Unit (ReLU) and SoftMax functions (1.6, 1.7). Use of ReLU AF (1.4a) makes layered NN, fast at learning and since

it ignores negative values its optimal for work with positive image pixel values. SoftMax function takes potential u from final layer neurons and normalises them so their sum equals to one. This gives NN probability classification by making output y equal to probability of class each neuron represents [89].

$$y = \max(0, u) \tag{1.6}$$

$$y_j = \frac{e^{u_j}}{\sum_{k=1}^{M} e^{u_k}} \tag{1.7}$$



(a) Rectangular Linear Unit - ReLU

1.2 Learning

In machine learning (ML) there are two main approaches to learning a NN. First is gives NN a data and lets NN learn input data inner characteristics which NN uses to correctly react to new introduced data. This is called unsupervised learning (UL), because NN has to learn everything just from data alone. Second approach gives with data additionally desired output d. NN then has to learn how to transform input i to y corresponding to d. Due to d being a human labeled information to data, this approach is supervised learning (SL). Scale of this thesis is narrowed to only SL approach [21][52].

1.2.1 Training and test data

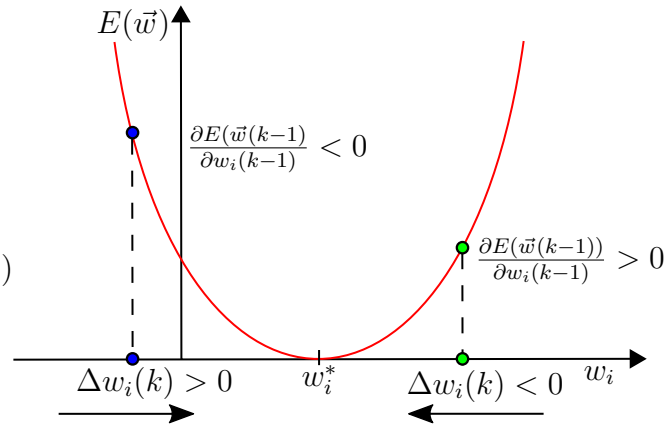
Training NN to learn from data, how to figure out problem, needs correctly labeled data. Training data set T for adaptive learning is collection of P (p = 1,2,...,P) predictors composed of associated pairs of $\vec{i}$ and $\vec{d}$. During one learning step, is one p-th set of pair vectors entered as input to NN. Training data set can be used to train network more than once. One phase of training using whole data set is called epoch. After training ends NN accuracy to problem solving, is tested on separate data unused in learning. This data is called test data set [89].

1.2.2 Objective function

By measuring offsets of y from d, the objective function E, otherwise called error, cost or loss function can be established. Goal of NN learning is to have this func-

tion minimized in each neurons during learning phase. Original backpropagation algorithm uses as loss function half of summed squared error (1.8) [89][52].

$$E(\vec{x}_p, \vec{w}) = \frac{1}{2}\sum_{k=1}^{M}(y_k - d_{pk})^2 \quad (1.8)$$



(a) Objective function relation to Δw

1.2.3 Backpropagation

NN learning can be as concept epitomized into finding correct neuron weight values for optimal problem solution. That is achieved by having E in its global minimum. Finding a $\vec{w}^*$ associated to the location of this minimum is done by using inverted gradient of E (1.5a) to iteratively update $\vec{w}$. This mathematical apparatus is called Gradient descent. It is used in Backpropagation algorithm. Backpropagation in layered NN calculates loss function of one p set training vectors by computing (1.8) from last layer outputs o (1.9). Then by using partial derivation of E_p with respect to $w(k-1)$, to find out how much individual weight w_i last iteration contributes to E_p. And next $w(k)$ update is done by subtraction of this amounth from it (1.10). This (1.11) is example of one neuron equation derivation for one weight update. Element δ is partial derivation of E with respect to u, lastly μ represents learning coefficient. Reason why δ is singled out is that in differential learning it differs from neuron to neuron by their ∂E function and ∂AF [89][21][52].

$$E_p = \frac{1}{2}\sum_{j=1}^{M}(d_{pj} - o_{pj})^2 \quad (1.9) \qquad \Delta w = -\mu \nabla E_p \quad (1.10)$$

$$\Delta w_i = -\mu\frac{\partial E(\vec{w}(k-1))}{\partial w_i(k-1)} = -\mu\frac{\partial E(\vec{w}(k-1))}{\partial u}\frac{\partial u}{\partial w_i(k-1)} = \mu\delta x_i \quad (1.11)$$

This process is during learning used to update all neurons weights in all L (l = 1,2,...,L) layers of NN. For output layer L neurons is output ${}^L\delta$ equal to $\frac{\partial E(\vec{w}(k-1))}{\partial u}$ but for other layers ($l < L$) is ${}^l\delta$ unknown. To get ${}^l\delta$ multiple derivations are needed. Starting with partial derivation of E function with respect to j-th neuron output y

of l non output layer (1.12). In this equation ${}^{l+1}N$ is number of neurons in $l+1$ above layer, $\frac{\partial E}{\partial {}^{l+1}u_k}$ is ${}^{l+1}\delta$ of above layer. And since (1.14) output ${}^{l}y_j$ is input ${}^{l+1}x_i$ to above layer, element $\frac{\partial {}^{l+1}u_k}{\partial {}^{l}y_j}$ is then simple LBF derivation which for (1.1) is equal to j-th neuron weights. Trimmed down this means that non last layer j-th output y contribution to E function is sum of asociated w_i for i equal to j weighted by neurons each own δ. Solution (1.12) is for each neuron combined with his individual AF derivation $\frac{\partial {}^{l}y_j}{\partial {}^{l}u_j}$ (1.13) and ${}^{l}\delta_j$ is now calculable from above layer neurons δ and $\vec{w}$. Examples of AF differentiated equations is in table (1.1) [89][21][52].

$$\frac{\partial E}{\partial {}^{l}y_j} = \sum_{k=1}^{{}^{l+1}N} \frac{\partial E}{\partial {}^{l+1}u_k} \frac{\partial {}^{l+1}u_k}{\partial {}^{l}y_j} = \sum_{k=1}^{{}^{l+1}N} -{}^{l+1}\delta_k \, {}^{l+1}w_{kj} \tag{1.12}$$

$${}^{l}\delta_j = \frac{-\partial E}{\partial {}^{l}y_j} \frac{\partial {}^{l}y_j}{\partial {}^{l}u_j} = \sum_{k=1}^{{}^{l+1}N} {}^{l+1}\delta_k \, {}^{l+1}w_{kj} \frac{\partial {}^{l}y_j}{\partial {}^{l}u_j} \tag{1.13}$$

$${}^{l}y_j = {}^{l+1}x_i \tag{1.14}$$

Now that ${}^{l}\delta_j$ and ${}^{l}x_i$ are known (1.13)(1.14), any weight update for one p input set, can be computed as i-th weight of j-th neuron from l-th non last layer equates to (1.15). Needed equation elements are taken from one layer above so weight update propagates back through NN starting from last L-th layer. Giving this algorithm appropriate name Backpropagation [89][21].

$${}^{l}\Delta w_{ij} = \mu \, {}^{l}\delta_j \, {}^{l}x_i = \mu \sum_{k=1}^{{}^{l+1}N} ({}^{l+1}\delta_k \, {}^{l+1}w_{kj}) \frac{\partial {}^{l}y_j}{\partial {}^{l}u_j} {}^{l}x_i \tag{1.15}$$

Tab. 1.1: Table of differential equations for basic activation functions

Neuron Activation function	Original form	Differentiated form
Linear function	ku	k
Sigmoid	$\frac{1}{1+e^{-u}}$	$y(1-y)$
Hyperbolic tangent	$\frac{e^{2u}-1}{e^{2u}+1}$	$(1-y)^2$
Rectified linear unit	$\max(0,u)$	$\begin{cases} 0 & u \leq 0 \\ 1 & u > 0 \end{cases}$
SoftMax function	$\frac{e^{u_j}}{\sum_{k=1}^{M} e^{u_k}}$	$\begin{cases} y_j(1-y_j) & j = i \\ -y_j y_i & j \neq i \end{cases}$

1.3 Feedforward neural network

NN can be divided by many of its characteristics from their type of learning, purpose or computational hardware requirements. One such characteristic is the NN

topology. It can be used to classify NN by their inner inter-neural input-output connections. Feedforward neural network (1.6) is layered acyclic NN. Layered means that its composed of several layers of neurons with each neuron having input-output connection only with neurons from neighboring and own layer. Layers between input layer and output layer called hidden layers. Acyclic adds restriction on intra layer connections. Feedforward neural network sends augmented input $\vec{x}$ to all neurons of first layer whose each neuron sends his output y to all neurons of second layer etc. until last layer often with small number of neuron outputs final output $\vec{o}$ [89][21].

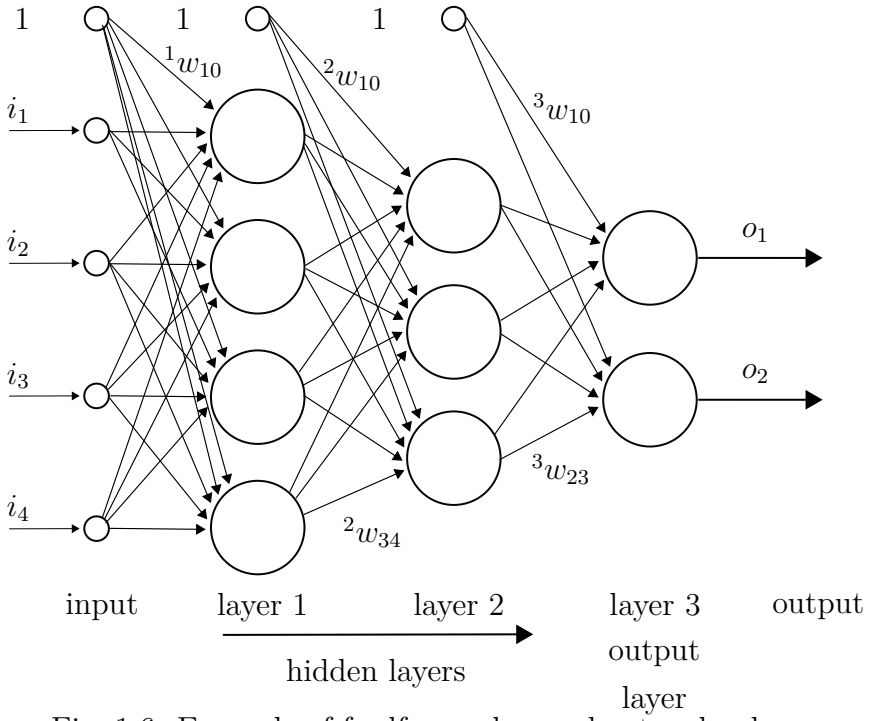


Fig. 1.6: Example of feedforward neural network scheme

1.4 Data augmentation

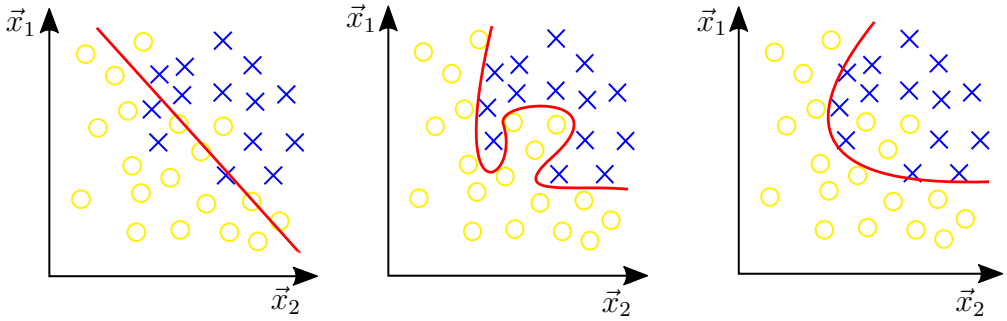
Data augmentation is a strategy that is used to increase diversity of training data, without actually collecting new data. It is important however that new data gotten by augmentation are in scope of instances normal to training data. Most basic augmentation is cropping,padding or horizontal flipping. There are also new approaches looking to solve low data generality with data augmentation aimed to fill data invariances [40].

1.5 Neural network problems

For NN to be apt at learning, required learning parameters need to be chosen appropriately. They define how fast and robustly NN learns during training. Badly chosen parameters cause NN to learn slowly, non generally or worse to learn nothing at all (1.8) [21][52].

1.5.1 Overfitting/Underfitting

If NN is not able, to learn correctly evaluate training data set it is denoted as underfitted on training data. But having global minimum of E for training data does not translate into trained NN correctly assessing data not related to training data set. Because NN accuracy is defined by, how well it learned from training data set to correctly evaulate test data set. This other problem case is called overfitting. It means that NN has not learned from training data its base features rather it learned its nuances or sequence. Shuffling training inputs and deliberate training end time, appertain to the most basic approaches to avoid overfitting. But in deep learning with large number of layers, shuffling is not enough, mainly in later epochs of training. But there are relatively more advanced approaches like dropout and batch normalisation layers. These layers will be explained in third chapter [21][52].



(a) Underfitted model (b) Overfitted model (c) Optimally fitted model

Fig. 1.7: UF, OF and optimally fitted $\vec{w}$, for binary classificator in 2D feature space

1.5.2 Slow learning start

High μ can lead to NN stepping over global E minimum, to circumvent this smaller μ can be used, but this leads to extremely time consuming and hardware dependent training runs. And increasing number of epochs elevates danger of overfitting on training data. To counter this, μ should be initialized with higher value in initial training phase and have adaptively lowered its value near E global minimum [89][48][52].

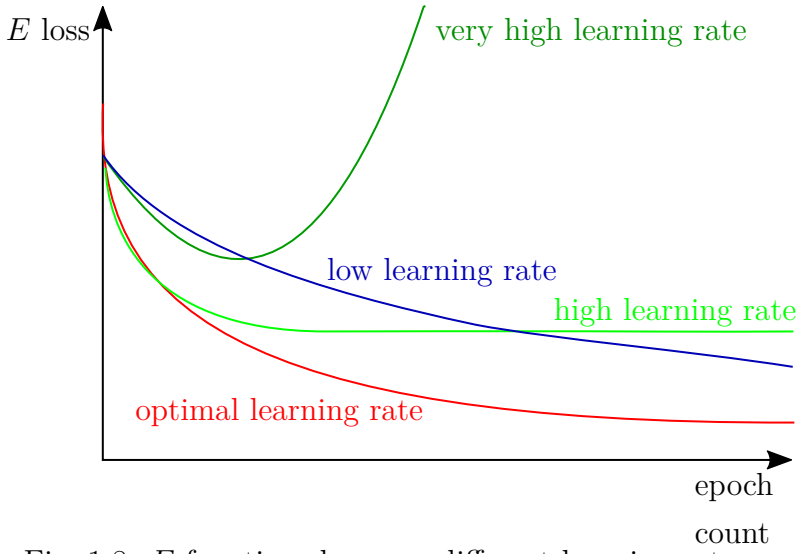


Fig. 1.8: E function shape on different learning rates

1.5.3 Vanishing gradient

$E(\vec{w})$ function in multidimensional feature space can take on various shapes. Often function values starts to resemble plateau shape. Then gradient based learning algorithms like Backpropagation, are not able to calculate weight update needed to leave this plateau. This lack of gradient is the foremost problem in deep learning. To solve this problem, skip connections or more specific case of residual block is utilized [89][48][52].

1.6 Recurrent networks

One of the more distinguished groups of neural networks categories are recurrent networks (RNN). They function mainly to extrapolate values from series of time dependent input. This is used with good results for tasks such as image captioning, language translation or handwriting recognition [60]. Older architectures had problem with vanishing gradient [89], but newer based on Long Short-Term Memory(LSTM,[42]) or Gated Recurrent Unit(GRU,[16]) sought to remedy that problem.

1.6.1 Long short-term memory

The most fundamental elements of recurrent networks which give network its memory are called memory cells. Memory cell for LSTM implementation can be seen on left side of figure 1.9. Where c represents cell state which is representation of cells long memory. Last output h acts as short memory. Operators σ and thg represent

in scheme sigmoid and hyperbolic tangens operations which transform input into <0,1> and <-1,1> range respectively.

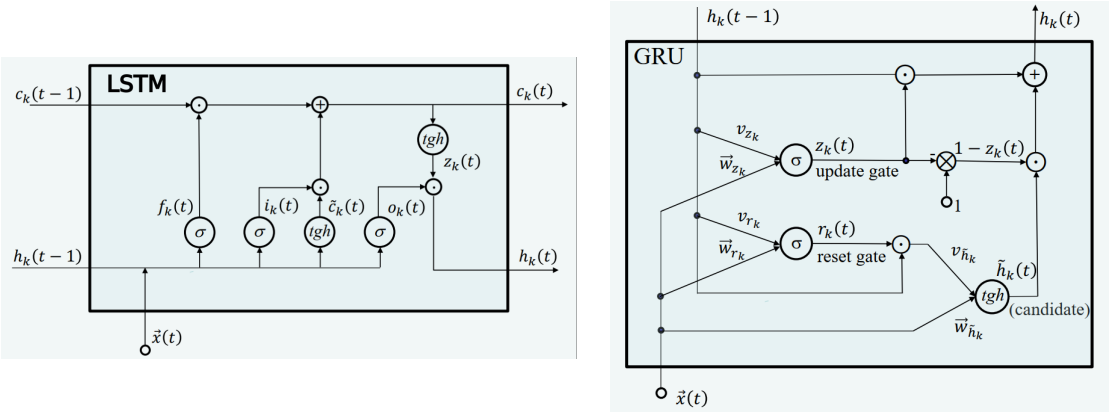


Fig. 1.9: LSTM and GRU cell architecture [89]

Sigmoid output is used as weight to penalize features and tangens output to transform features. Series of operations between cells input and its short and long memory variants are defined by term gate. Cell process for k-th cell in time t can be explained in following steps. At first, input $\vec{x}$ with short memory h is concatenated and transformed to act as f forget weights to penalize c, making it forget unnecessary features. Concatenated $\vec{x}$ and h is used again to tranform itself into $\widetilde{c}$ and at same time penalize by itself i to learn from current input the most essential features. These are then added to cells long memory c. One last time is concatenated $\vec{x}$ and h used as o penalize transformed cell memory z to make output h, which will act as short memory for next iteration of process in time $t+1$ [42]. Network can now learn to memorize important temporal features, but it has slight shortcoming in form of need to train network on sequences which results in slower training.

1.6.2 Gated recurrent unit

Memory cell of GRU does not contain explicit represantation of cells inner state like LSTM and uses only two gates as can be seen on right side of figure 1.9. Last output h and k-th input vector $\vec{x}$ in time t are used as input to these gates. Reset gate is used to set how much last output h will be part of so called candidate $\widetilde{h}$ and update gate determines ratio between summed elements $\widetilde{h}$ and h to create new output. GRU cell is not as proficient as LSTM in long memory features memorization, but thanks to its simpler structure its propagation is faster [16].

2 Convolutional neural networks

Discrete convolution is popular mathematical operator for image processing. Its 2D operator form is defined by matrix called kernel which iterates over image pixel matrix. Individual values of kernel multiply with overlapping image pixel values and sum of these multiplications is output containing spatial information about overlapped image pixels 2.1. This spatial information representation is unique and useful, because it carries validity of presence of certain shape with absence of other shapes. Convolution of one kernel overlapping with image pixels can be redefined as one neuron linear base function with input vector $\vec{x}$ as pixels values, and weight vector $\vec{w}$ of kernel values 2.2. Kernel values are then learned during training. This makes convolutional neural network (CNN) able to learn detect small features like different edge shapes from image by itself [48][21][52].

$$\mathbf{conv}(i,j) = \sum_{m=-W_k/2}^{W_k/2} \sum_{n=-H_k/2}^{H_k/2} x(i-m, j-n)k(m,n) \tag{2.1}$$

$$\mathbf{conv}(i,j) = \sum_{index=1}^{W_k H_k} x_{index} k_{index} \tag{2.2}$$

2.1 Main layer types

These layers are components of CNN core functionality fall. For general CNN this consist of convolution layers for encoding the features, activation layers for neuron output processing, pooling layers for feature dimension reduction and fully connected layers tasked with regression or classification of CNNs final output based on incoming prior layers output.

2.1.1 Convolution layer

Convolution layer makes convolution operation over 2D input, defined by kernel size f (for rectangular kernel) and two other parameters. First one stride s represents length of step, kernel makes over input pixels. And second is padding p, representing expansion of input boundaries with zeros, mostly used for keeping output of convolution same size n as original input. Equation (2.3) can be calculated to get convolution output size n_{new} [89][48][52].

One kernel output is called activation map, where high values represent activation for feature encoded in kernel weights. Each convolutional layer can have multiple filters each trained to respond to certain feature. Layer output is then a set of

$$n_{new} = \frac{n - f + 2p}{s} \quad (2.3)$$

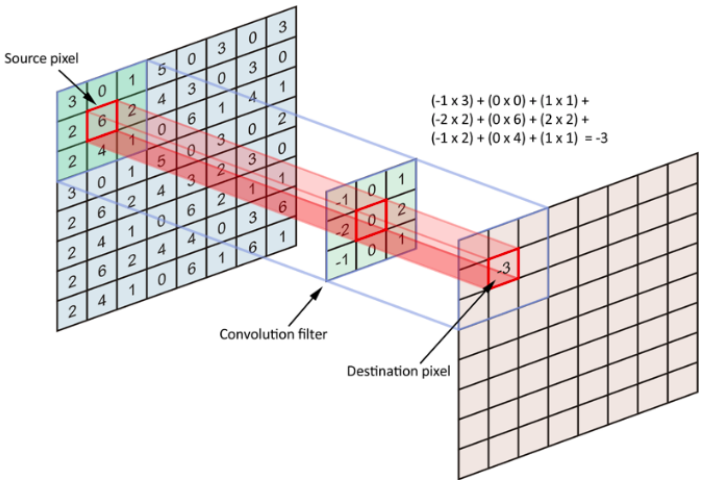


Fig. 2.1: Convolution operation visualization (right side) [107]

feature activation maps. CNN that uses only convolution and activation layers is called fully convolutional neural network (FCN) [52].

2.1.2 Activation layer

Without activation layers CNN would be simply linearly mapping x to y. Where for different x, y values would span over a large range of values. So for classification problems activation of output fully connected layer is needed to create probabilities for each class. Activation layers have AF they apply on all values in activation maps, leading to point-by-point nonlinear transformation of activation maps into feature maps [89][48][21].

2.1.3 Pooling layer

Pooling layer helps in achieving greater redundancy in CNN propagation and makes it invariant to small translation of detected object in input. There are many pooling operations, current most popular are max pooling and average pooling. Pooling can be used as a form of downsampling, where if there is sufficient activation in one part of region, this activation will transferred further while reducing region of values to one. This significantly improves CNN computational performance and invariance to slight input differences for one desired output. Pooling dimension reduction depends on chosen stride of pooling kernel. Most used pooling operation is $2x2$ size kernel with stride 2, feature maps have then reduced its dimensions to half [89][48][21][52].

2.1.4 Fully connected layer

Fully connected layers are responsible for transformation of feature maps to desired output format. And for understanding impact of individual features on final output. Since each neuron requires all feature values, 3D features of input to fully connected layers, must be first "flattened" to 1D vector, by concatenation. Final number of neurons in last output layer is decided by the problem NN solves. For regression its number of regressed values, for classification problem its number of classified classes [89][48][21].

2.2 Support layer types

Support layers provide additional stability and accuracy to NN by implementing various data science techniques on to propagated features maps. This helps to avoid NN problems covered in Chapter 2. And lowers amount of data needed for training.

2.2.1 Dropout layer

Purpose of dropout layers is to avoid overfitting by dividing NN to sub-nets, each trained to be responsible for different type of feature. This is done simply by introducing parameter p as probability of turning off neurons between batches. Turned off neurons do not learn, so other neurons learn more from batch on how to get desired feature. This makes neuron differentiated in their purpose, helping to achieve accuracy with lower number of neurons. To cover loss of energy from disabled neurons, output from layer is multiplied by p. Trained subnets are combined by weight sharing during training. This means that in final NN, after training phase ends, dropout layer is disabled and output multiplication with p is no longer needed. Dropout layer has to be always before fully connected or convolution layer [89][48][52].

2.2.2 Batch normalisation layer

Weigth updates (plural) backpropagate after NN processes one batch. But batch can be comprised from inputs with extreme individual differences in feature values. For NN it is easier to learn from more generic representation then specific instances of values stored within batch. To implement this concept, batch B of outputted d feature activation map is normalised by equation 2.8. Where ϵ is small constant to exclude division by zero situation, μ_{Bd} and σ^2_{Bd} are batches mean and variance. Normalised data are then scaled and shifted by learnable parameters γ_d and β_d 2.9.

$$\mu_{Bd} = \frac{1}{m}\sum_{i=1}^{m} x_d^{(i)} \qquad (2.4) \qquad \sigma^2_{Bd} = \frac{1}{m}\sum_{i=1}^{m}(x_d^{(i)} - \mu_{Bd}) \qquad (2.5)$$

$$\mu_{Dd} = \alpha\mu_{Dd} - (1-\alpha)\mu_{Dd} \tag{2.6}$$

$$\sigma^2_{Dd} = \alpha\sigma_{Dd} - (1-\alpha)\sigma_{Dd} \tag{2.7}$$

$$\hat{x}_d^{(i)} = \frac{x_d^{(i)} - \mu_{Bd}}{\sqrt{\sigma^2_{Bd} + \epsilon}} \tag{2.8}$$

$$\hat{h}_d^{(i)} = \gamma_d \hat{x}_d^{(i)} + \beta_d \tag{2.9}$$

After training BN layers is turned off, and instead input is scaled and shifted in equation 2.10. Where μ_{Dd} and σ^2_{Dd} are mean and variance of feature d from entire training data set. BN helps to avoid overfitting, since its linear transformation meaning not computationally hard to implement. If BN is not needed NN can just learn to have $\gamma_d = 1$ and $\beta_d = 0$, thus creating just identity function. It makes training less dependent on initial values and enables faster training process. Its effect is greater with bigger batch size. Normalisation in general reduces need to preprocess data before training [89][48][52].

$$\hat{h}_d^{(pred)} = \frac{\gamma_d}{\sqrt{\sigma^2_{Dd} + \epsilon}} x_{pred} + \beta_d - \frac{\gamma_d \mu_{Dd}}{\sqrt{\sigma_{Dd}}^2 + \epsilon} \tag{2.10}$$

2.2.3 Transposed convolution layer

Transposed convolution layer is a special upsampling method that uses convolution with sparse kernel to expand feature in height and width. Classic interpolation for upsampling image, use nearest neighboor, bi-linear or bi-cubic interpolation, but when transposed convolution is used, we allow NN to learn on its own, the way to interpolate values from features. This is useful for super-resolution CNN solving super-resolution, but upsampled features are damaged by checkerboard artifact. This artifact can be filtered by subsequent convolution [26][48][70].

2.3 Special blocks

Special blocks are composed from solutions to the vanishing gradient problem. So NN can use information from data to the fullest potential.

2.3.1 Skip connections block

Solves vanishing gradient problem by passing activation maps from earlier layers to later layers. This reintroduces raw background information to the propagated processed input containing only features activation. This makes up for loss of information from propagation and helps backpropagated gradient reach first layers. Data

are joined by concatenation of earlier feature maps to depth dimension of processed feature maps. Best used for shallow NN [89][48].

2.3.2 Residual block

In deeper NN its hard for gradient to reach first layers, so optimalisation of these layers is limited. Solution to this problem is by making residual blocks. These blocks have two branches, one with regular convolution layers and second only passing unproccesed current input otherwise called identity function. Output of these branches are summed together. Since identity branch is unchanging constant part of block output, the remaining part of output either improves the final block output or NN learns to nullify its weights. Thanks to identity of unprocessed input, can backpropagation gradient reach further back to first layers by using identity branch [89][48].

2.3.3 Point-wise convolution

Is in-depth combination of input features to output features. Its kernel is size 1x1 this means that it preserves feature dimensions but recalculates point by point value from all features into new combination of features. This can be used to lower number of feature maps while having effect of fully connected layer. Overall speeds up computational speed and lowers hardware requirements [48][89].

2.3.4 Diluted convolution

To broaden view of kernel in convolution, it can take feature values not next to each other like classic convolution but with fixed gaps δ. This way kernel learns more contextual information from features allowing it to have wider receptive field [48][99].

2.4 Examples

While use of convolution operator in NN can be traced as far back to 1972 Neocognitron [32]. First modern CNN architecture was proposed in [53]. But its potential was first noted with Alex Krizhevsky AlexNet [49] winning ILSVRC challenge. This started new wave of AI development which reverberates still. Here are examples of most renown CNN architectures, each introducing new principles used as of now in today CNNs.

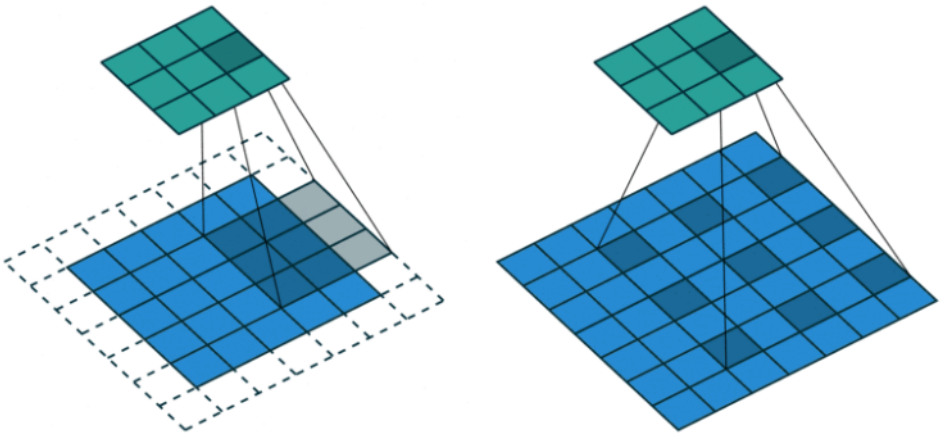


Fig. 2.2: Comparison of classic and dilated convolution with $\delta = 2$ [107]

2.4.1 AlexNet

AlexNet architecture was composed from series of 5 convolution layers and 3 fully-connected layers. Its inputs were RGB 224x224 images. Outputs from convolution layers, were pooled and its AF, was ReLU function. Notable feature of AlexNet is its use of dropout layers. Output from convolution was flattened and propagated through fully connected layers and final layer output used SoftMax function to access class probabilities. This simple and fast to learn architecture was able to outperform other competitors by nearly 10%. AlexNet had 650000 neurons and 60 million hyperparameters total [49].

2.4.2 VGGnet

Following CNNs architectures, were much more deeper with better results. But number of hyperparameters started to rise exponentially. One of them VGGnet was first to recognise that use of bigger kernel sizes than 3 is redundant. Differing versions of VGGnet are the largest NN with nearly 150 million hyperparameters [85].

2.4.3 ResNet

Vanishing gradient problem is often unavoidable in deeper architectures. To resolve this ResNet employed residual blocks. These blocks used short skip connections, making NN able to learn identity function at blocks place. Letting gradient flow through identity branch of block. ResNet also used batch normalisation, effectively reducing training time [39].

2.4.4 InceptionV

First Inception NN introduced so called inception module which was composition of multiple parallel branches. These branches used differing convolutional bottleneck layers whose output was concatenated in depth at end of the module. This enhanced with batch normalisation lead to even more improved training time reduction [86].

2.4.5 CapsNet

Newest promising CNN architecture makes use of so called capsules. These capsules are sets of neurons which unlike traditional CNN architectures, enable NN to learn features by equivariant mapping and hierarchy of parts. This means that NN is able to understand direction and spatial information about feature. CNN is able to classify object by presence of features at location but cant detect unrealistic pose, skale or associate object like arm to specific person. This problem is called Picasso problem and CapsNet should be a way to overcome it [83].

3 Facial detection

Finding human face on image is for machine task with two phases. In first phase face-like object is found. Then in second phase detected object is classified as human face. These two phases termed localization and classification of each object embodies every object detection (OD) task. To train facial detector means to train object detector network on data set with anotated locations of human face. Automatic face detection is the prerequisite step of facial image analysis for many applications such as facial attributes analysis (age estimation,expression,gender etc.).

3.1 Deep learning object detection fundamentals

All CNNs used for OD tasks, have in common functionality of parts of architecture or provide results in form that allows comparison of their outputs while making them human readable.

3.1.1 Transfer learning

It has been found that first CNN layers weights tend to find same set of reccuring small patterns. That is why in OD frameworks typically transfer learning is used. Transfer learning means that weights can be reused from NN trained to solve general OD tasks, like ImageNet or MS COCO data sets, to more specific OD task like facial detection [48].

3.1.2 Feature extractor

Feature extractor (FE) reduces image to a fixed set of features essential to solution of OD task. They do not necceserily need to be deep learning based, but other computer vision approaches can be used. In most cases are used architectures of general purpose CNN classificators like ResNet, InceptionV or VGGnet.

3.1.3 Bounding box

Most often used standard for inscribing object on image location, is by visualising axis-aligned boxes in an image. This boxes are called Bounding box(BB). BB location is represented in 4 parameters, 2D coordinates of upper left corner and its height and width. In OD NN tries to fit on object BB of different shapes. These shapes are called anchor boxes. Each one has confidence score that represents probability that given anchor box contains object. Additionaly each BB has also class

probability for each class NN is supposed to classify. So the final BB size is determined tensor of anchor boxes their confidence scores and geometric parameters and class probabilities [48].

3.1.4 Intersection over Union

Special loss function for evaluation of difference between ground-truth(GT) BB and predicted BB, for objects in general is known as Jaccard similarity index. Its computed as ratio of their overlapping areas to their areas union (3.1). Loss needed for training and evaluation is then calculated simply by subtraction (3.2).

$$IoU = \frac{\boldsymbol{I}}{\boldsymbol{U}} = \frac{B^g \cap B^p}{B^g \cup B^p} \quad (3.1) \qquad E_{IoU} = 1 - IoU \quad (3.2)$$

But IoU downside is that it will not take into account difference between closely and distantly missed prediction BB. Also in this situation there is zero gradient, so to have better convergence and more accurate regression, modified IoU were recently proposed. Generalized IoU introduces smallest convex shape (or rectangle) C enclosing area of both predicted and GT BB. $GIoU$ uses normalized area A^C of this shape excluding area of BB union U to represent free space between BBs. This means that $GIoU$ has in gradient all possible cases, including non-overlapping situations. As of now only 2D analytical solution was proposed [80].

$$GIoU = \frac{\boldsymbol{I}}{\boldsymbol{U}} - \frac{A^C - U}{A^C} \quad (3.3) \qquad DIoU = \frac{\boldsymbol{I}}{\boldsymbol{U}} - \frac{\rho^2_{B^g,B^p}}{c^2} \quad (3.4)$$

While $GIoU$ solves vanishing gradient problem it tends to increase prediction BB area first and then increase IoU in its equation. This leads to longer convergence time. Solution to this is Distance-IoU which directly minimalizes normalized distance between center points of GT BB and prediction BB. By giving giving IoU penalty term in equation 3.4 . Where ρ_{B^g,B_p} is euclidean distance between central points and c is diagonal length of shape C which encloses GT BB and prediction BB. Concurrently in same paper, Complete-IoU was proposed. that adds to overlap area, central points distance another geometric factor that is match of BBs aspect ratio. With positive trade off α parameter and v consistency of aspect ratio 3.5 (h,w - BB height,width). Trade off α gives higher priority to overlapping area. Final $CIoU$ has form 3.7. Reason why same paper introduces two IoU losses is that $DIoU$ can be also used to improve non-maximum suppression (NMS) of overlapping region proposals [93]. All modified IoUs can be used to calculate loss like in equation 3.2. For $DIoU$ novelty, only IoU will be used as verified overlapping metric for explanation purposes of this thesis.

$$v = \frac{4}{\pi^2}(arctan\frac{w^g}{h^g} - arctang\frac{w^p}{h^p})^2 \quad (3.5) \qquad \alpha = \frac{v}{(1 - IoU) + v} \quad (3.6)$$

$$CIoU = \frac{\boldsymbol{I}}{\boldsymbol{U}} - \frac{\rho^2_{B^g,B^p}}{c^2} - \alpha v \quad (3.7)$$

3.1.5 Non-maximum suppression

Last step in most OD algorithms,in which redundant detection boxes are removed, when their IoU with the highest confidence score BB exceeds a certain threshold. After NMS remain only few region proposals from which BB with highest class probability is picked [8].

3.1.6 Mean average precision and Average recall

Evaluation metrics Mean average precision (mAP) and Average recall (AR) are commonly used to compare accuracy of different multi-label OD algorithms. Precision and recall for one class is defined by equations 3.8,3.9. Where TP is count where predicted BB must satisfy three conditions. Its confidence score must be above threshold, predicted class matches GT BB class and its IoU must be above threshold. If latter two conditions are not met then BB counts as FP. Not detected GT BB are counted as FN, but since FN is used only in denominator as sum equal to known number of GT BB, there is no need to keep FN count.

$$precision = \frac{TP}{TP + FP} \quad (3.8) \qquad recall = \frac{TP}{TP + FN} \quad (3.9)$$

Precision and recall is used to plot precision-recall curve (PRC) for different thresholds of overlapping metric. PRC is used to rate performance of detector. From PRC is hard to compare different detectors so precision in single scalar metric is wanted. For this average precision AP is used, but since PRC has a lot of sharp fluctuations in values, only highest precision value $max\ p(r')$ for any r' - recall level($r' > r$) is used, to create interpolated PCR $p_{interp}(r)$ 3.10. From this AP is calculated as area under interpolated PCR 3.11. Getting mAP is then simply calculating mean AP for all classes 3.12 [41].

$$p_{interp}(r) = \max_{r'>r} p(r') \quad (3.10) \qquad \mathrm{AP} = \sum_{i=1}^{n-1} (r_{i+1} - r_i) p_{interp}(r_{i+1}) \quad (3.11) \qquad m\mathrm{AP} = \frac{\sum_{i=1}^{K} \mathrm{AP}_i}{K} \quad (3.12)$$

Another use for recall is to calculate AR as two times the area under recall-IoU curve where IoU is from range from 0.5 to 1 3.13. In this equation o represents IoU and $recall(o)$ is corresponding recall value. Different OD challenges use to evaluate detectors by their AR individually for each class or use mean AR 3.14 [50].

$$\mathrm{AR} = 2 \int_{0.5}^{1} recall(o) do \quad (3.13) \qquad m\mathrm{AR} = \frac{\sum_{i=1}^{K} \mathrm{AR}_i}{K} \quad (3.14)$$

3.1.7 Hard negative mining

Best performing classifier needs both positive and negative training samples. Positive training sample has BB with GT detection of object, while negative training

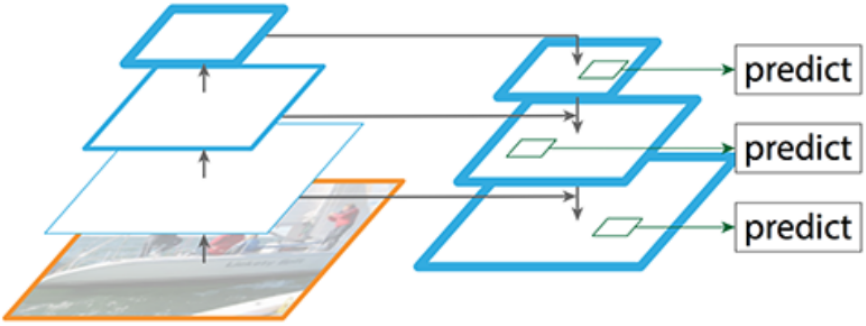


Fig. 3.1: FPN with skip connections [107]

sample has BB with GT lack of object. Hard samples are samples with big classification loss. Hard negative mining is a bootstrapping inspired method that after classification with weakly trained classifier, uses false detected BB, and explicitly uses them to create hard negative sample which is added to training data. Retrained classifier should then yield better results.

3.1.8 Feature pyramid networks

To detect object of differing scales, prediction layers has to take FE feature maps from layers at different depth or used as input image of different sizes. This is computationally expensive and inaccurate so combined approach in form of Feature pyramid networks (FPN) on figure 3.1 solves this. FPN is composed of a convolutional bottom-up and upsampling top-down pathways. With smaller feature size in higher depth rises features semantic information, but resolution with spatial information is lost. By using top-down pathway to create higher resolution features from semantic rich layer and adding earlier higher resolution information by skip connection, is final accuracy greatly improved and whole process is solved in non computationally expensive manner [57].

3.2 Deep learning object detection approaches

All OD algorithms use FE to get features from which BBs are calculated, NMS sifted and finally evaluated by differing metrics. Main split in algorithms solving OD is in their approach to getting individual BB propositions. OD algorithms performance is mostly evaluated on OD challenges like ImageNet, MS COCO data set (3.2) or PascalVOC data set(3.1) [22][59][29].

3.2.1 Faster R-CNN

Most commonly used object detection CNN. Its current version nicknamed faster, name derived from substantional gains in detection per second speed in comparison to first versions. Its conjoined network of the two CNNs sharing FE pretrained on ImageNet. First deep CNN called region proposal network is used on extracted features to get 2000 region proposals representing possible BB locations. Overlapping regions proposals are NMS merged to lower the number of regions. Second CNN is Fast R-CNN (region-CNN) that has task of object localization in these regions and subsequent object classification into correct classes. Training such complex architecture has multiple different phases. And since it uses region proposal network to propose regions and second Fast R-CNN to detect object in them, is this method called two-stage and that makes it time consuming. But properly trained faster R-CNN is overall the most accurate object detection CNN [35][36][79].

3.2.2 Single shot detector

Faster OD algorithms like single shot detector (SSD) gain speed from not using region proposal network to generate region proposals, but implement FE and OD in single deep NN, which makes them one-stage method. SSD approach utilizes as base pretrained classification CNN like VGGNet, from which draws feature maps at differing depth for OD anchor BB predictions. Each anchor BB has his spatial geometric parameters regressed and class probabilities estimated at each drawn depth level. By combining various resolutions feature maps, SSD can handle objects of various sizes. Augmenting SSD with additional transposed convolution layers to feed-forward additional large-scale context helps object detection and improves accuracy and offsets error for small objects, [61][31].

3.2.3 You only look once detector

Next one-stage method is called You only look once detector (YOLO). This approach divides image into grid of cells, where for each cell all anchor BB and their parameters are predicted. All this takes place in form of FCN, so YOLO is extremely fast and invariant to the input image size. One grid cell detects only one object, this limits size of object YOLO is capable to detect. So YOLO struggles with smaller objects but has lower number of false positives then Faster R-CNN. Thanks to using whole image for classification rather then individual regions, which allows flow of contextual information to output layers. Next two YOLO iterations gained in speed, accuracy and improved small object detection. From old deep learning approaches is YOLO first in speed of prediction per second [78].

3.2.4 Focal Loss for Dense Object Detection

Is a composite one-stage network composed of FE-FPN backbone network and pair of parallel subnetworks using backbone network output. First subnetwork is used for object classification and second for BB regression. Each feature map drawn at different scale has its own pair of subnetworks. RetinaNet output of BB predictions is similarly gained from proccessed anchor boxes like in other OD algorithms. At lower resolution images RetinaNet has similar accuracy as Faster R-CNN with twice as fast predictions [58].

3.2.5 Objects as points

New modern approach treats objects as a single point, center point of its BB. This way it circumvents the need for brute force search through potential object locations and their classification. Instead it uses keypoint estimation to find object center points and regresses other object properties like size, orientation and pose. CenterNet architecture claims to be simpler, faster and more accurate then other BB based detectors [94].

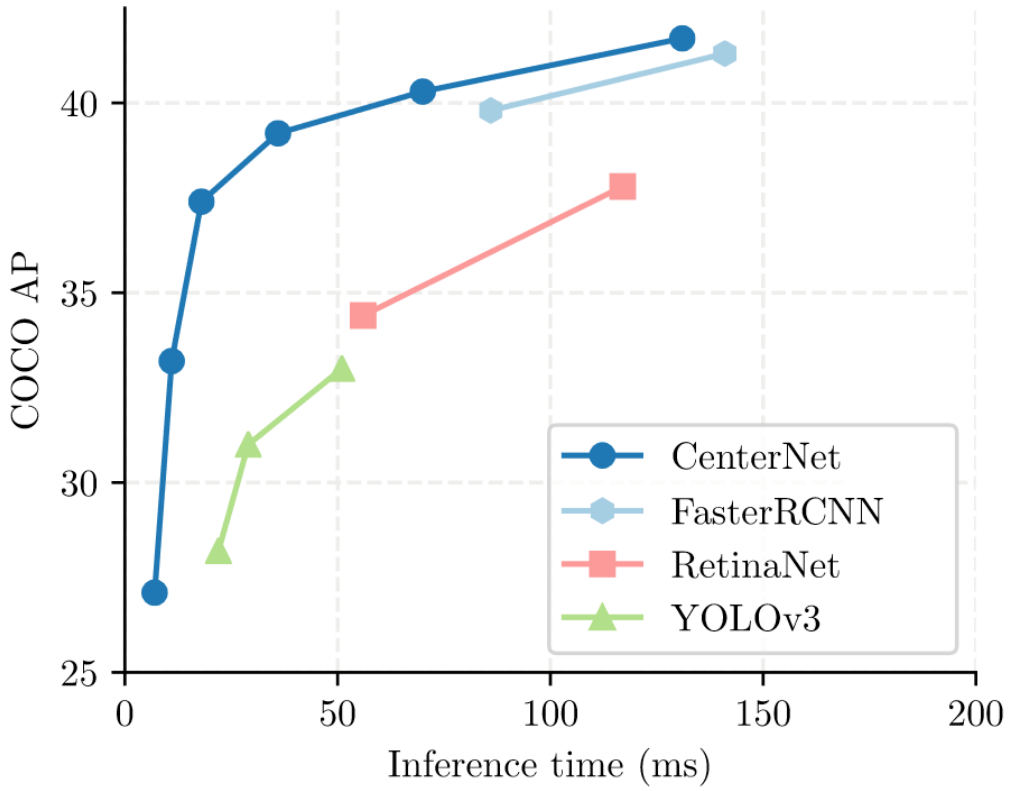


Fig. 3.2: Speed-accuracy trade-off on COCO validation for some of the state-of-the-art real-time detectors [94].

Tab. 3.1: Table of comparison for state-of-the-art OD algorithm performance on PascalVOC 2007 test (April 2019). The results are shown in mAP@0.5. The FPSs for other methods than CenterNet are copied from the original publications [94].

	Resolution	mAP@0.5	FPS
Faster RCNN	600×1000	76.4	5
Faster RCNN*	600×1000	79.8	5
R-FCN	600×1000	80.5	9
Yolov2	544×544	78.6	40
SSD	513×513	78.9	19
DSSD	513×513	81.5	5.5
RefineDet	512×512	81.8	24
CenterNet-Res18	384×384	72.6	142
CenterNet-Res18	512×512	75.7	100
CenterNet-Res101	384×384	77.6	45
CenterNet-Res101	512×512	78.7	30
CenterNet-DLA	384×384	79.3	50
CenterNet-DLA	512×512	80.7	33

3.3 RetinaNet object detector

Reason why one-stage methods are inferior in accuracy to two-stage methods was determined to be in extreme foreground-background class imbalance. There are three groups of solutions: hard sampling methods, soft sampling methods and generative methods. RetinaNet is one of the soft-sampling solutions. Sampling methods rely on selecting samples from a set of bounding boxes, while generative methods generate samples. In sampling methods, the contribution (e_{BB_i}) of a BB_i to the Cross-Entropy loss function CE is adjusted, where p_t is the estimated probability for the ground-truth class 3.15. Hard sampling has e_{BB_i} with binary values, effectively discarding or selecting BB. On the other hand soft sampling approach weights $e_{\mathrm{BB}_i} \in (0, 1)$ contributions, so all BB loss is included [71].

$$\mathrm{CE} = \sum_{i=1}^{N} e_{\mathrm{BB}_i} CE(p_t) \tag{3.15}$$

3.3.1 Focal loss

To properly explain focal loss, CE loss function must be explained as well. Starting with its version for binary classification 3.16. And its rewritten variant 3.18. Sample represents BB class probability. For this loss can sum of easy samples $p_t >> 0.5$ (easy to classify) have nontrivial magnitude that can overwhelm rare class with low number of hard samples. This happens because easy samples comprise the majority

Tab. 3.2: Table of comparison for state-of-the-art OD algorithm performance on COCO test-dev (April 2019). Top: two-stage detectors; bottom: one-stage detectors. One-stage detectors mostly multi-scale testing. Italic FPS highlight the cases, where the performance measure was copied from the original publication. A dash indicates methods for which neither code and models, nor public timings were available [94].

	Backbone	FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
MaskRCNN	ResNeXt-101	**11**	39.8	62.3	43.4	22.1	43.2	51.2
Deform-v2	ResNet-101	-	46.0	67.9	50.8	27.8	49.1	59.5
SNIPER	DPN-98	*2.5*	46.1	67.0	51.6	29.6	48.9	58.1
PANet	ResNeXt-101	-	47.4	67.2	51.8	30.1	**51.7**	60.0
TridentNet	ResNet-101-DCN	0.7	**48.4**	**69.7**	**53.5**	**31.8**	51.3	**60.3**
YOLOv3	DarkNet-53	20	33.0	57.9	34.4	18.3	25.4	41.9
RetinaNet	ResNeXt-101-FPN	5.4	40.8	61.1	44.1	24.1	44.2	51.2
RefineDet	ResNet-101	-	36.4 / 41.8	57.5 / 62.9	39.5 / 45.7	16.6 / 25.6	39.9 / 45.1	51.4 / 54.1
CornerNet	Hourglass-104	4.1	40.5 / 42.1	56.5 / 57.8	43.1 / 45.3	19.4 / 20.8	42.7 / 44.8	**53.9** / 56.7
ExtremeNet	Hourglass-104	3.1	40.2 / 43.7	55.5 / 60.5	43.2 / 47.0	20.4 / 24.1	43.2 / 46.9	53.1 / 57.6
FSAF	ResNeXt-101	*2.7*	**42.9** / 44.6	**63.8 / 65.2**	**46.3** / 48.6	**26.6 / 29.7**	**46.2 / 47.1**	52.7 / 54.6
CenterNet-DLA	DLA-34	**28**	39.2 / 41.6	57.1 / 60.3	42.8 / 45.1	19.9 / 21.5	43.0 / 43.9	51.4 / 56.0
CenterNet-HG	Hourglass-104	7.8	42.1 / **45.1**	61.1 / 63.9	45.9 / **49.3**	24.1 / 26.6	45.5 / **47.1**	52.8 / **57.7**

of the loss and dominate the gradient.

$$CE(p,y) = \begin{cases} -log(p) & y = 1 \\ -log(1-p) & y \neq 1 \end{cases} \tag{3.16}$$

$$p_t = \begin{cases} p & y = 1 \\ 1-p & y \neq 1 \end{cases} \tag{3.17}$$

$$CE(p,y) = -log(p_t) \tag{3.18}$$

First remedy to this problem is balanced CE, which introduces weighting factor $\alpha_t \in (0,1)$ 3.19. In practice its value is set by inverse class frequency or hyperparameter set by cross-validation. Even though it balances importance of easy/hard samples it does not differentiate between easy/hard samples.

$$\text{CE}(p,y) = -\alpha_t log(p_t) \tag{3.19}$$

$$e_{\text{BB}_i} = \alpha_t (1-p_t)^\gamma \tag{3.20}$$

$$FL(p_t) = -\alpha_t (1-p_t)^\gamma log(p_t) \tag{3.21}$$

Focal loss solves this by adding a modulating factor $(1-p_t)^\gamma$ to the the balanced cross entropy loss, with tunable focusing parameter $\gamma \geq 0$. So RetinaNet e_{BB_i} value is 3.20. Final equation for focal loss is then 3.21. For $\gamma = 0$ is loss function just vanilla balanced CE function. As for γ effect on loss it reduces the loss contribution from easy examples and extends the range in which an example receives low loss 3.3. As an example $\gamma = 2$ would give sample classified with $p_t = 0.9$, 100 times lower loss and for $p_t \approx .968$ it would have 1000 lower loss than with basic CE.

Leading to increased importance of correcting misclassified samples. This improves the prediction accuracy. Because it allows substantially larger amount(100k) of BB propositions than other one-stage methods like YOLO(98) or SSD(8k-26k)[58].

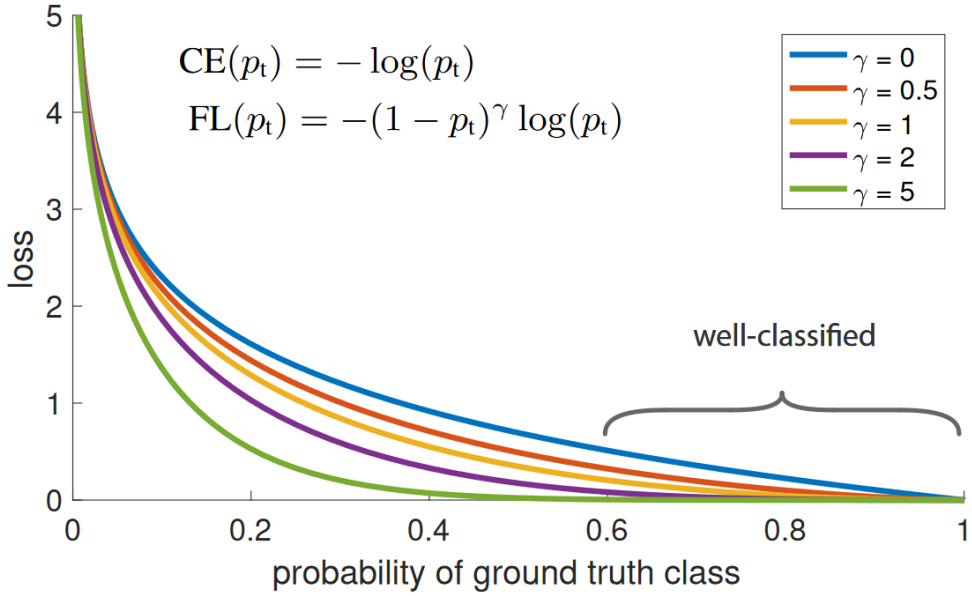


Fig. 3.3: Focal loss for different γ values, at higher values it penalizes loss from more probable easy samples [58].

3.3.2 Architecture

As a FE, original RetinaNet paper uses ResNet50(101) 3.4(a) with FPN on top 3.4(b), constructing a rich multi-scale feature pyramid from one single resolution input image. Larger backbone networks yield higher accuracy, but also slower inference speeds. There are another two task-specific subnetworks for classification and bounding box regression on backbones output. Classification subnet 3.4(c) predicts

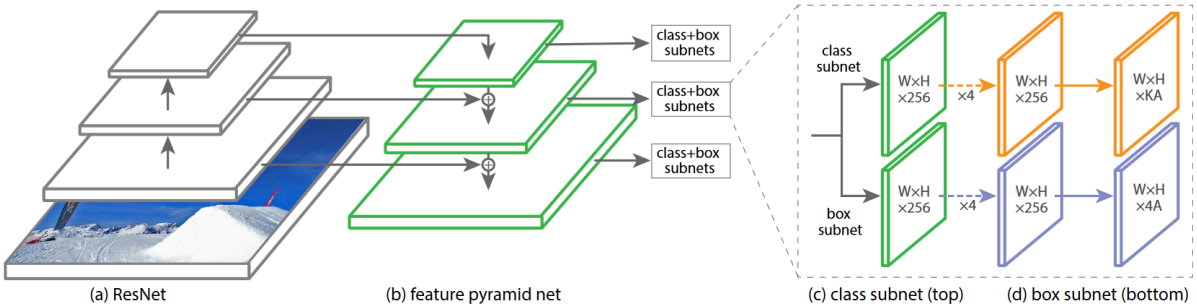


Fig. 3.4: The RetinaNet neural network architecture [58].

the probability of object presence at each spatial position for each of the A anchors and K object classes. Using backbone output feature map with C channels from

a pyramid level, the subnetwork applies four 3x3 conv layers, each with C filters and ReLU as AF. ReLU output enters sigmoid function and that gives final object probability outputs. Loss function it then computed as Focal loss. BB regression subnetwork 3.4(d) is a FCN at each pyramid level with goal of regressing offsets from anchor box to possible GT object. Its layers are identical to classification subnetwork, only difference is in that output layer is remodeled for regression with 4 linear outputs per anchor box and per spatial location. Since it regresses same BB values for all classes its termed as class-agnostic bounding box regressor which authors found equally effective [58].

3.3.3 Training

Before training starts authors have assumed a prior probability of 0.01 for all the anchor boxes and assigned this to the bias of last convolutional layer of classification sub network. This initialization is extremely important otherwise will loss function never converge to optimum. Idea behind prior probability value is that foreground (positive anchor boxes) to background (negative anchor boxes) objects ratio is $1000/100000 = 0.01$. Weight decay is set to 0.0001 with momentum of 0.9 and initial learning rate of 0.01 is used for first 60k iterations. Learning rate is divided by 10 after 60k iterations and 80k iterations. During training is total loss of an image computed as the sum of the focal loss over all 100k anchors, normalized by the number of anchors assigned to a ground-truth box [58].

3.3.4 Inference

Detected BB are thresholded by their BB confidence score at 0.05. From remaining BB predictions were selected 1000 top scoring predictions per FPN level. These are from all levels merged and NMS with a IoU threshold of 0.5 is applied to yield the final predictions [58].

3.4 RetinaFace facial detector

Is a current record holder for WIDER FACE Hard data set 3.5 [106][96]. This OD detector specialized for face detection is based on RetinaNet architecture [58]. It broadens goals of face detection from just traditional BB parameters prediction to include also face alignment, pixel-wise face parsing and 3D dense correspondence regression. This means that RetinaFace simultaneously predicts face score, face box, five facial landmarks, and 3D position and correspondence of each facial pixel.

With this level of dense face localisation, accuracy of facial position information, radically improves for all various face and image sizes [23].

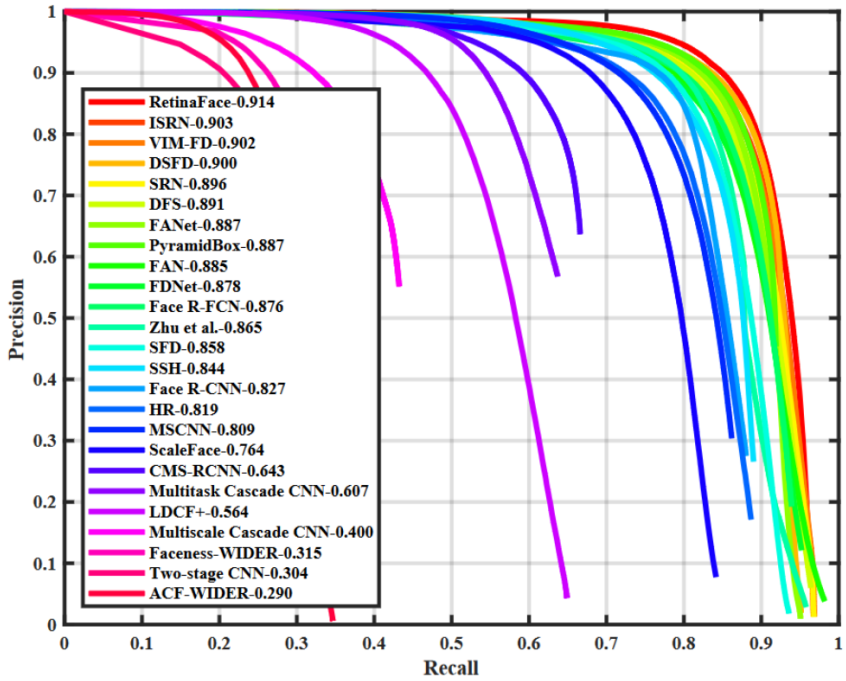


Fig. 3.5: Various facial detectors PCR curve on WIDER FACE Hard data set challange, in the legend, number represents AP [23]

3.4.1 Multi-task loss

Each training anchor i, has minimized following loss function 3.22.

$$L = L_{cls}(p_i, p_i^*) + \lambda_1 p_i^* L_{box}(t_i, t_i^*) + \lambda_2 p_i^* L_{pts}(l_i, l_i^*) + \lambda_3 p_i^* L_{pixel} \tag{3.22}$$

First element $L_{cls}(p_i, p_i^*)$ is face classification binary softmax loss, with p_i being probability of anchor box having a face and p_i^* equal to 1 for positive anchor and 0 for negative anchor. Second element $L_{box}(t_i, t_i^*)$ is box regression loss equal to $\mathrm{R}(t_i - t_i^*)$ where R is $smooth - L_1$ from [36], parameters $t_i = [t_x, t_y, t_w, t_h]$, $t_i^* = [t_x^*, t_y^*, t_w^*, t_h^*]$ are predicted and GT BB geometric transformation parameters associated with positive anchor. Third element is $L_{pts}(l_i, l_i^*)$ is facial landmark regression loss with l_x and l_y coordinates parameters for 5 landmarks and each predicted l_i and GT l_i^* landmarks. Their transformation and loss is computed same as with box regression loss. Fourth element L_{pixel} is Dense Regression Loss and has equation 3.23. Dense Regression Loss is pixel-wise difference of rendered 2D face $\mathcal{R}(\mathcal{D}_{P_{ST}}, P_{cam}, P_{ill})$ and GT face. W and H are width and height of anchor crop $I_{i,j}^*$. $\mathcal{R}$ is 3D mesh renderer used with

shape and texture parameters P_{ST} to project a colouredmesh $\mathcal{D}_{P_{ST}}$ onto 2D image plane with camera parameters P_{cam} and illumination parameters P_{ill}.

$$L_{pixel} = \frac{1}{W * H} \sum_{i}^{W} \sum_{j}^{H} ||\mathcal{R}(\mathcal{D}_{P_{ST}}, P_{cam}, P_{ill})_{i,j} - I^*_{i,j}||_1 \tag{3.23}$$

Last loss-balancing parameters λ_1,λ_2,λ_3 are weights with values 0.25,0.1 and 0.01 this means that BB and landmark loss is deemed more important than dense correspondence regression loss. Visualized multi-task loss is on figure 3.6.

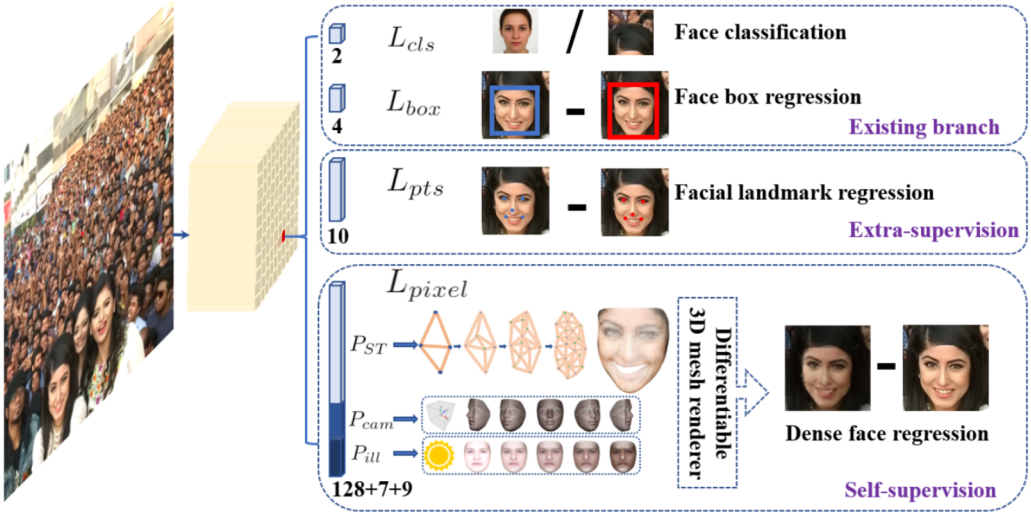


Fig. 3.6: RetinaFace one-stage pixel-wise face localisation employs extra-supervised and self-supervised multi-task learning in parallel with the existing box classification and regression branches [23]

Mathematical apparatus to predict P_{ST} uses mesh decoder with graph convolution method based on fast localised spectral filtering formulated as Chebyshev polynomial truncated at order K, with each order evaluated at scaled Laplacian $\hat{L}$. This filtering operation uses sparse matrices and is extremely fast but its thorough explanation is above extent of this thesis task so for full mesh decoder explanation one should read RetinaFace paper [23].

3.4.2 Training data

Authors anotated WIDER FACE data set (closer description in chapter 6) landmark anotations for eyes centers,nose and mouth corners. In total they anotated 84.6k faces on training set and 18.5k faces on validation set. Random crop data augmentation was used on 20% tiny faces in the WIDER FACE training set. Random crop square patches from the original images were resized to generate larger training faces. Other augmentation of training data was by random horizontal flips with the probability of 0.5 and photo-metric colour distortion [23].

3.4.3 Architecture

RetinaFace architecture contains FE-FPN conjoined network with its outputs flowing through context modules to shared loss head 3.7. From C1 to C5 level is FE a ResNet-152 classification network. Pre-trained on the ImageNet-11k data set while C6/P6 is randomly initialised with the "Xavier" method [100]. FPN levels from P2 to P5 are computed from the output of the corresponding FE level (C2 to C5) using top-down and lateral connections. P6 level is calculated through a 3x3 convolution with stride 2 from C5.

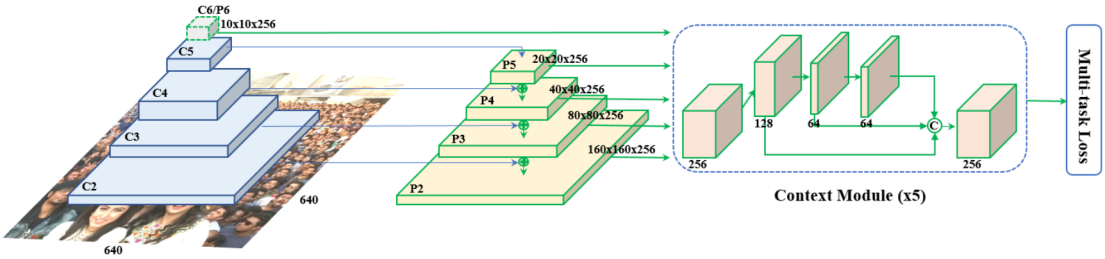


Fig. 3.7: The RetinaFace neural network architecture [58].

Independent context module is applied to each of the feature pyramid levels to increase the receptive field and enhance the rigid context modelling power. In lateral conections the deformable convolution network strengthens the non-rigid context modelling capacity [23].

Shared loss head (1x1conv) uses all different levels of feature maps H_nxW_nx256,$n \in [2, ..., 6]$. As mesh decoder is used pretrained network from paper [95].

3.4.4 Anchor boxes

Scale-specific anchors were employed on the feature pyramid levels from P2 to P6. Here P2 is designed to capture tiny faces by tiling small anchors at the cost of more computational time and at the risk of more false positives. Scale step was 2.33 and the aspect ratio at 1:1. For example input image with size 640x640, has anchors cover scales from 16x16 to 406x406 on the feature pyramid levels. In total, there are 102,300 anchors, and 75% of these anchors are for tiny faces detection on P2 pyramid level [23].

3.4.5 Training

In training anchors are matched to a GT BB when $IoU > 0.5$, and to the background when $IoU < 0.3$. Unmatched anchors do not count during training. To alleviate significant imbalance between the positive and negative training examples

hard negative mining is employed. So from negative anchors sorted by loss, ones with top loss value are selected so that the ratio between the negative and positive samples is at least 3:1. Positive anchors have multi-task loss calculated, negative anchors contribute only classification loss.

RetinaFace is trained using stochastic gradient optimiser with parameters momentum at 0.9, weight decay at 0.0005, batchsize of 8x4 on four NVIDIA Tesla P40 (24GB) GPUs. The learning rate starts from 10^{-3}, and rises to 10^{-2} after 5 epochs, then is divided by 10 at 55 and 68 epochs. The training process ends after 80 epochs [23].

3.4.6 Inference

With confidence threshold of 0.05 are most of the BB predictions filtered out. NMS is applied with IoU threshold equal to 0.3 on top 400 confidence score BB. Authors in the last step refined final BB parameters by using Box voting with IoU threshold value 0.4 [34]. Box voting changes final BB by using BBs from before NMS was applied to vote for final BB location if their IoU with final BB is above threshold. Each BB contribution to final BB location is weighted by his BB confidence score [23].

4 Gender and emotion classification

After face or body of a person is localized, whether in bounding box or segmented area, the next step is find appropriate way how from these pixel values extract features we want. Next sections provide brief review of methods used to determine gender and emotion, and then describe solution used by this work to predict both. To narrow wide field of methods these reviews focus only on related facial recognition methods from RGB or grayscale images.

4.1 Gender classification methods review

Face gender classification is done by extracting features from facial area and their subsequent classification. Conventional approaches created by experts can be categorized into older geometric and newer appearance-based approaches [68]. These approaches can be also used together to supplement each other.

In geometric approach points that mark important features of the face, such as eyes, nose tip, mouth corners, ears and chin, are called facial landmarks or fiducial points. Fiducial distances are the distances between these points, they are then used determine gender. For example 18 point-to-point distances to train a hyper basis function network classifier [9], or selecting 40 points to calculate 22 normalized vertical and horizontal fiducial distances and from them five dimensions were derived using discriminant analysis and then used to classify gender [30].

In appearance-based approach, methods are based on some operation or transformation performed on the pixels of an image. This can be done at the global or local level. At the global level, features are computed from the whole image resulting in a single feature vector. In local feature extraction, the image is partitioned beforehand into regions such as eyes, nose and mouth areas [68]. A feature vector is then obtained from each patch, and used to train classifier. So therefore methods differ mainly in type of features they use. Since there is massive number of them focus will be on newer methods.

Using pixel intensity values as features combined into vectors obtained from pixels of the whole face image, local face regions and the gradient image. The vectors are combined to form a single vector on which Principal Component Analysis was then applied [10]. In similar way Linear Discriminant Analysis can be utilized [6].

Local binary patterns (LBP) have myriad of methods, one of them uses Adaboost to learn discriminative multiscale LBP histogram bins. Patterns are here extracted at different resolutions and subregions of the image [84]. Other similar patterns to LBP were proposed [68].

The diversity of appearance-based features that can be used is enormous so here are some other examples: likelihood values from a regression function ,pixel pattern-based texture feature, Viola and Jones rectangle features, AdaBoost used with on weak simple pixel comparison operations, Scale Invariant Feature Transform descriptor, Weber Local Descriptor, Histogram of Oriented Gradients, Discrete Cosine Transform, wavelets of Gabor filters, wavelets of Radon transform, multiscale filter banks consisting of first- and second-order Gaussian derivative and Laplacian of Gaussian, eigenmoments and pseudo-Zernike moments [68].

Newer deep learning approach uses neural networks to perform end to end predictions, where the feature extraction is integrated with classification. First well established method using CNN was [55], others followed [3], [4]. Last mentioned CNN method is used in this work and will be covered in section 4.3 .

4.2 Emotion classification methods review

In same way as with gender, emotion is classified from pixels of detected face, with same approaches: conventional geometric or appearance-based methods, and deep learning methods. But unlike gender more emotion methods are based on video frames taking advantage of spatio-temporal information in expression dynamics of facial expression sequence [47]. These methods deliver better results but have downside that they have high computational cost. Second difference is that emotion classification works with much more class types.

There is Basic Emotion group: happiness, surprise, anger, sadness, fear disgust, and neutral, then there is Compound Emotion group which consist of 7 basic emotions, 12 compound emotion (combination of 2 basic emotions) with three rare emotions appeal, hate and awe. Lastly there are Micro expressions which indicate more spontaneous and subtle facial movements that occur involuntarily [47]. And then exist the most descriptive The Facial Action Coding System (FACS) to encode the movements of specific facial muscles called action units, which reflect distinct momentary changes in facial appearance [28].

As with gender here are also features extracted and then classified. Geometric based methods use as features the relationship between facial components to construct a feature vector for training. For example using two types of geometric features based on the position and angle of 52 facial landmark points, then calculating angle and Euclidean distance between each pair of them within a frame. Following with the distance and angles subtraction from the corresponding distance and angles in the first frame of the video sequence. For the classifier, two methods are presented, either using multi-class AdaBoost with dynamic time warping, or using a Support Vector Machines on the boosted feature vectors [33].

The appearance-based methods use features from the global or local regions of face. This is similar to gender appearance methods with most of the methods functionality overlapping. An example of using global features utilized a LBP histogram of different block sizes from a global face region as the feature vectors, and classified various facial expressions using a Principal Component Analysis. Example of local region method is extraction of region-specific appearance features by dividing the entire face region into domain-specific local regions. Important local regions are determined using an incremental search approach, which results in a reduction of the feature dimensions and an improvement in the recognition accuracy [38].

Methods using video sequences in addition to spatial features measure the geometrical displacement of facial landmarks between the current frame and previous frame as temporal features. The main difference between still images and video sequence based methods is that the landmarks in the latter are tracked frame-by-frame and new dynamic features are generated through displacement between the previous and current frames. For example dividing face regions into specific regions and then 3D-Gradients orientation histogram is generated from the motion in each region as feature for classification [76].

Deep learning methods employ for emotion classification CNN architecture sometimes modified with recurrent neural network cells. Standalone CNN methods adapted directly for FACS action units detection. One method used used two different types of CNN: the first extracts temporal appearance features from the image sequences, whereas the second extracts temporal geometry features from temporal facial landmark points. These two models are combined using a new integration method to boost the performance of facial expression recognition [46]. Other standalone CNN proposed deep region and multi-label learning where region layer uses feed-forward functions to induce important facial regions,and forces the learned weights to capture structural information of the face [92].

However, because CNN-based methods cannot reflect temporal variations in the facial components, hybrid approaches combining CNN and RNN emerged, there are many of them [47], but in general their framework can be described as combination of an LSTM with a deep hierarchical visual feature extractor such as a CNN model. Therefore, this hybrid model can learn to recognize and synthesize temporal dynamics for tasks involving sequential images. And it does that by passing visual features from CNN, to the corresponding LSTM. Where is produced a fixed or variable-length vector representation. The outputs are then passed into a recurrent sequence-learning module. Finally, the predicted distribution is computed by applying softmax [24].

These methods offer good result but their inference speed suffers and as emotion is not main feature of this paper, more faster standalone CNN is used [4]. Which

will be examined in next section.

4.3 CNN used for gender and emotion classification

Most of the CNN parameters tend to be in the fully connected layers at the end of architecture. Modern CNN architectures such as Xception tend to reduce amount of these parameters by using residual modules and depth-wise separable convolutions [18]. This way separated the processes of feature extraction and combination within a convolutional layer, need less parameters to function with same accuracy. Network mini-Xception used for classification of gender and emotion in this paper is based on Xception architecture which was was further improved to have best accuracy over number of parameters ratio [4].

4.3.1 Training data

Authors used IMBD data set to to train gender variant of mini-Xception. This data set which contains 460,723 RGB images where each image belongs to the class "woman" or "man" [82]. For emotion variant they used FER-2013 data set [88]. In this data set there are 35,887 grayscale images where each image has person whose facial expression belongs to one of the classes of basic emotion group.

4.3.2 Architecture

The mini-Xception architecture on figure 4.1, is a FCN that contains 4 residual depth-wise separable convolutions where each convolution is followed by a batch normalization operation and a ReLU activation function. The last layer applies a global average pooling and a soft-max activation function to produce a prediction. This architecture has approximately 60,000 parameters, which corresponds to 80 times smaller model compared to the original Xception. Whole architecture weights can be store in small 855 kilobytes file [4].

4.3.3 Training

During training was used square hinged loss 4.1. Last layer uses softmax activation function to output vector of probabilities. Gender network implements this for binary case and emotion network implements this for multiple classes. As a final prediction is picked maximal probability in output vector.

$$L(y, \hat{y}) = \sum_{i=0}^{N} (max(0, 1 - y_i \cdot \hat{y}_i)^2) \tag{4.1}$$

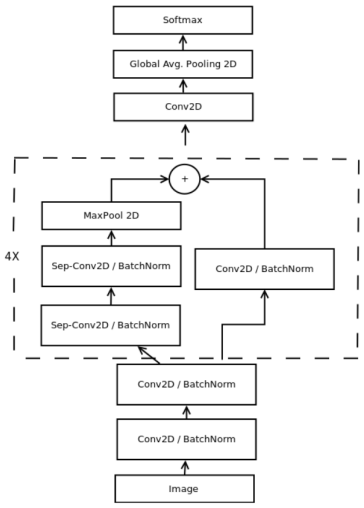


Fig. 4.1: The mini-Xception network architecture [4].

4.3.4 Inference

Authors evaluated gender variant on test data set of IMBD and it achieved 95% in gender classification task. Emotion variant obtained accuracy of 66% for the emotion classification task on the FER-2013 test data set. Authors pipeline using their own face detection had inference speed 0.22 ms.

5 Age classification

Age estimation is an intriguing problem because, it is field where humans themselves often cant deduce age of other person. But unlike OD task where main output is BB prediction trying to be identical clone of reasoning of human who labeled training data, here NN trains on actual values without possibility of interference from human error. This means that age estimation task should be problem where NNs solution will greatly outperform human capabilities. Current approaches estimate age directly as regression metric or treat age as ordinal property and use probability classification. Ordinal regression (OR) CNNs have been proven to outperform metric regression CNN [69].

5.1 Ordinal Regression in age estimation

In the field of machine learning OR has given a way to extend classification algorithms by reformulating problem to utilize multiple binary classification tasks. First attempts used perceptrons and Support Vector Machines [20][19]. Then came unifying general reduction framework from OR to binary classification [56]. This framework has later been used for CNN estimating age and had proven to be at state-of-art level [69]. It portrayed OR problem with K ranks as $K-1$ binary classification problems. Where k ($k \in (1, 2, .., K-1)$) task predicted whether age label of a face on image is greater than tasks rank r_k. All tasks share intermediate layers but have distinct weights in output layer. While successful at the time this approach had flaw in its classifier inconsistency. There was no assurance that tasks do not contradict each other. For example one task predicts age not above 20 and higher rank task predicts age above 30. Last algorithm in next section proposes solution to this flaw [69][12].

5.2 Current age estimation CNN algorithms

Modification of [69] called Ranking-CNN trains a series of CNNs and aggregates their output to predict age label of a given face image. This improves the predictive performance in comparison with a single CNN with multiple binary outputs. But there are downsizes in form of considerable increase in training complexity and no answer to classifier inconsistency [15].

Another approach utilizing binary classifiers for OR is the siamese CNN architecture. With only single output neuron, comparisons are made between the input image and multiple meticulously selected anchor images. From this comparisons is

computed outputted rank of age label, this is one of solutions to class inconsistency [75].

Example of non OR approach is All-in-one CNN, training for more general face attributes analysis tasks (face detection,gender prediction, age estimation, etc.) improves the overall performance of metric regression, by sharing lower-layer parameters [77].

Different non OR approach CNN with cascades was designed to classify face images into age groups followed by metric regression modules for more accurate age estimation [14].

Last mentioned method is OR Consistent Rank Logits (CORAL) framework, that will be thoroughly examined in next section, as it is currently holds 3 first places and one second place in various age data set challenges [106][12].

5.3 Consistent Rank Logits

CORAL model is CNN with ordinal responses. Its authors provide theoretical guarantees of class consistency, well defined generalization bounds, task-specific importance weighting, while accomplishing lower number of parameters to be trained in comparison to other state-of-art CNNs . Generalization bounds define how well model learns to generalize from the training data, otherwise said that model does not suffer overfitting. Task-specific importance weighting is used to solve problem of different sizes of class samples in data set (for example low number of samples for rare class like age label for 105 years old humans)[12].

5.3.1 Math support

Let training data set $D = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$ have N samples, where one sample is pair of $\mathbf{x}_i$ ($\mathbf{x}_i \in \mathcal{X}$) i-th image and its y_i rank ($y_i \in \mathcal{Y} = \{r_1, r_2, .., r_K\}$). Ranks are ordered $r_K \succ r_{K-1} \succ ... \succ r_2 \succ r_1$, symbol $\succ$ stand for ordering trend in ranks. Where $h(h : \mathcal{X} \to \mathcal{Y})$ is a ranking rule to be trained into CNN weights so loss function $L(h)$ is minimized to global minimum.

Next explanation will present unified ordinal reduction framework from [56]. This math groundwork will be important later for explanation of authors theorems of theoretical guarantees.

Let $\mathcal{C}$ be a KxK cost matrix where $\mathcal{C}_{y,r_k}$ is the cost of predicting sample $(\mathbf{x}, y)$ as rank r_k. There is no cost for $y = r_k$ 5.1. For OR is prefered to have $\mathcal{C}$ matrix rows V-shaped so $y = r_k$ case is at minimum spot of V 5.2. In OR tasks is cost defined as $C_{y,r_k} = |y - r_k|$.

$$C_{y,r_k} = \begin{cases} 0 & y = r_k \\ otherwise & y \neq r_k \end{cases} \quad (5.1) \qquad \begin{cases} C_{y,r_{k-1}} \geq C_{y,r_k} & y \leq r_k \\ C_{y,r_k} \leq C_{y,r_{k+1}} & y \geq r_k \end{cases} \quad (5.2)$$

Unified ordinal reduction framework extended OR problem to several binary classification problems. And that meant that cost matrix has to be convex in each row ($C_{y,r_{k+1}} - C_{y,r_k} \geq C_{y,r_k} - C_{y,r_{k-1}}$) for each y to become rank-monotonic model. This is not possible to put into practice, because each cost-related weighting of binary task is specific for each training sample [56]. But authors used math to get around it in CORAL [12].

5.3.2 CORAL OR label extension

Rank label y_i is extended into $K-1$ binary labels $y_i^{(1)}, ..., y_i^{K-1}$, where $y_i^{(k)} \in \{0, 1\}$ indicates if y_i exceeds rank r_k 5.3. This indication is for latter purposes, defined as indicator function $\mathbb{1}\{\cdot\}$ which if inner condition is true returns 1, otherwise returns 0.

$$y_i^{(k)} = \begin{cases} 1 & y_i > r_k \\ 0 & y_i \leq r_k \end{cases} \quad (5.3)$$

So instead of single ordinal age label, has CORAL anotation form of binary vector where ones start at index of original age label ordinal value 5.1. This allows for training simple CNN with $K-1$ binary classifiers in output layer. They share weight parameter but have independent bias which effectively deals with class inconsistency and lowers number of parameters.

$$h(\mathbf{x_i}) = r_q \quad (5.4) \qquad q = 1 + \sum_{k=1}^{K-1} f_k(\mathbf{x_i}) \quad (5.5)$$

From binary classifier tasks responses a predicted rank r_q is obtained in equation 5.4. And binary label vector index q is equal to one plus sum of predictions $f_k(x_i) \in \{0, 1\}$ where k is index of binary classifier in output layer 5.5. All predictions f_k must reflect ordinal information and at same time be rank-monotonic. Rank-monotonic rule for ordinal values can be simply explained for age case as moving one ordinal value to higher index must increase her actual non ordinal metric value and same must hold for moving in opposite direction 5.6 [12].

$$f_1(x_i) \geq f_2(x_i) \geq ... \geq f_{K-2}(x_i) \geq f_{K-1}(x_i) \quad (5.6)$$

5.3.3 Loss function

Let $\mathbf{W}$ denote NN weight parameters excluding the bias units of output layer. Let penultimate layer, whose output is denoted as $g(\mathbf{x}_i, \mathbf{W})$, shares a single weight with

all neurons of output layer. To $g(x_i, \mathbf{W})$ are then added $K-1$ independent bias units. Together $\{g(\mathbf{x}_i, \mathbf{W}) + b_k\}_{k=1}^{K-1}$ create inputs to the corresponding binary classifiers in the output layer. Then predicted empirical probability for binary classifier task k is defined in equation 5.7. Where s is logistic sigmoid function 5.8.

$$\hat{P}(y_i^{(k)} = 1) = s(g(\mathbf{x}_i, \mathbf{W}) + b_k) \quad (5.7) \qquad s(z) = \frac{1}{(1 + exp(-z))} \quad (5.8)$$

In training CORAL minimized loss function is weighted CE of $K-1$ binary classifiers 5.9. Where $\lambda^{(k)}$ denotes the weight of loss associated with k-th classifier . In further explanation $\lambda^{(k)}$ will be refered to as importance parameter for task k. For rank prediction are binary labels $f_k(\mathbf{x}_i)$ obtained from equation 5.10 [12].

$$L(\mathbf{W}, \mathbf{b}) = -\sum_{i=1}^{N}\sum_{k=1}^{K-1} \lambda^k [\log(s(g(\mathbf{x}_i, \mathbf{W}) + b_k)) y_i^{(k)} + \log(1 - s(g(\mathbf{x}_i, \mathbf{W}) + b_k))(1 - y_i^{(k)})] \quad (5.9)$$

$$f_k(\mathbf{x}_i) = \mathbb{1}\{\hat{P}(y_i^{(k)} = 1) > 0.5\} \quad (5.10)$$

5.3.4 Theoretical guarantees of classifier consistency

Ensuing theorem, says that by minimizing L (5.9), the learned bias units of the output layer are non increasing 5.11 and consequent predicted probability scores of $K-1$ tasks are decreasing for all i 5.12. This ensures classifier consistency. Authors proof of theorem 1 can be found in CORAL paper [12].

$$b_1 \geq b_2 \geq ... \geq b_{K-1} \quad (5.11)$$

$$\hat{P}(y_i^{(1)} = 1) \geq \hat{P}(y_i^{(2)} = 1) \geq ... \geq \hat{P}(y_i^{(K-1)} = 1) \quad (5.12)$$

Theorem 1 (ordered biases) *By minimizing loss function L* 5.9 *, optimal solution* $(\mathbf{W}^*, \mathbf{b}^*)$ *satisfies* $b_1^* \geq b_2^* \geq ... \geq b_{K-1}^*$.

Theorem 1 is free from V-shaped $\mathcal{C}$ row requirement 5.2, and this allows option to choose a fixed weight for each task that is independent from training examples. This greatly reduces the training complexity and allows simple uniform task weighting or taking data set imbalances into account, while guaranteeing 5.11 and 5.12 [12].

5.3.5 Generational bounds

New generalization binary classification bounds for CORAL model are derived that require only 5.1. Proof of theorem 2 can be found in CORAL paper [12].

Theorem 2 (reduction of generalization error) *Suppose* $\mathcal{C}$ *cost matrix with* 5.1*, and* P *is underlying distribution of* $(\mathbf{x}, y)$*. If binary classification rules* $\{f_k\}_{k=1}^{K-1}$ *obtained by optimizing* L 5.9 *are rank-monotonic then* 5.13. (E is generalization error.)

$$\underset{(x,y)\sim P}{E} \mathcal{C}_{y,h(x)} \leq \sum_{k=1}^{K-1} |\mathcal{C}_{y,r_k} - \mathcal{C}_{y,r_{k+1}}| \underset{(x,y)\sim P}{E} \mathbb{1}\{f_k(x) \neq y^{(k)}\} \tag{5.13}$$

5.3.6 Task-specific importance weighting

Authors proposed task-weighting schema defined as follows. Let $S_k = \sum_{i=i}^{N} \mathbb{1}\{y_i^{(k)} = 1\}$ be sum of samples whose ranks y exceed r_k. This with rank ordering in mind means that for these sums must 5.14. Now let $M_k = \max(S_k, N - S_k)$ be the number of majority binary label for each task. From this authors define importance of the k-th task as the scaled $\sqrt{M_k}$ 5.15. To remind importance parameter $\lambda^{(k)}$ is used in 5.9.

$$S_1 \geq S_2 \geq ... \geq S_{K-1} \tag{5.14}$$

$$\lambda^{(k)} = \frac{\sqrt{M_k}}{\max\limits_{1 \leq i \leq K-1}(\sqrt{M_i})} \tag{5.15}$$

Thanks to this weighting scheme general class imbalance in data set is taken into account. It should be noted that task-importance weighting is only used for training CORAL model, when computing 5.4 predicted rank by adding binary results each task has the same influence on final rank prediction [12].

5.3.7 Training data

Coral was trained and tested individually for 4 age estimation data sets. Specifically MORPH-2 data set [81], CACD database [13], UTKface database [91], AFAD database [69]. First two had to have preprocessed images so faces spanned the whole image with the nose tip being in the center. Latter two have face in image in such form as default. Then for each data set was divided into 80 % training data and 20% test data. All images were resized to 128x128x3 pixels and then randomly cropped to 120x120x3pixels to augment the model training [12].

5.3.8 Architecture

Authors used ResNet-34 classification 5.1 network architecture where they replaced output cross-entropy layer with CORAL model binary classification tasks.

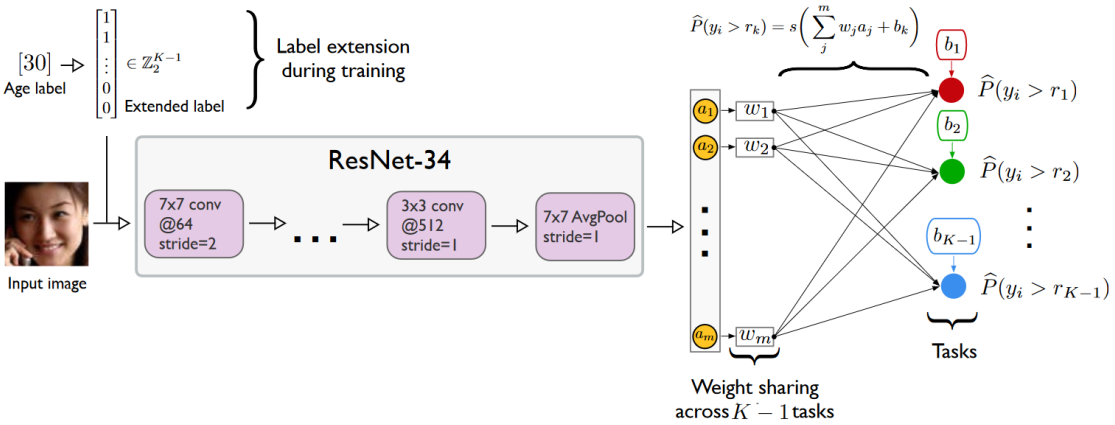


Fig. 5.1: CORAL model architecture [12]

5.3.9 Training

Authors trained CORAL model, original OR-CNN model [69] and unchanged ResNet-34 [39] multi class entropy model 3 times. Each time for different random seed weight initialization for fair comparison between methods. They were trained for 200 epochs with stochastic gradient descent via adaptive moment estimation using exponential decay rates $\beta_0 = 0.9$ and $\beta_2 = 0.99$. Authors found that all models performed best with learning rate 0.00005. For all models, the loss converged after 200 epochs.

All loss functions and neural network models were implemented in PyTorch 1.1.0 and trained on NVIDIA GeForce1080Ti and Titan V graphics cards [12].

5.3.10 Inference

To evaluate and compare trained models, authors used mean absolute error (MAE) 5.16 and root mean squared error (RMSE) 5.17 metrics. Where y_i is GT rank of i-th sample and $h(\mathbf{x_i})$ is corresponding predicted rank. As can be seen from table 5.1 has lowest errors and at current time this holds the record for 3 of these data sets and second place for remaining data set [12].

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - h(\mathbf{x_i})| \qquad (5.16)$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - h(\mathbf{x_i}))^2} \qquad (5.17)$$

Tab. 5.1: Age prediction MAE and RMSE on the data sets test sets without task importance weighting. [12]

Method	Random Seed	MORPH-2		AFAD		UTKFace		CACD	
		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
CE-CNN	0	3.40	4.88	3.98	5.55	6.57	9.16	6.18	8.86
	1	3.39	4.87	4.00	5.57	6.24	8.69	6.10	8.79
	2	3.37	4.87	3.96	5.50	6.29	8.78	6.13	8.87
	AVG ± SD	3.39 ± 0.02	4.89 ± 0.01	3.98 ± 0.02	5.54 ± 0.04	6.37 ± 0.18	8.88 ± 0.25	6.14 ± 0.04	8.84 ± 0.04
OR-CNN	0	2.98	4.26	3.66	5.10	5.71	8.11	5.53	7.91
	1	2.98	4.26	3.69	5.13	5.80	8.12	5.53	7.98
	2	2.96	4.20	3.68	5.14	5.71	8.11	5.49	7.89
	AVG ± SD	2.97 ± 0.01	4.24 ± 0.03	3.68 ± 0.02	5.13 ± 0.02	5.74 ± 0.05	8.08 ± 0.06	5.52 ± 0.02	7.93 ± 0.05
CORAL-CNN	0	2.68	3.75	3.49	4.82	5.46	7.61	5.56	7.80
	1	2.63	3.66	3.46	4.83	5.46	7.63	5.37	7.64
	2	2.61	3.64	3.52	4.91	5.48	7.63	5.25	7.53
	AVG ± SD	**2.64 ± 0.04**	**3.68 ± 0.06**	**3.49 ± 0.03**	**4.85 ± 0.05**	**5.47 ± 0.01**	**7.62 ± 0.01**	**5.39 ± 0.16**	**7.66 ± 0.14**

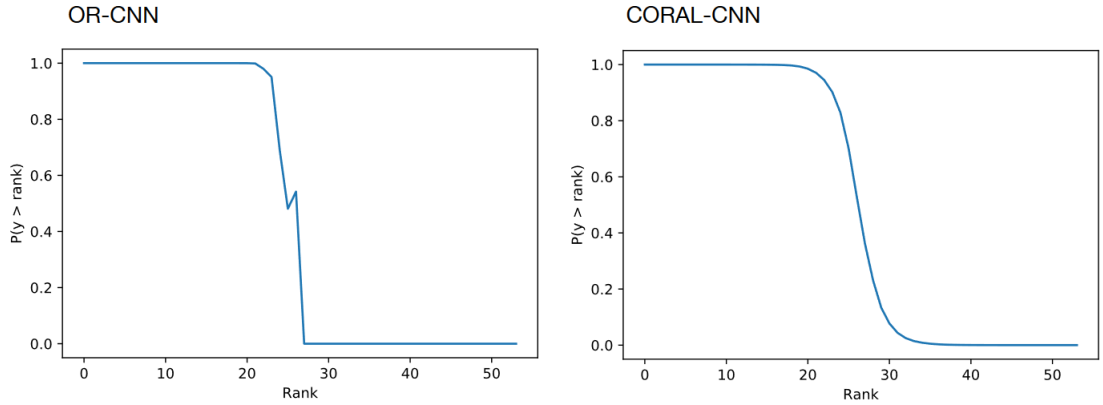


Fig. 5.2: Graphs of the predicted probabilities for each binary classifier task on one sample from the MORPH-2 data set by OR-CNN and CORAL-CNN. Classic OR-CNN has an inconsistency at rank 26. The CORAL-CNN is class consistent with rank prediction being cumulative distribution function. [12]

Last metric is Cumulative score (CS) 5.18, computed as the proportion of images for which the absolute differences between the predicted rank and GT rank are below a threshold value T. In sense its similar to recall-IoU curve metric. Averaged comparison of CS metric from all runs shows clearly CORAL model dominance 5.3. And from graph in figure 5.2, it can be seen that CORAL model has solved class inconsistency problem [12].

$$\mathrm{CS}(T) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\{(|y_i - h(\mathbf{x_i})| \leq T\} \tag{5.18}$$

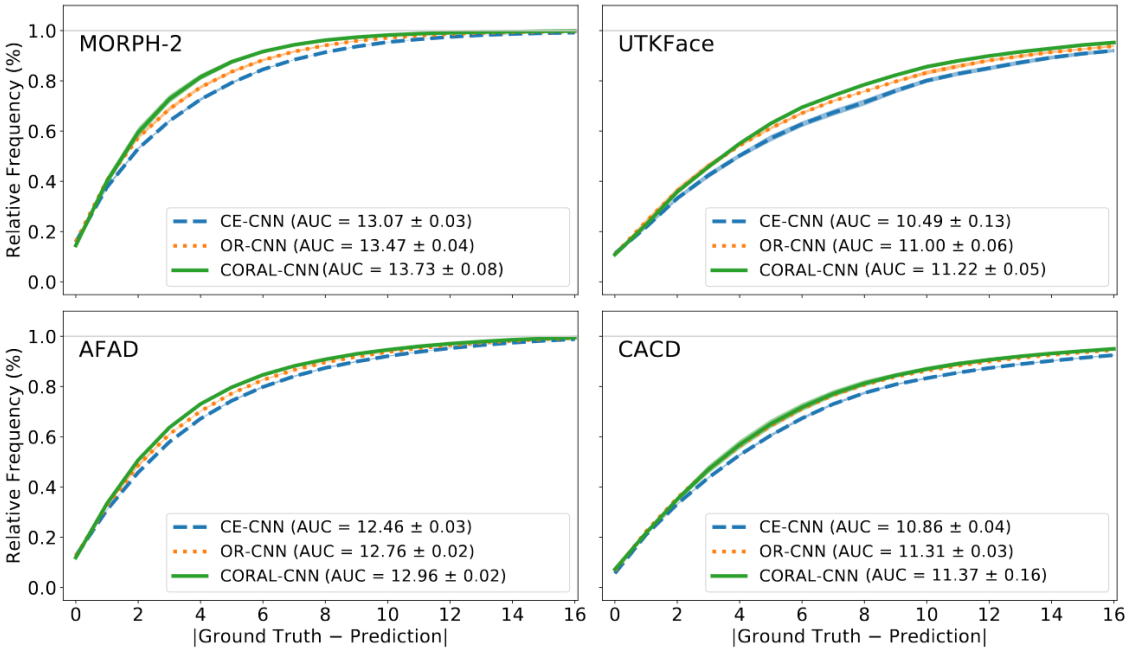


Fig. 5.3: Averaged comparison from three runs of age prediction models on 4 data sets, without task importance weighting [12].

5.4 Proposed enhancement of CORAL with LSTM elements

As age estimation is the most difficult task with relatively high offset from ground truth prediction, this task CNN architecture was picked to be experimentally improved by introducing LSTM cell into architecture.

5.4.1 Proposed CORAL-LSTM architecture

Architecture for age estimation CORAL extracts features from face image with ResNet-34, then concatenates features into 1D feature vector which is passed to final linear layer consisting of binary classification tasks. By inserting LSTM layers to process 1D feature vector, CORAL can learn to utilize temporal information to improve overall accuracy. Therefore network has to work with time sequence of face images, which after feature extraction concatenate their feature vectors into one feature vector which is then passed LSTM layers. These layers then process feature vector into new feature vector with length that final classification layer expected. This way can be LSTM effect on CORAL performance properly assessed.

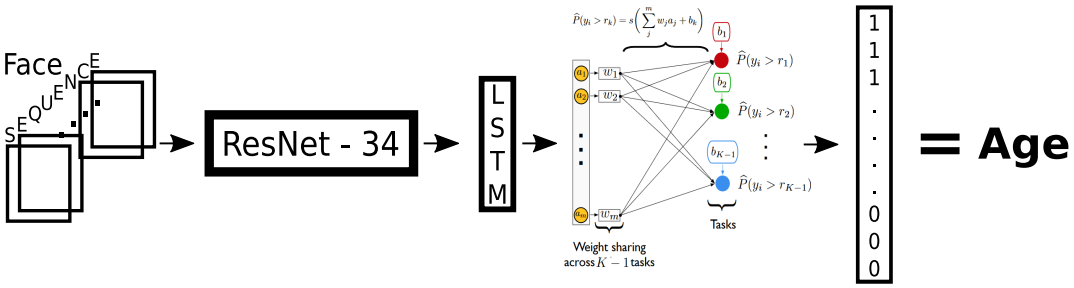


Fig. 5.4: Proposed CORAL-LSTM architecture

5.4.2 Training and test data requirements

For training and testing LSTM layers new video face sequences have to be created. Training data set and test data set has to have all representation and variety of age, gender, skin color, facial expressions and poses to achieve CORAL learning only general useful features. This means that there should not be many sequences from same subject and if so, then they should not be sequences where subject has same age or expression. Test and training data set should not share subjects, to achieve objective evaluation.

5.4.3 Training

To properly observe impact of LSTM addition on CORAL age estimation performance, CORAL will be trained from start on UTK data set, then compared in performance with original CORAL authors weights performance. This CORAL UTK weights will be then saved. On these weights various LSTM modifications will be inserted to be fine-trained on data set described by previous subsection. These modifications include testing optimal number of LSTM layers to insert, data augmentation of input sequences and various settings of freezing original CORAL layers.

5.4.4 Inference

Each trained model will have its performance evaluated with RMSE and MSE metric. Comparisons will be made with aim to find optimal LSTM modification and compare its performance against saved copy of unmodified CORAL weights. Since its important to retain reasonable inference speed for live inference, inference speed increases for each tried setting of number of LSTM layers will be measured.

6 Data sets and Frameworks

Data and framework are essential to development of NN designs. Data set benchmarks are used to train and correctly assess quality of designed NN architecture across growing field of general or specified OD tasks. Neural network frameworks, not to be confused with specific NN architecture, are environments to build, train and test NN, their biggest boon is in allowing researchers to focus on NN itself rather then programming problems of development. Data sets in general can not be easily categorized because they does not have same number of anotated attributes. Next section face data sets have all BB anotation, while in latter sections age estimation data sets have always age anotation and emotion data sets have always emotion anotation. Gender anotation is included most of the time in these data sets and does not have its own section.

6.1 Facial detection data sets

There are many face data sets in use and their size and properties have varying degrees. To make proper choice one should take in to account task given for example pose, lightning, expression etc. or depending property tested to gain insight about algorithm behavior.

6.1.1 Wider face data set

Publicly available Wider Face data set has 32.203 images with labeled 393 703 faces with a high degree of variability in scale, pose and occlusion. Wider Face data set is organized based on 61 event classes. For each event class, authors randomly selected 40%/10%/50% data as training, validation and testing sets. Testing sets do not have GT BB anotations released, so users are required to submit final prediction files for evaluation [96].

6.1.2 FDDB: A Benchmark for Face Detection in Unconstrained Settings

Face Detection Data Set and Benchmark (FDDB) is a data set created by labeling faces on Faces in the Wild data set [5]. It was designed for studying the problem of unconstrained face detection. This data set contains the annotations for 5171 faces in a set of 2845 images [45].

6.1.3 LFW: Labeled faces in wild data set

As with FDDB, LFW is too a data set created by labeling Faces in the Wild data set [5]. With same purpose for researchers to study the problem of unconstrained face recognition. The data set contains the annotations for 5749 subjects in a set of 13233 images. Each face has been labeled with the name of the person pictured. From them 1680 of the people pictured have two or more distinct photos in the data set. Data set has a major flaw in that the faces were detected by the Viola-Jones face detector, not hand-labeled. Authors discourage other than for unconstrained face recognition because LFW data set has mostly adult male non ethnic faces with lack of data for extreme conditions of lighting,pose or occlusions [43][44][54].

6.1.4 VGGFace2 data set

Visual geometry group VGGFace2 is a large-scale face data set. Images were acquired from Google Image Search site and have large variations in pose, age, lightning, ethnicity and profession. The data set contains the annotations for over 9000 subjects in a set with over 3.3 million images. Subject face is on average in 362 images. Since data set is product of image search browser, its contents does not in general have in the wild property. But its size alone is reason why it is one the most used data sets for facial detection training [11].

6.1.5 CASIA Webface data set

Chinese Academy of Science Institute of Automation released a large-scale face data set with goal of standardizing evaluation protocol and reproducible research. Source of images is Internet Movie database (IMDb) which structure allowed authors to create data set with over 10575 subjects in nearly half million images. Same as VGGFace2 data set, CASIA Webface data set does not have pure in the wild property [97].

6.1.6 CelebA data set

Another large-scale face data set this time originating from the Chinese University of Hong Kong Multimedia Laboratory. CelebFaces Attributes Data set (CelebA) is data set with more than 200000 celebrity images, each with 40 attribute annotations. The images have great diversity in pose variations and background with large size and rich annotations [62].

6.1.7 SCface - Surveillance Cameras Face Database

Is a face database of human faces released by University of Zagreb. Images were taken in uncontrolled indoor environment using five video surveillance cameras of various qualities. The data set contains the annotations of 130 subjects in a set with 4160 images. Various quality of images should mimic real-world conditions and enable robust face recognition algorithms testing, emphasizing different law enforcement and surveillance use case scenarios. While data set offers great quality for law enforcement facial detection tests, its small size and low diversity in gender and ethnicity are some of its shortcomings, making it unfit for training and general facial detection of different ethnics [37].

6.2 Age estimation data sets

Data sets in this section are similar to previous section. But unlike facial data sets, they do not have to necessarily have to have BB anotation, but must have age anotation. These data sets also have most of the time gender anotation.

6.2.1 MORPH dat aset

One of the top benchmarks for age estimation in computer vision, the MORPH data set is made up from 55134 mugshots with longitudinal spans, taken in between years 2003 and late 2007. There are large pose, lighting and expression variations alone with occlusion in this database. Each image has anotation consisting of subject ID number, picture number,date of birth, date of arrest, race, gender, age, time since last arrest, and image filename. Individuals that were arrested multiple times over this five year span means quality data for age estimation. On average there are 4 images per subject. For its size,diversity and longitudinal span is MORPH true benchmark of age estimation [81].

6.2.2 FG-NET data set

Facial and gesture recognition network paper released in 2004, same year as original MORPH, a data set containing 400 color face images of 40 subjects where each individual in the database supplied a collection of photographs taken over many years in three-year intervals on average. The age of subjects, range between newborns up to 35 years. Number of images per subject greatly varies with mean around 10 images. Database was split into two parts, used independently for training and testing. Subject can occur only in one part of data set [51].

6.2.3 Adience data set

Collection of images in this data set is intended to be as true as possible to the challenges of real-world imaging conditions. It attempts to capture all the variations in appearance, noise, pose, lighting with prospect of capturing face of subject without careful preparation or posing. There is total number of 2284 subjects in 26580 images. Its anotation includes age groups and gender label. Data set source of images are Flickr albums, acquired from automatic upload by smartphone devices [27].

6.2.4 CACD data set

Cross-Age Celebrity Data set (CACD) is a large-scale data set for face recognition and retrieval across age. The data set contains more than 160,000 images of 2,000 subjects with age ranging from 16 to 62. The images are sourced from search engines using celebrity name and year (2004-2013) as keywords. CACD is by far one of the largest publicly available cross-age face data sets [13].

6.2.5 AFAD data set

The Asian Face Age Data set (AFAD) is a new data set proposed for evaluating the performance of age estimation. This data set is oriented to age estimation on Asian faces. Contains more than 160K facial images and the corresponding age and gender labels. The AFAD data set is built by collecting selfie photos on a particular social network RenRen Social Network. The network is widely used by Asian students and graduates and this makes data set age range 15-years to more than 40-years old [69].

6.2.6 UTKface data set

UTKFace data set is a large-scale face data set with long age span (range from 0 to 116 years old). The images cover large variation in pose, facial expression, lightning, occlusion, resolution. The data set contains over 20,000 single face images with annotated age, gender, and ethnicity. This data set can be used on a variety of tasks from face detection, age estimation, age progression/regression, landmark localization, etc. [91].

6.2.7 IMDB data set

IMDB-WIKI dataset, the largest public dataset of face images with age and gender labels. Large dataset of 100,000 actors as listed on the IMDb website. With metadata like date of birth, name and gender. Subject can have multiple images.

Each image having been assigned biological age at time of photo. There are in total 523,051 face images [82].

6.2.8 Large Age-Gap Face Verification data set

The Large Age-Gap (LAG) data set data set containing images ranging from child/young to adult/old. The data set contains 3,828 images of 1,010 subjects. For each identity at least one child/young image and one adult/old image are present [7].

6.2.9 Specs on Faces data set

Is a collection of face images with subjects wearing glasses and its focused on two challenges harsh illumination environments and face occlusions, which highly affect face detection, recognition, and classification. Data set contains 42,592 images for 112 subjects (66 males and 46 females). The glasses are the common natural occlusion in all images of the data set. There are also two more synthetic occlusions (nose and mouth) added to each image. Each image got augmented by three image filters, that may increase chance of evading facial recognition systems. Anotation consist of the subject ID, facial landmarks, face and glasses rectangles, gender and age labels, year that the photo was taken, facial emotion, glasses type etc. . Three levels of difficulty (easy, medium, and hard) are used to categorize images [2].

6.2.10 EURECOM Visible and Thermal paired Face database

Reduced number of databases is acquired in thermal spectrum and that limits exploration of human face temperature as attribute for facial recognition tasks. This database of face images was acquired simultaneously in visible and thermal spectra under various variations: lightning, expression, pose and occlusion. Collected from 50 subjects of different age, sex and ethnicity, resulting in total of 2100 images. Anotation includes date of acquisition,age,gender and if the subject agreed that his/her face images can displayed in scientific articles/presentations [66].

6.3 Emotion data sets

Emotion data sets contain images containing persons face with a specific facial expression. Each data set has its emotion categorized into groups covered in section 4.2. Facial emotion images can captured under different angles. And some data sets are made up from video sequences.

6.3.1 CK+ data set

The Extended Cohn-Kanade Dataset (CK+) is made of 593 video sequences on both posed and non-posed emotions. The age range of its 123 subjects is from 18 to 30 years, most of whom are female. Image sequences may be analyzed for both AU and basic emotion group [63].

6.3.2 CE data set

Compound Emotion (CE) data set contains 5060 images corresponding to 22 categories of basic and compound emotions for its 230 human subjects (130 females and 100 males, mean age of 23). Most ethnicities and races are included, including Caucasian, Asian, African, and Hispanic. Facial occlusions are minimized, with no glasses or facial hair [25].

6.3.3 FER - 2013 data set

The Facial Emotion Recognition 2013 data set is a data set released public as a Kaggle challenge. Its data consists of small grayscale images of faces. The faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. Challenge task was to categorize images to emotions of basic emotion group [88].

6.3.4 DISFA data set

Denver Intensity of Spontaneous Facial Action Database (DISFA) consists of 130,000 stereo video frames at high resolution of 27 adult subjects (12 females and 15 males) with different ethnicities. The intensities of the AUs for all video frames were manually scored using two human experts in FACS. The database also includes 66 facial landmark points for each image in the database [67].

6.3.5 MMI data set

MMI Facial Expression Database consists of over 2900 video sequences and high-resolution still images of 75 male and female subjects. It is fully annotated for the presence of AUs in the video sequences, indicating for each frame whether an AU is in a neutral, onset, apex, or offset phase [74].

6.3.6 BU-3DFE data set

Binghamton University 3D Facial Expression (BU-3DFE) data set contains a total of 100 subjects, 56 females and 44 males, displaying emotions of basic emotion group. There are 25 3D facial emotion models per subject in the database, and a set of 83 manually annotated facial landmarks associated with each subject [98].

6.3.7 JAFFE data set

Japanese Female Facial Expressions (JAFFE) data set contains 213 images with emotions of basic emotion group, posed by ten different female Japanese subjects [65].

6.3.8 BP4D-Spontanous data set

Binghamton-Pittsburgh 3D Dynamic Spontaneous (BP4D-Spontanous) is a 3D video database that includes a diverse group of 41 young adults (23 women, 18 men) with spontaneous facial expressions. The subjects are in 18–29 years age range. Eleven of them are Asian, six are African-American, four are Hispanic, and 20 are Caucasian. The facial features were tracked in the 2D and 3D domains [90].

6.3.9 KDEF data set

The Karolinska Directed Emotional Face (KDEF) data set contains 4900 images of human emotional facial expressions. Data set has 70 subjects, each displaying seven different emotional expressions captured from five different angles [64].

6.4 Popular frameworks

For easy research and development purposes neural networks frameworks offer simplicity and intuitively abstract NN architecture design. Which is important if one is to correctly reproduce and experiment in the field of deep learning. Most popular frameworks of today were created by big data information giants of current world, hoping to utilize their vast amounts of data.

6.4.1 Tensorflow

TensorFlow is an end-to-end general purpose high-performance computing library written in C/C++ as a Python API. Provides high-performance APIs for building NN. It has a comprehensive, flexible ecosystem of tools, to let researchers push the

state-of-the-art in ML and developers easily build and deploy ML-powered applications. Originally developed in 2015 by Google Brain team within Google's Machine Intelligence Research organization to replace previous Google ML library Theano. Theano was inflexible because of the static computation graphs and this imperfection passed on to Tensorflow library. But Tensorflow 1.X is currently undergoing big revision to Tensorflow 2.X aimed to solve this flaw. TensorFlow provides stable Python and C++ APIs, as well as non-guaranteed backward compatible API for other languages [1] [87].

6.4.2 Keras

Keras is a high-level neural networks Python API capable of running on top of TensorFlow, CNTK, or Theano. Allows for easy and fast prototyping thanks to its user friendliness, modularity, and extensibility. Its development started when Pytorch framework rolled out, with goal of creating similar lightweight intuitive framework. Main prospect was to be able go from idea to result in least amount of time is key to doing good research. It is now default high level API for TensorFlow 2.X [17][87].

6.4.3 Pytorch

PyTorch is a Python clone of the framework of Torch framework, which was originally written in computationally potent Lua language. Since Lua was for most users unfamiliar language it was decided to rewrite it as Python API. Its creators are Facebook AI labs and it was released in 2007. It had major advantage against Tensorflow that it had dynamic computation graphs. That are valuable for situations where you cannot determine the computation, like recursive computations that are based on variable data. Two main parts of PyTorch are tensor computing with strong acceleration via graphics processing units and that its NN are build on reverse automatic differentiation system Autograd. Pytorch also has a C++ interface [72][73].

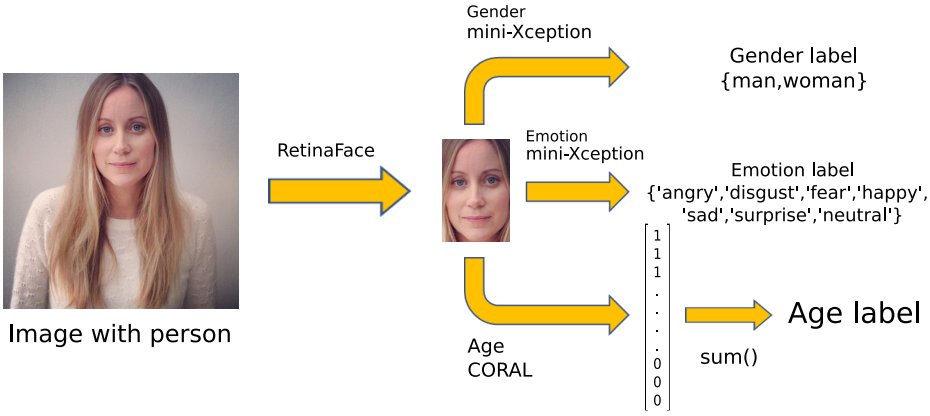


Fig. 7.1: Simplified scheme of automated human recognition solution pipeline

7 Implementation

This chapter covers prerequisites and actions needed for implementation of human recognition solution. This solution uses combination of CNNs to detect face and subsequently analyze from face gender, emotion and age of detected person. These CNNs were covered in chapters 3,4 and 5.

The following sections describe creation of new video data set which is used for training and testing modified age classification network. This data set will be for easier orientation in text named Age Video Sequence data set (AVS). Modification to age classification network was covered in subsection 5.4.

All following methods are implemented in programming language Python 3.7.4 and used NN frameworks Pytorch 1.3.1 and Keras 2.3.1 wrapper for TensorFlow 2.1 framework [72][17][1]. Training and testing is done on Nvidia GPU Geforce GTX 1060, for CPU tests was used AMD Ryzen 5 2600 Six-Core 3.4 GHz Processor. Software dependencies are in dependencies.txt file which is in project folder in thesis attachments.

7.1 Automated human recognition pipeline

Solution is implemented in script "Recognition_net.py" that takes as input directory of images to be analyzed and directory to save analyzed images with prediction metadata. Other optional inputs are RetinaFace base architecture and weights (default ResNet-50 or MobileNet-0.25), CORAL weights training data set (AFAD [69], CACD[13], MORPH[81], UTK[91], default 1-100 UTK (trained in this thesis)) and processing unit (default CPU, GPU).

7.1.1 Initialization

Script contains functions and classes from authors of individual CNNs [23][4][12], which are used to initialize network model and load weights. Script starts with reading user inputs and based on user input initializes each CNN.

7.1.2 Face detection

Script start loading images from input data directory, each image is processed same way as in RetinaFace paper [23]. Image is then passed to RetinaFace network which returns bounding boxes in form of top left and bottom right coordinates and their probability scores. Third output are parametrized landmarks that are decoded into image coordinates from image size. Boxes with low probability are rejected and overlapping boxes are united using non-maximum suppression.

7.1.3 Facial analysis

Information about top left and bottom right corner is used to crop raw image. Cropped image of face is resized into 128x128 and centered. In this form is image passed to CORAL model which returns vector of age probabilities, this vector is thresholded by 0.5 (5.10) and then is summed into estimated age label. Bottom age limit of picked data set of CORAL weights is added to estimated age label. This is the final age prediction.

Next bounding box location is used again to crop out face, but this time is used area larger than detected bounding box. It takes 50% wider and 66% longer face area than original bounding box. This cropped face image is converted to gray scale. Gray face image is normalized and has two copies created. First is input for gender network and is resized into 64x64 size. Second is for emotion network and is resized into 48x48 size. Both networks output a vector of label probabilities where prediction is label with maximum probability.

7.1.4 Saving predictions

All recognized faces with their predictions are drawn onto raw image and saved to chosen save directory. Along with analyzed image the following data is stored in .txt file with same name: recognized faces bounding boxes metadata with predicted gender, emotion and age.

7.1.5 Individual CNN task evaluation

RetinaFace face detection is tested with both ResNet-50 and MobileNet-0.25 architectures on data set Wider Face[**?**]. Evaluation uses code from RetinaFace authors[23] and produces output of evaluation in form of AP metric.

Evaluation inference script for Wider Face is test_widerface.py, it has a great variate of inputs for customization. For example inputs to set different paths to weights, results, datasets, or to set different thresholds for NMS,box confidence. There is also option to choose run on CPU or GPU. All possible inputs have default values. Main output is folder of event folders each containing .txt file with predicted boxes for each image. As backbone network was left default value - Resnet50. To have individual BB visualized into image and saved for purposes of results, there needs to be entered optional input –save_image.

To evaluate folder with .txt files, there is need to run script setup.py for input of type build_ext with value –inplace. This setups something from Cython package, to be used by evaluation script. Finally evaluation can be done by running evaluation.py script, with no input needed. This script reads prediction and computes Easy, Medium and Hard AP metrics for Wider Face validation dataset.

Next gender classification network mini-Xception was tested on IMBD data set [**?**]. Evaluation code is implemented in script "test_IMBD.py" by printing evaluated accuracy. Because IMBD data set did not have cropped faces, to crop faces was used RetinaFace MobileNet as face detector.

Following with emotion classification mini-Xception network tested on FER-2013 test data set [88]. Evaluation code is implemented in script "test_FER.py" and prints out accuracy and confusion matrix which is also ploted ans saved as image.

Age estimation network CORAL was tested on UTK data set. Tests were evaluated using pretrained weights from CORAL authors and 1-100 UTK weights. All weights were trained with different age ranges AFAD(15-40), CACD(14-62), MORPH(16-70), UTK(21-60), so to evaluate their performance objectively, evaluation is done twice. First for images with persons in data sets age range intersection (21-40) and second for whole UTK. Evaluation code is implemented in script "test_UTK.py" and outputs MSE and RMSE metrics.

7.2 Age Video Sequence data set

LSTM layers require as input time sequence of features concatenated into 1D vector. Length of time sequence was empirically chosen as 20 frames. Higher number of frames could increase LSTM effectiveness but would lead to longer training and

most importantly longer inference times which should be avoided. AVS data set requirements are covered in subsection 5.4.2.

7.2.1 Origin of data

Videos of subjects with age range 1-100 were downloaded from Youtube channel SoulPancake [101]. Each video had around 30 or more unique individuals with their age displayed in bottom left corner. Each subject has at least one continuous scene. Subject scene duration have different times but mostly around 2 to 8 seconds, since video has 25 FPS, this means that only 0.8 second long scene is needed to create one time sequence. Subjects gender, emotion, skin tone and age greatly vary therefore are considered appropriate for AVS data set.

7.2.2 Processing data

Each person had their continuous scene extracted and divided into 20 frame sequences, with frames saved in jpg format. Each frame has its scene index, sequence index, frame index and age encoded into filename and additionally saved into csv file. If last sequence created from scene did not had exactly 20 frames then created images were erased. This processing is done by script "read_sequence.py" .

7.2.3 Organizing data

AVS training data set is made from 166 scenes from which were created 808 sequences. AVS testing data set has only 35 sequences but each with unique individual. Each data set has its own csv file. Only about 20% of subjects from test data set is in training data set. In these few cases where subject is present in both data sets, there is age difference of around 3 to 4 years.

7.3 CORAL-LSTM implementation

New modified CORAL architecture is designed with variable number of LSTM layers. Then original CORAL architecture is trained on UTK data set [91], but unlike CORAL authors, here is used whole age range of UTK data set (1-100). To avoid confusion and for easier referencing CORAL weights trained in this thesis on UTK data set will be refered as 1-100 UTK weights. This trained model has freezed layer and is used as start point to try out inserting LSTM layers and unfreezing old layers to find best training setting to fine-train CORAL-LSTM to get best results from AVS test data set. Fine-training is done on AVS training data set. Weights with best results are used in "Recognition_net_LSTM.py", which is script with

same functionality as "Recognition_net.py" but age classification part is changed. It waits until 20 images have face detected to create face time sequence, which is then used as input for CORAL-LSTM. After inference is oldest face image discarded and new face image is added to time sequence.

To properly observe impact of LSTM addition on CORAL age estimation performance, CORAL will be trained from start on UTK data set, then compared in performance with original CORAL authors weights performance. This CORAL UTK weights will be then saved. On these weights various LSTM modifications will be inserted to be fine-trained on AVS data set. These modifications include testing optimal number of LSTM layers to insert, data augmentation of input sequences and various settings of freezing original CORAL layers.

7.3.1 CORAL-LSTM architecture

In CORAL architecture is input processed by feature extractor ResNet-34 into 2D feature output which is concatenated into 1D feature vector. This is done for single face image, since torch uses batch of inputs, all that is needed to create 1D time feature sequence vector is to concatenate whole batch. This batch has to have 20 inputs and correct order of inputs inside batch. After that sequence is passed to LSTM layers which output feature vector of same length that last layer expected in original architecture.

7.3.2 Training unmodified CORAL weights on 1-100 UTK data set

Since UTK training data set images are not just faces, but have often more than half of body of subject, faces need to be detected. Script "read_UTK.py" reads all images, gets their label from their filename and detects face in them. Image filename with age label and detection bounding box parameters are saved into csv file. Both training and test data set have their metadata csv file from "read_UTK.py", then with script "train_CORAL.py" are trained new weights. Script takes as input paths to directories with UTK training and test data set and their metadata csv files. Training used optimizer Adam, learning rate 0.0005 and batch size 20 shuffled inputs. Inputs used as online augmentation random image flip around vertical axis. Unique to CORAL architecture weight importance is used to increase loss for rare age groups. It is computed from train csv file, and used to influence loss.

7.3.3 Training CORAL-LSTM

Script "train_LSTM.py" takes input paths to directories with AVS training and test data set and their metadata csv files. It takes newly trained 1-100 UTK weights and loads them into CORAL-LSTM architecture. These weights are freezed, and pnly LSTM layers are allowed to train. Same as with original CORAL training, script uses optimizer Adam, learning rate 0.0005, weight importance and batch size 20. But here batch size is used as a way for network to process face images into 1D vector individually and then concatenate them into sequence. So batch size 20, since sequence is 20 frames long, is mandatory setting. Input images in batch therefore can not be shuffled. But shuffling is done by creating new order of sequences at end of each epoch, which does not compromise time continuity inside frames of face time sequences. Online augmentation flip around vertical axis is implemented at start of sequence with respect for all following sequence frames to be or not to be fliped uniformly.

7.3.4 LSTM impact evaluation

New CORAL-LSTM weights are trained for 1,2 and 3 layers. LSTM layers propagation time for each setting is measured. Best setting is picked and is tried to be trained with unfreezing more than just LSTM layers. And that includes option of unfreezing all layers or unfreezing last classification layer. During training there are multiple times saved model weights, after passing of certain number of epochs. All new trained weights, including 1-100 UTK weights are evaluated AVS test data set for comparison. Evaluation is done by script "test_VideoDataset.py".

8 Results

This chapter consist of various evaluations and outputs described by chapter 7. First is evaluation of recognition solution and its output examples. Second is summary description with tables, images and histograms from new AVS data set. Third part shows evaluated weights on AVS test data set from all tried CORAL-LSTM training settings and weights from original CORAL authors.

8.1 Automated human recognition solution

Average inference time for each used CNN is evaluated on tables 8.1 and 8.2, first table was tested on images with 1920x1080 resolution and second table on images with 1280x720 resolution. Time of full analysis of single image by solution is in tables referred as one image inference. For this inference solution used for face detection RetinaFace architecture based on MobileNet-0.25 . Times are averaged over 1000 image inferences, because GPU betters its resource allocation in time. Images contained only one person. On figure 8.1 can be seen output examples, bottom part is single image. Gender prediction is highlighted by BB color (woman-red,man-blue).

Tab. 8.1: Table of CNN average inference speed for each used network and combined solution, tested on images with 1920x1080 resolution

CNN network	CPU inference	GPU inference
RetinaFace ResNet-50	28.8152s	0.0410s
RetinaFace MobileNet-0.25	1.8344s	0.0222s
CORAL network	0.1895s	0.0145s
Gender network	0.0080s	0.0045s
Emotion network	0.0129s	0.0078s
One image inference	2.4502s	0.3392s

Tab. 8.2: Table of CNN average inference speed for each used network and combined solution, tested on images with 1280x720 resolution

CNN network	CPU inference	GPU inference
RetinaFace ResNet-50	10.6836s	0.0380s
RetinaFace MobileNet-0.25	0.6755s	0.0224s
CORAL network	0.1452s	0.0146s
Gender network	0.0058s	0.0044s
Emotion network	0.0100s	0.0084s
One image inference	0.9871s	0.1828s

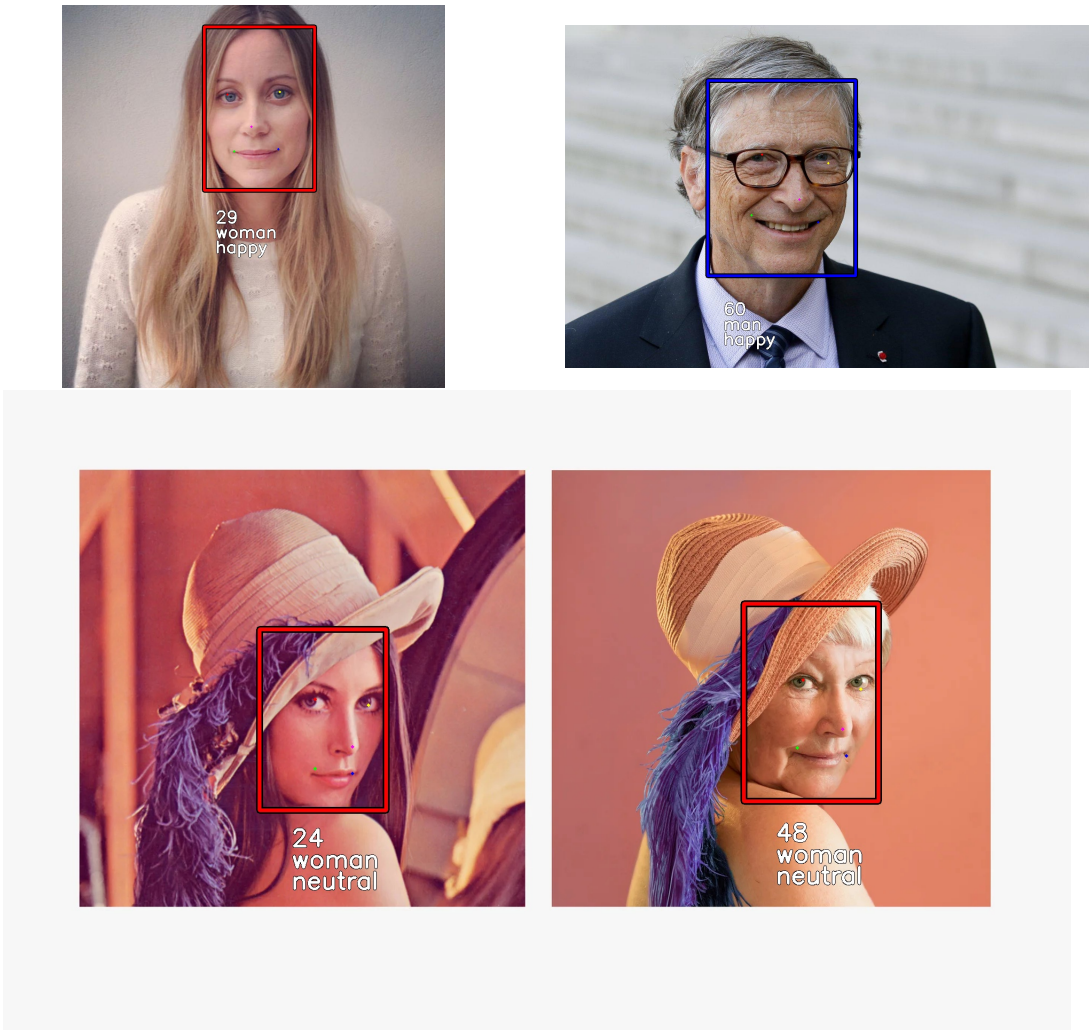


Fig. 8.1: Example of recognition solution output

8.1.1 Individual task evaluation

Table 8.3 contains evaluation of RetinaFace on Wider face validation data set[96]. This data set consist of three parts, Easy, Medium and Hard. Gender classification network mini-Xception had 85.33% accuracy on IMDB data set [82]. Emotion classification network mini-Xception had 65.52% accuracy on FER-2013 test data set [88]. Outputted confusion matrix is on figure 8.2.

Tab. 8.3: RetinaFace AP on validation data set Wider Face

Architecture	Easy	Medium	Hard
Resnet-50	95.482%	94.046%	84.430%
MobileNet-0.25	90.708%	88.165%	73.827%

Tab. 8.4: Table of UTK test data set evaluation, evaluated with weights trained on different data sets, by original CORAL network

Training data set	UTK 21-41 MAE	UTK 21-41 RMSE	UTK 1-100 MAE	UTK 1-100 RMSE
AFAD	5.35	6.81	13.97	19.36
CACD	11.13	13.41	11.49	14.84
MORPH	7.32	8.96	12.12	15.82
UTK	7.53	10.12	9.63	13.19
1-100 UTK	5.09	7.01	6.39	8.86

Table 8.4 shows evaluation of all CORAL weights used in this thesis. Weights 1-100 UTK were trained in this thesis, other weights were trained by CORAL authors [12]. Left side of the table has results for age range intersection of tested weights, and right side for whole UTK test data set age range.

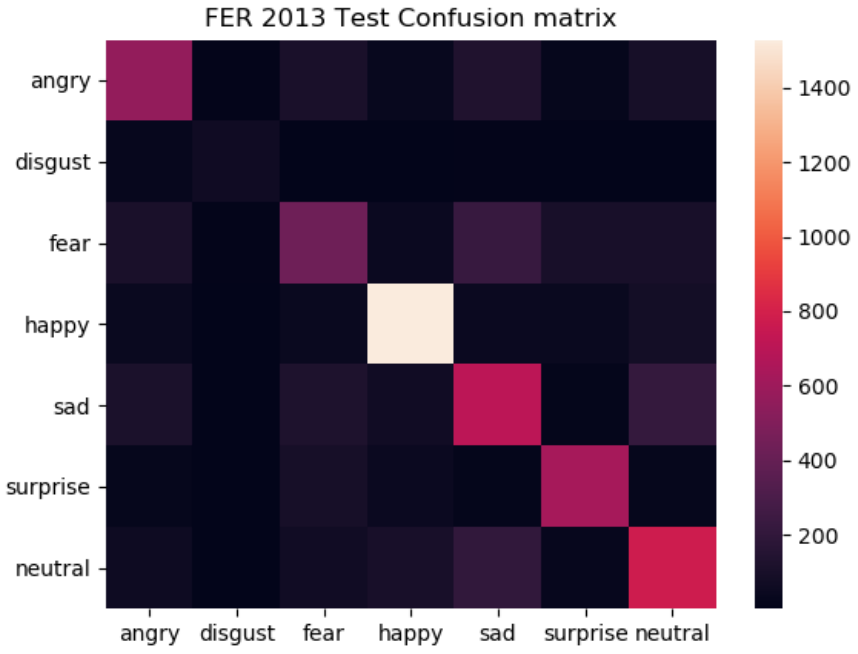


Fig. 8.2: Confusion matrix of FER-2013 data set evaluation, evaluated by emotion mini-Xception network

8.2 AVS data set

New AVS data set has training and test parts. Training data set is made from three new videos from SoulPancake Youtube channel and test data set is made from one of older videos [101]. Extracted scenes have trademark of Kapwing platform which

was used for extraction [103]. This trademark is never covering face so it has no effect on data set facial information. Training data set has 80 unique subjects and 808 sequences. Test data set is smaller and has 33 unique subjects and 35 sequences. Both data sets overlap in about 11 subjects but these subject have age difference as can be seen on right example from figures 8.3 and 8.4. Also both have subjects with glasses training set has 11 and test set 4. Each data set has great variability in ethnicity but their representation is not equal and test data sets lacks subjects of Asian descent. Subjects with hard to discern ethnicity were categorized into Mixed descent (including people of Arab or various native descent). Table 8.5 shows data sets subject properties.

Tab. 8.5: Table of subjects properties for new AVS data set

AVS Dataset	Number of subjects	Male	Female	African	Asian	Caucasian	Indian	Mixed
Training	80	25	55	13	7	38	6	16
Test	33	14	19	4	0	20	3	6

Fig. 8.3: Example of images from AVS training data set

Fig. 8.4: Example of images from AVS test data set

On age group histograms in figure 8.5, can be seen data sets subjects age layout. Next figure 8.6 showcases same histograms, but for data sets sequences. Because some subjects had longer scenes more sequences of their age group are in data set, which can have effect on training CORAL-LSTM.

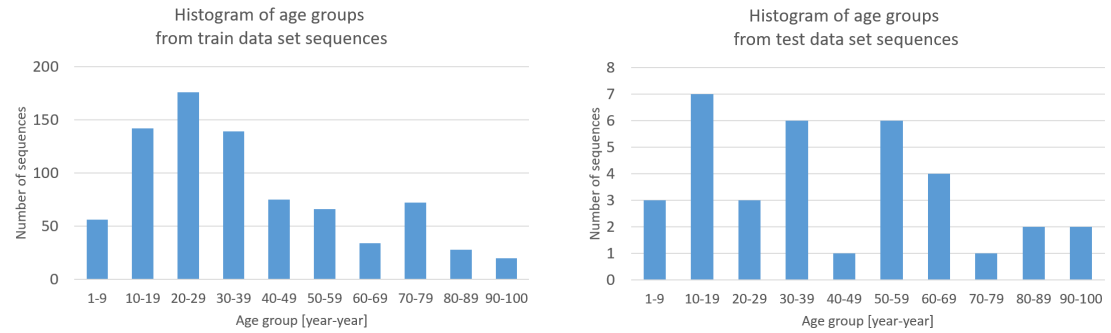


Fig. 8.5: Histogram of age groups from subjects in AVS train (left) and test (right) video data set

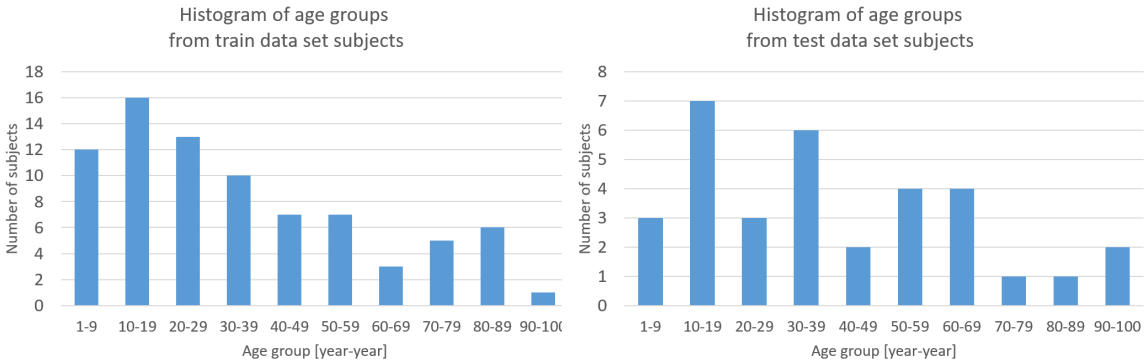


Fig. 8.6: Histogram of age groups from sequences in AVS train (left) and test (right) video data set

8.3 CORAL-LSTM performance comparison

Following tables 8.6,8.7,8.8 show CORAL-LSTM performance on AVS data set for different number of LSTM layers and training epochs. CORAL-LSTM network LSTM layers average propagation speed for different number of LSTM layers is in table 8.9.

For CORAL-LSTM architecture, best performing weights on AVS test data set, were trained with setting two LSTM layers after 75 epochs. Therefore unfreezing of all layers or unfreezing last fully connected classification layer was tested with same settings in table 8.10.

Final comparison between best performing weights on AVS test data set (2 LSTM layers,75 epochs,unfreeze LSTM layers only) and original weights without LSTM insertion, trained only on UTK data set is on table 8.11. This table also compares with performance of pretrained weights from CORAL authors. Best CORAL-LSTM network had average inference time 0.0159s. Which is longer than unchanged CORAL and reflected in longer one image inference in solution with CORAL-LSTM. Specifically new one image inference for images with 1920x1080 resolution was 0.3769s.

Tab. 8.6: Table of CORAL-LSTM network with 1 LSTM layer, evaluated on AVS data set for various number of training epochs

Training epoch	AVS training MAE	AVS training RMSE	AVS test MAE	AVS test RMSE
55	4.3	6.02	8.97	11.38
60	4.26	6.41	8.06	10.48
65	5.29	7.47	8.69	11.79
70	4.28	6.36	9.91	13.3
75	3.89	5.82	9.4	11.22
80	3.34	5.08	9.69	13.05
85	3.39	5.02	8.94	12.37

Tab. 8.7: Table of CORAL-LSTM network with 2 LSTM layers, evaluated on AVS data set for various number of training epochs

Training epoch	AVS training MAE	AVS training RMSE	AVS test MAE	AVS test RMSE
55	3.99	5.76	8.83	11.03
60	5.41	8.29	10.6	13.83
65	4.97	7.21	8.46	11.58
70	4.38	6.51	9.31	11.39
75	3.63	5.47	7.14	9.34
80	3.54	5.39	7.89	10.22
85	4.03	5.92	7.17	10.07

Tab. 8.8: Table of CORAL-LSTM network with 3 LSTM layers, evaluated on AVS data set for various number of training epochs

Training epoch	AVS training MAE	AVS training RMSE	AVS test MAE	AVS test RMSE
55	12.44	17.74	16.31	21.33
60	8.94	11.73	12.06	14.48
65	9.68	13.01	13.2	17.17
70	10.76	14.98	13.86	17.37
75	8.94	11.78	12.69	16
80	7.65	10.32	12.31	14.66
85	8.05	10.5	12.06	14.31

Tab. 8.9: LSTM layers average propagation speed for CORAL-LSTM architecture

LSTM layers	CPU propagation	GPU propagation
1	0.08677s	0.00071s
2	0.16824s	0.00110s
3	0.25903s	0.00155s

Tab. 8.10: Table of comparison in age prediction performance between weights trained by best found number of LSTM layers and epoch setting (2 layers,75 epochs), but for different unfreezing settings, on AVS data set

Unfreezed layers	AVS training MAE	AVS training RMSE	AVS test MAE	AVS test RMSE
All	8.61	12.15	9.34	11.54
LSTM + classification layer	8.92	12.42	10.26	13.98
LSTM	3.63	5.47	7.14	9.34

Tab. 8.11: Table of comparison in age prediction performance for all pretrained weights from CORAL authors and new 1-100 trained UTK weights and best CORAL-LSTM weights on AVS data set

Training data set	AVS training MAE	AVS training RMSE	AVS test MAE	AVS test RMSE
AFAD	17.04	22.69	22.14	28.12
CACD	15.07	19.01	16.08	20.07
MORPH	18.3	22.43	19.14	24.02
UTK	15.56	19.21	15.79	19.99
1-100 UTK	9.65	13.15	9.7	12.94
CORAL-LSTM	3.63	5.47	7.17	9.34

9 Discussion

All achieved results will be discussed under one of the two following fields. First being human recognition solution, its accuracy, speed and potential live-inference viability. And second field of improving prediction accuracy by changing single image network architecture into recurrent architecture. Last section will discuss possible improvements to the solution.

9.1 Recognition solution

Implemented solution has achieved functionality defined by thesis instructions, and expanded it, with additional emotion classification as can be seen on examples 8.1.

Average inference speeds on tables 8.1 and 8.2 show that most time consuming task is face detection and it heavily depends on image resolution when using CPU. RetinaFace ResNet-50 was as authors stated in their paper optimized for GPU use only [23]. Other network speed performance was not as much substantially changed by change of resolution, therefore solution can be capable for live inference applications if used with GPU or with CPU on images with lower resolution. All of this of course relies on number of faces on image, but for 3 to 5 faces should solution with RetinaFace MobileNet-0.25 architecture on GPU, be viable for use.

On individual CNN task performance, can be stated that face detection functions properly, classification of gender and emotion has same state-of-the-art accuracy as authors of respective paper claimed. Age prediction has 4 different weights from original authors and 1 new weights trained in this thesis for still images. Their comparison on table 8.4, shows that performance of new 1-100 UTK weights is better than weights trained by CORAL authors in their PyTorch implementation. This performance is a lot more close to performance CORAL authors achieved in their paper using other framework than PyTorch, which can be seen on table 5.1 from chapter 5. Modified solution using CORAL-LSTM network will be discussed in the next section.

9.2 LSTM impact on age inference

Architecture CORAL used for age prediction was experimentally modified with LSTM layers with aim of improving age prediction accuracy. This new CORAL-LSTM architecture for training and testing, required a data set consisting of video sequences with labeled age. This has been achieved with creation of Age Video Sequence data set. Training data set has 808 sequences from 80 unique subjects and test data set has 35 sequences from 33 unique subjects. Each sequence is made

up from 20 frames in time order containing single person. Some sequences contain at start few freezed frames, this error is attributed to faulty scene extraction by Kapwing platform [103]. All sequence frames have age label and face BB parameters stored in .csv file. While AVS has great share of ethnic diversity it has downside in non equal representation as can be seen on table 8.5. Subjects are more Caucasian-centric and male to female ratio is heavily skewed to female gender, with greatest flaw being zero Asian subjects in AVS test data set. Nevertheless there is good number of different poses and faces of various scales in the data set, which is visible on examples 8.3 and 8.4. Effect of 11 subjects overlapping between data sets is mitigated by age difference between subject scenes.

On histograms 8.5 and 8.6 is representation for each age group in the data sets. Since some subjects had longer scenes or even more than one scene, their age group has more sequences from these subjects which leads to change in histogram pattern between subject and sequence histograms of same data set. This can be seen on training data set where age groups 1-9 and 80-89 had more sequences per subject in their age group then other age groups. Since aim of test data set is to be most objective benchmark for age prediction and LSTM improvement its sequences have in nearly all cases unique subject, therefore there is not much change in histogram pattern between subject to sequence histograms. Overall the CORAL weight importance technique, which has been used during training, helped to mitigate this effect of unequal age group representation in sequences.

New 1-100 UTK weights for original CORAL network performed with same accuracy as UTK weights trained by authors. Next 1-100 UTK weights were used as start point to fine-train on AVS training data set best training setting for CORAL-LSTM architecture. This training setting had number of LSTM layers based from tables 8.6, 8.7 and 8.8, which show that 2 LSTM layers are better performing than 1 or 3 layers. After that for 2 LSTM layer setting were tested also options for unfreezing more than just LSTM layers. Results of this test on table 8.10 confirm that unfreezing just LSTM layers is best option. And finally table 8.11 shows performance of CORAL authors weights and 1-100 UTK weights in original CORAL architecture against CORAL-LSTM architecture weights trained on best found training setting. Pretrained CORAL weights results should be taken with reserve, as their low performance can be reasonably explained by having been trained on smaller age range than 1-100 years, therefore only 1-100 UTK weights and CORAL-LSTM should be noted as equally compared weights. From results can be stated that CORAL-LSTM network performs better at AVS test data set making it objectively better age predicting architecture than its original CORAL architecture. This comes at cost of needing 20 frame time sequence and increased propagation time from LSTM layers which can be seen on table 8.9. It should be noted that solution using CORAL-

LSTM is for cases with single person in video data. If there were multiple persons there would be need to create a way to store and sort BB detections by each person, during each image inference. Which could prove ambiguous to make for videos where person scene changes between whole body images and close ups on face. And would inevitably lead to very long inference times.

9.3 Future work

Improvements to human recognition solution in this thesis can be divided into 3 categories: improving individual network performance or adding additional recognition functionality, enlarging and refining AVS data set and optimizing solutions code.

Based on CORAL-LSTM results it can be said that each other used network could have its performance improved by adding LSTM layers, this of course requires appropriate data sets. Or changing architecture for task with better one, which are published on year to year basis. Since live-inference on CPU is important goal it could be achieved by reducing face detection inference time by resizing input image into lower resolution and then transform detected face BB parameters into original image dimensions. This could come at cost of accuracy but with increased speed, live-inference for CPU would become viable. Reduced speed creates time for more possibilities using facial analysis, like skin tone, eye,hair color classification or scar,tattoo detection or large scale facial landmark extraction.

CORAL-LSTM performance could be improved by fixing AVS data sets faults like unequal ethnicity and gender representation or freezed frames. Both AVS training and test data set could be enlarged with images from subjects of rare age groups.

Code optimizing can be done in many different ways. To start with original image is loaded multiple times by different libraries, this could be done by just one loading and use just one of image processing libraries. CORAL-LSTM as it is now makes during each inference 19 out of 20 feature extractions (before LSTM concatenation) redundant, if these features could be saved in memory instead of new inference, CORAL and CORAL-LSTM inference time difference would be diminished. Another option would be creating GUI or making both "Recognition_net.py" and "Recognition_net_LSTM.py" usable on video files. Best possible improvement in terms of code is to implement it in other programming language. Torch has a C++ front end interface capable of running weights trained by Pytorch in python. This could be further improved by better hardware resources allocation. This can be done by not using neural frameworks, and loading models only in weights form using ONNX format to load into performance-focused inference engine like ONNX Runtime [105], which can run atop of NVIDIA TensorRT Inference Accelerator [104].

Conclusion

This thesis attempts to solve task of human recognition as series of distinct CNNs interconnected to create automated human face detection and subsequent age, gender and emotion classification from face images.

It starts with describing a vast part of convolution neural networks theory, including data science interlinked to training such networks and has small introduction to recurrent networks. It reviews basic architectures and their specialized counterparts for task of object detection. RetinaNet object detection architecture is thoroughly examined and followed by same treatment for its specialized version RetinaFace, designed for facial detection. In similar, but modest way is made review of gender and emotion classification methods and with it an architecture is picked which solves both tasks in convenient and precise manner. Next is basic review of age estimation algorithms and introduction to neural networks solving ordinal regression problems. Age estimation algorithm using ordinal approach named Consistent Rank Logits is picked and examined. Thesis theory concludes with proposition of enhancement for age prediction accuracy by harnessing temporal information with LSTM layers, effectively changing network from analyzing still images to recurrent network.

Continuing with reviews there are reviews for facial detection, emotion and age data sets, with gender often anotated among these data sets. And last review is oriented on popular neural network frameworks. Then thorough implementation covers human recognition solution for all required tasks. To implement LSTM enhancement new data set is needed. Therefore Age Video Sequence data set creation process is covered and subsequently is given detailed implementation of LSTM enhancement to age prediction CORAL network.

Results indicate that implemented human recognition solution has reasonable accuracy and one image inference time, when using RetinaFace lightweight architecture which uses MobileNet for feature extraction. But this is confirmed only for when network model is run on GPU with low number of persons. On CPU even using light weight face detection has overall image inference time around 0.3 second which is slow for live inference, where FPS values range from 30 to 60. And since this face detection time is linked to image resolution, possible remedy based on image downsizing is discussed in discussion. Other networks used for facial analysis of gender and emotion had already good inference speed and provided adequate precision. Age classification network CORAL had second worst inference speed after face detection network but since it uses fixed size input, its inference speed did not suffer from higher resolutions. In terms of accuracy evaluation, were used pretrained weights and one set of weights trained in the thesis. To objectively compare their performance on chosen data set, evaluation has been done twice. First time only

images from their shared age range were used for evaluation and second time whole age range. Because for unknown reason to thesis author, pretrained weights were not trained for whole age range of their data sets. Results showed that trained weights had performance akin to authors weights trained on same data set.

But main purpose of these weights was to use them as start point to fine-train added LSTM layers and compare before and after addition age prediction performance. Here was used Age Video Sequence data set to train and test CORAL-LSTM architecture. This data set was created with good number of unique subjects in different poses and facial expression. And while subjects ethnicity diversity is great certain groups had unequal representation in data set, mainly subjects of Asian descent in test data set. Age group histograms showed skew of subjects and their sequences into younger age groups, but this has been partially mitigated by using weight importance for age groups. And since largest facial transformation phases are during younger years their over representation should not be fully considered as having only bad impact. On AVS training data set was trained CORAL-LSTM architecture and various training setting were tested mainly optimal number of LSTM layers and whether should be during training unfreezed layers, which were trained as part of original CORAL architecture. Test concluded that 2 LSTM layers and unfreezing only these layers was best training setting. Best trained CORAL-LSTM weights were then used in comparison with pretrained CORAL weights and 1-100 weights. Since comparison was made on AVS data set using age range 1-100 pretrained weights showed bad performance, but results show that CORAL-LSTM architecture outperformed original CORAL architecture. Lastly thesis implement expanded human recognition solution using CORAL-LSTM architecture where before age prediction must be first made face detection on 20 frames from time sequence to produce 20 face images in time order as input to CORAL-LSTM network. This solution had small inference time increase attributed to LSTM layer propagation which scales linearly with number of added layers.

Possible improvements solving speed inference problem using image downsizing and code optimization by circumventing neural frameworks and using just weights as model for performance-focused engines were proposed. Accuracy improvement for networks other than age could be done experimentally by implementing LSTM layers, or by using new architectures. Otherwise here presented human recognition solution, if inference speed is ignored, can always be expanded with new tasks based on facial analysis.

Bibliography

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M., 2016. *Tensorflow: A system for large-scale machine learning*. In 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16) (pp. 265-283).

[2] Afifi, M. and Abdelhamed, A., 2019. *AFIF4: deep gender classification based on adaboost-based fusion of isolated facial features and foggy faces*. Journal of Visual Communication and Image Representation, 62, pp.77-86.

[3] Antipov, G., Baccouche, M., Berrani, S.A. and Dugelay, J.L., 2017. *Effective training of convolutional neural networks for face-based gender and age prediction*. Pattern Recognition, 72, pp.15-26.

[4] Arriaga, O., Valdenegro-Toro, M. and Plöger, P., 2017. *Real-time convolutional neural networks for emotion and gender classification*. arXiv preprint arXiv:1710.07557.

[5] Berg, T.L., Berg, A.C., Edwards, J. and Forsyth, D.A., 2005. *Who's in the picture*. In Advances in neural information processing systems (pp. 137-144).

[6] BEKIOS-CALFA, Juan; BUENAPOSADA, Jose M.; BAUMELA, Luis. *Revisiting linear discriminant techniques in gender recognition*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2010, 33.4: 858-864.

[7] Bianco, S., 2017. *Large age-gap face verification by feature injection in deep networks*. Pattern Recognition Letters, 90, pp.36-42.

[8] BODLA, Navaneeth, et al. *Soft-NMS–Improving Object Detection With One Line of Code*. In: Proceedings of the IEEE international conference on computer vision. 2017. p. 5561-5569.

[9] BRUNELLI, Roberto; POGGIO, Tomaso. *Face recognition: Features versus templates*. IEEE transactions on pattern analysis and machine intelligence, 1993, 15.10: 1042-1052.

[10] BUI, Len, et al. *Face gender recognition based on 2D principal component analysis and support vector machine*. In: 2010 Fourth International Conference on Network and System Security. IEEE, 2010. p. 579-582.

[11] Cao, Q., Shen, L., Xie, W., Parkhi, O.M. and Zisserman, A., 2018, May. *Vggface2: A dataset for recognising faces across pose and age*. In 2018

13th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2018) (pp. 67-74). IEEE.

[12] Cao, W., Mirjalili, V. and Raschka, S., 2019. *Consistent rank logits for ordinal regression with convolutional neural networks*. arXiv preprint arXiv:1901.07884.

[13] Chen, B.C., Chen, C.S. and Hsu, W.H., 2014, September. *Cross-age reference coding for age-invariant face recognition and retrieval*. In European conference on computer vision (pp. 768-783). Springer, Cham.

[14] Chen, J.C., Kumar, A., Ranjan, R., Patel, V.M., Alavi, A. and Chellappa, R., 2016, September. *A cascaded convolutional neural network for age estimation of unconstrained faces*. In 2016 IEEE 8th International Conference on Biometrics Theory, Applications and Systems (BTAS) (pp. 1-8). IEEE.

[15] S. Chen, C. Zhang, M. Dong, J. Le and M. Rao, "*Using Ranking-CNN for Age Estimation*," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 742-751. doi: 10.1109/CVPR.2017.86

[16] CHO, Kyunghyun, et al.*On the properties of neural machine translation: Encoder-decoder approaches*. arXiv preprint arXiv:1409.1259, 2014.

[17] François Chollet, 2015 *Keras* GitHub, GitHub repository, [Online], https://github.com/fchollet/keras, commit = 5bcac37

[18] CHOLLET, François. *Xception: Deep learning with depthwise separable convolutions*. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017. p. 1251-1258.

[19] Chu, W. and Keerthi, S.S., 2005, August. *New approaches to support vector ordinal regression*. In Proceedings of the 22nd international conference on Machine learning (pp. 145-152). ACM.

[20] Crammer, K. and Singer, Y., 2002. *Pranking with ranking*. In Advances in neural information processing systems (pp. 641-647).

[21] Daniel, G., 2013. *Principles of artificial neural networks (Vol. 7)*. World Scientific.

[22] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K. and Fei-Fei, L., 2009, June. *Imagenet: A large-scale hierarchical image database*. In 2009 IEEE conference on computer vision and pattern recognition (pp. 248-255). Ieee.

[23] Deng, J., Guo, J., Zhou, Y., Yu, J., Kotsia, I. and Zafeiriou, S., 2019. *RetinaFace: Single-stage Dense Face Localisation in the Wild*. arXiv preprint arXiv:1905.00641.

[24] DONAHUE, Jeffrey, et al. *Long-term recurrent convolutional networks for visual recognition and description*. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2015. p. 2625-2634.

[25] DU, Shichuan; TAO, Yong; MARTINEZ, Aleix M. Compound facial expressions of emotion. Proceedings of the National Academy of Sciences, 2014, 111.15: E1454-E1462.

[26] Dumoulin, V. and Visin, F., 2016. *A guide to convolution arithmetic for deep learning*. arXiv preprint arXiv:1603.07285.

[27] Eidinger, E., Enbar, R. and Hassner, T., 2014. *Age and gender estimation of unfiltered faces*. IEEE Transactions on Information Forensics and Security, 9(12), pp.2170-2179.

[28] EKMAN, Paul; FRIESEN, Wallace V. *Facial action coding system: Investigator's guide*. Consulting Psychologists Press, 1978.

[29] Everingham, M., Van Gool, L., Williams, C.K., Winn, J. and Zisserman, A., 2007. *The PASCAL visual object classes challenge 2007 (VOC2007) results*.

[30] FELLOUS, Jean-Marc. *Gender discrimination and prediction on the basis of facial metric information*. Vision research, 1997, 37.14: 1961-1973.

[31] FU, Cheng-Yang, et al. *Dssd: Deconvolutional single shot detector*. arXiv preprint arXiv:1701.06659, 2017.

[32] Fukushima, K., 1988. *Neocognitron: A hierarchical neural network capable of visual pattern recognition*. Neural networks, 1(2), pp.119-130.

[33] GHIMIRE, Deepak; LEE, Joonwhoan. Geometric feature-based facial expression recognition in image sequences using multi-class adaboost and support vector machines. Sensors, 2013, 13.6: 7714-7734.

[34] Gidaris, S. and Komodakis, N., 2015. *Object detection via a multi-region and semantic segmentation-aware cnn model*. In Proceedings of the IEEE international conference on computer vision (pp. 1134-1142).

[35] GIRSHICK, Ross, et al. *Rich feature hierarchies for accurate object detection and semantic segmentation*. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2014. p. 580-587.

[36] GIRSHICK, Ross. *Fast r-cnn*. In: Proceedings of the IEEE international conference on computer vision. 2015. p. 1440-1448.

[37] Grgic, M., Delac, K. and Grgic, S., 2011. *SCface–surveillance cameras face database*. Multimedia tools and applications, 51(3), pp.863-879.

[38] HAPPY, S. L.; GEORGE, Anjith; ROUTRAY, Aurobinda. *A real time facial expression classification system using local binary patterns*. In: 2012 4th International conference on intelligent human computer interaction (IHCI). IEEE, 2012. p. 1-5.

[39] HE, Kaiming, et al. *Deep residual learning for image recognition*. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. p. 770-778.

[40] Ho D, Liang E, Stoica I, Abbeel P, Chen X. *Population Based Augmentation: Efficient Learning of Augmentation Policy Schedules*. arXiv preprint arXiv:1905.05393. 2019 May 14.

[41] HOSANG, Jan, et al. *What makes for effective detection proposals?*. IEEE transactions on pattern analysis and machine intelligence, 2015, 38.4: 814-830.

[42] HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. *Long short-term memory*. Neural computation, 1997, 9.8: 1735-1780.

[43] Huang, G.B., Mattar, M., Berg, T. and Learned-Miller, E., 2008, October. *Labeled faces in the wild: A database forstudying face recognition in unconstrained environments*.

[44] Huang, G.B. and Learned-Miller, E., 2014. *Labeled faces in the wild: Updates and new reporting procedures*. Dept. Comput. Sci., Univ. Massachusetts Amherst, Amherst, MA, USA, Tech. Rep, pp.14-003.

[45] Jain, V. and Learned-Miller, E., 2010. *Fddb: A benchmark for face detection in unconstrained settings*.

[46] JUNG, Heechul, et al. *Joint fine-tuning in deep neural networks for facial expression recognition*. In: Proceedings of the IEEE international conference on computer vision. 2015. p. 2983-2991.

[47] Ko, B.C., 2018. *A brief review of facial emotion recognition based on visual information*. sensors, 18(2), p.401.

[48] Kollář, R.; Chmelík J.; Jakubíček R., 2019. *Machine learning lectures*. VUT FEKT MPC-MLR course.

[49] KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. *Imagenet classification with deep convolutional neural networks*. In: Advances in neural information processing systems. 2012. p. 1097-1105.

[50] KUZNETSOVA, Alina, et al. *The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale*. arXiv preprint arXiv:1811.00982, 2018.

[51] Lanitis, A., Draganova, C. and Christodoulou, C., 2004. *Comparing different classifiers for automatic age estimation*. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 34(1), pp.621-628.

[52] LeCun, Y., Bengio, Y. and Hinton, G., 2015. *Deep learning*. nature, 521(7553), pp.436-444.

[53] LECUN, Yann, et al. *Gradient-based learning applied to document recognition*. Proceedings of the IEEE, 1998, 86.11: 2278-2324.

[54] Learned-Miller, E., Huang, G.B., RoyChowdhury, A., Li, H. and Hua, G., 2016. *Labeled faces in the wild: A survey*. In Advances in face detection and facial image analysis (pp. 189-248). Springer, Cham.

[55] Levi, G. and Hassner, T., 2015. *Age and gender classification using convolutional neural networks*. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops (pp. 34-42).

[56] Li, L. and Lin, H.T., 2007. *Ordinal regression by extended binary classification*. In Advances in neural information processing systems (pp. 865-872).

[57] Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B. and Belongie, S., 2017. *Feature pyramid networks for object detection*. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2117-2125).

[58] Lin TY, Goyal P, Girshick R, He K, Dollár P. *Focal loss for dense object detection*. InProceedings of the IEEE international conference on computer vision 2017 (pp. 2980-2988).

[59] Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C.L., 2014, September. *Microsoft coco: Common objects in context*. In European conference on computer vision (pp. 740-755). Springer, Cham.

[60] LIPTON, Zachary C.; BERKOWITZ, John; ELKAN, Charles. *A critical review of recurrent neural networks for sequence learning*. arXiv preprint arXiv:1506.00019, 2015.

[61] LIU, Wei, et al. *Ssd: Single shot multibox detector*. In: European conference on computer vision. Springer, Cham, 2016. p. 21-37.

[62] Liu, Z., Luo, P., Wang, X. and Tang, X., 2015. *Deep learning face attributes in the wild*. In Proceedings of the IEEE international conference on computer vision (pp. 3730-3738).

[63] LUCEY, Patrick, et al. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In: 2010 ieee computer society conference on computer vision and pattern recognition-workshops. IEEE, 2010. p. 94-101.

[64] Lundqvist, D., Flykt, A., and Öhman, A. (1998). The Karolinska Directed Emotional Faces – KDEF, CD ROM from Department of Clinical Neuroscience, Psychology section, Karolinska Institutet, ISBN 91-630-7164-9.

[65] LYONS, Michael, et al. Coding facial expressions with gabor wavelets. In: Proceedings Third IEEE international conference on automatic face and gesture recognition. IEEE, 1998. p. 200-205.

[66] Mallat, K. and Dugelay, J.L., 2018, September. *A benchmark database of visible and thermal paired face images across multiple variations*. In 2018 International Conference of the Biometrics Special Interest Group (BIOSIG) (pp. 1-5). IEEE.

[67] MAVADATI, S. Mohammad, et al. Disfa: A spontaneous facial action intensity database. IEEE Transactions on Affective Computing, 2013, 4.2: 151-160.

[68] NG, Choon-Boon; TAY, Yong-Haur; GOI, Bok-Min. *A review of facial gender recognition*. Pattern Analysis and Applications, 2015, 18.4: 739-755.

[69] Niu, Z., Zhou, M., Wang, L., Gao, X. and Hua, G., 2016. *Ordinal regression with multiple output cnn for age estimation*. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4920-4928).

[70] Odena, A., Dumoulin, V. and Olah, C., 2016. *Deconvolution and checkerboard artifacts*. Distill, 1(10), p.e3.

[71] Oksuz, K., Cam, B.C., Kalkan, S. and Akbas, E., 2019. *Imbalance problems in object detection: A review*. arXiv preprint arXiv:1909.00169.

[72] PASZKE, A.; GROSS, S.; CHINTALA, S.; et al.: *PyTorch*. [online]. [Cit. 2019-05-10]. Retrieved from: https://pytorch.org

[73] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga and A. Lerer, "*Automatic differentiation in PyTorch*," NIPS-W, 2017

[74] PANTIC, Maja, et al. Web-based database for facial expression analysis. In: 2005 IEEE international conference on multimedia and Expo. IEEE, 2005. p. 5 pp.

[75] Polania, L., Fung, G. and Wang, D., 2019, January. *Ordinal Regression using Noisy Pairwise Comparisons for Body Mass Index Range Estimation*. In 2019 IEEE Winter Conference on Applications of Computer Vision (WACV) (pp. 782-790). IEEE.

[76] POLIKOVSKY, Senya; KAMEDA, Yoshinari; OHTA, Yuichi.*Facial micro-expressions recognition using high speed camera and 3D-gradient descriptor*. 2009.

[77] Ranjan, R., Sankaranarayanan, S., Castillo, C. and Chellappa, R., University of Maryland and College Park, 2019. *All-in-one convolutional neural network for face analysis*. U.S. Patent Application 16/340,859.

[78] REDMON, Joseph; FARHADI, Ali. *Yolov3: An incremental improvement*. arXiv preprint arXiv:1804.02767, 2018.

[79] REN, Shaoqing, et al. *Faster r-cnn: Towards real-time object detection with region proposal networks*. In: Advances in neural information processing systems. 2015. p. 91-99

[80] REZATOFIGHI, Hamid, et al. *Generalized intersection over union: A metric and a loss for bounding box regression*. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019. p. 658-666.

[81] Ricanek, K. and Tesafaye, T., 2006, April. *Morph: A longitudinal image database of normal adult age-progression*. In 7th International Conference on Automatic Face and Gesture Recognition (FGR06) (pp. 341-345). IEEE.

[82] Rothe, R., Timofte, R. and Van Gool, L., 2018. *Deep expectation of real and apparent age from a single image without facial landmarks*. International Journal of Computer Vision, 126(2-4), pp.144-157.

[83] SABOUR, Sara; FROSST, Nicholas; HINTON, Geoffrey E. *Dynamic routing between capsules. In: Advances in neural information processing systems*. 2017. p. 3856-3866.

[84] Shan, C., 2012. *Learning local binary patterns for gender classification on real-world face images*. Pattern recognition letters, 33(4), pp.431-437.

[85] SIMONYAN, Karen; ZISSERMAN, Andrew. *Very deep convolutional networks for large-scale image recognition*. arXiv preprint arXiv:1409.1556, 2014.

[86] SZEGEDY, Christian, et al. *Going deeper with convolutions*. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2015. p. 1-9.

[87] THALLES, Silva, 2019 *Everything you need to know about TensorFlow 2.0*. [Online], Retrieved from: https://hackernoon.com/everything-you-need-to-know-about-tensorflow-2-0-b0856960c074

[88] Wolfram Research ,*The Facial Expression Recognition 2013 (FER-2013) Dataset*. From the Wolfram Data Repository (2018).

[89] Zbořil František V., 2019. *Soft computing lectures*. VUT FIT Soft computing course.

[90] ZHANG, Xing, et al. A high-resolution spontaneous 3d dynamic facial expression database. In: 2013 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG). IEEE, 2013. p. 1-6.

[91] Zhang, Z., Song, Y. and Qi, H., 2017. *Age progression/regression by conditional adversarial autoencoder*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5810-5818).

[92] ZHAO, Kaili; CHU, Wen-Sheng; ZHANG, Honggang. *Deep region and multi-label learning for facial action unit detection*. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016. p. 3391-3399.

[93] ZHENG, Zhaohui, et al. *Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression*. arXiv preprint arXiv:1911.08287, 2019.

[94] ZHOU, Xingyi; WANG, Dequan; KRÄHENBÜHL, Philipp. *Objects as Points*. arXiv preprint arXiv:1904.07850, 2019.

[95] Zhou, Y., Deng, J., Kotsia, I. and Zafeiriou, S., 2019. *Dense 3D Face Decoding over 2500FPS: Joint Texture and Shape Convolutional Mesh Decoders*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1097-1106).

[96] Yang, S., Luo, P., Loy, C.C. and Tang, X., 2016. *Wider face: A face detection benchmark*. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 5525-5533).

[97] Yi, D., Lei, Z., Liao, S. and Li, S.Z., 2014. *Learning face representation from scratch*. arXiv preprint arXiv:1411.7923.

[98] YIN, Lijun, et al. A 3D facial expression database for facial behavior research. In: 7th international conference on automatic face and gesture recognition (FGR06). IEEE, 2006. p. 211-216.

[99] Yu, F., Koltun, V. and Funkhouser, T., 2017. *Dilated residual networks*. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 472-480).

[100] Glorot, X. and Bengio, Y., 2010, March. *Understanding the difficulty of training deep feedforward neural networks*. In Proceedings of the thirteenth international conference on artificial intelligence and statistics (pp. 249-256).

[101] *0-100 series, SoulPancace, Youtube* [Online] at https://www.youtube.com/playlist?list=PLzvRx-johoA-7DXw5EThWpQlVYH0hm1aC

[102] *Facial recognition: top 7 trends (tech, vendors, markets, use cases and latest news)* [Online] at https://www.gemalto.com/govt/biometrics/facial-recognition

[103] *https://www.kapwing.com/* [Online], Kapwing is a collaborative platform for creating images, videos, and GIFs.

[104] *NVIDIA TensorRT™ - SDK for high-performance deep learning inference* [Online] at https://developer.nvidia.com/tensorrt

[105] *ONNX Runtime: cross-platform, high performance scoring engine for ML models* [Online] at https://github.com/microsoft/onnxruntime

[106] *https://paperswithcode.com/task/face-detection*, [Online], Useful web page with leaderboards of algorithms performance on various tasks, mostly involving Machine learning.

[107] *www.towardsdatascience.com*,[Online], Web page dedicated to data science content

List of symbols, physical constants and abbreviations

AF	activation function
AN	artificial neuron
AP	average precision of classificator in detections of one class
API	application programming interface
AR	averaged recall
AVS	Age Video Sequence data set
BB	bounding box
CE	croos-entropy loss function
$CIoU$	Complete Intersection over Union
CNN	convolutional neural network
CORAL	Consistent Rank Logits
CORAL-LSTM	CORAL architecture with added LSTM layers
CPU	central processing unit
CS	Cumulative score metric
$DIoU$	Distance Intersection over Union
E	error,loss or objective function
FCN	fully convolutional neural network
FE	feature extractor network
FPN	feature pyramid network
FPS	frames per second
$GIoU$	Generalized Intersection over Union
GPU	graphic processing unit
GT	ground truth
IoU	Intersection over Union
LBF	Linear base function
LSTM	long short term memory
MAE	mean absolute error
m**AP**	mean average precision
m**AR**	mean averaged recall
NN	neural network
NMS	non-maximum suppression
OD	deep learning object detection
OF	overfitting
OR	ordinal regression
PRC	precision-recall curve

RBF	Radial base function
R-CNN	regional convolutional neural network
ReLU	Rectified Linear Unit
RMSE	root mean squared error
SSD	Single shot detector
UF	underfitting
YOLO	You only live once