DOI
https://doi.org/
10.18690/um.feri.1.2024.7

ISBN
978-961-286-837-6

SPROTNA ANALIZA SLIK VOZIL Z METODAMI GLOBOKEGA UČENJA V OGRODJU FLUTTER

ALEKSANDR SHISHKOV,[1,3] STEVANČE NIKOLOSKI[2,3]

[1] Univerza v Mariboru Fakulteta za elektrotehniko, računalništvo in informatiko, Maribor, Slovenija
aleksandr.shishkov@student.um.si
[2] Univerza v Novem mestu Fakulteta za ekonomijo in informatiko, Novo mesto, Slovenija
stevance.nikoloski@uni-nm.si
[3] Result d.o.o., Ljubljana, Slovenia
aleksandr.shishkov@student.um.si, stevance.nikoloski@uni-nm.si

V članku raziskujemo integracijo modela MobileNetV3 v ogrodju Flutter, osredotočajoč se na napredno klasifikacijo slik avtomobilov. Preučujemo večplasten pristop, ki vključuje uporabo raznolikih podatkovnih zbirk, fino prilagajanje modela ter njegovo brezhibno implementacijo v mobilno aplikacijo. S poudarkom na izboljšanju uporabniške izkušnje smo ustvarili tri specializirane modele z visoko stopnjo natančnosti (97%), ki prepoznajo ustrezne slike, klasificirajo tip slike (vozilo, armaturna plošča ali dokument) ter določajo stran avtomobila (spredaj, levo, desno, zadaj). Rezultati kažejo izjemno hitrost in odzivnost aplikacije, pri čemer MobileNetV3 zagotavlja natančno klasifikacijo v le 60 ms, kar prispeva k izjemni učinkovitosti celotnega sistema.

Ključne besede:
MobileNetV3,
Flutter,
klasifikacija slik,
fino prilagajanje,
globoko učenje

Univerzitetna založba
Univerze v Mariboru

DOI
https://doi.org/
10.18690/um.feri.1.2024.7

ISBN
978-961-286-837-6

Instantaneous Vehicle Image Analysis With Deep Learning Methods With Flutter Framework

Aleksandr Shishkov,[1,3] Stevanče Nikoloski[2,3]

[1] University of Maribor Faculty of Electrical Engineering and Computer Science, Maribor, Slovenija
aleksandr.shishkov@result.si

[2] University of Novo mesto Faculty of Economics and Informatics, Novo mesto, Slovenia
stevance.nikoloski@uni-nm.si

[3] Result d.o.o., Ljubljana, Slovenia
aleksandr.shishkov@result.si, stevance.nikoloski@uni-nm.si

Keywords:
MobileNetV3,
Flutter,
image classification,
fine-tuning,
deep learning

The article explores the integration of the MobileNetV3 model into the Flutter framework, focusing on advanced image classification of vehicles. We examine a multi-faceted approach that includes the use of diverse datasets, fine-tuning the model, and seamlessly implementing it into a mobile application. With an emphasis on enhancing the user experience, we have created three specialized models with a high level of accuracy (97%), capable of recognizing appropriate images, classifying the type of image (vehicle, dashboard, or document), and determining the side of the vehicle (front, left, right, rear). The results demonstrate exceptional speed and responsiveness of the application, with MobileNetV3 providing precise classification in just 60 ms, contributing to the overall efficiency of the system.

University of Maribor Press

1 Uvod

Umetna inteligenca in globoko učenje sta revolucionirala našo sposobnost optimizacije in avtomatizacije raznovrstnih nalog. Eden od zanimivih izzivov na tem področju je klasifikacija slik. Naloga strojnega učenja, ki je temu namenjena se imenuje večciljna klasifikacija (angl. Multi-label classification), pri čemer za vsak nov klasifikacijski objekt, model vrača vrednosti med 0 in 1 za vsako posamezno kategorijo, odvisno od izračunane verjetnosti, da je v novem klasifikacijskem objektu prepoznana posamezna kategorija (de Carvalho & Freitas, 2009). V tem članku se bomo posvetili uporabi večciljnega klasifikacijskega modela MobileNetV3 (Qian et al., 2021). Osredotočili se bomo na njegovo uporabo pri validaciji ustreznih slik, klasifikaciji v treh različnih kategorijah, kot so armaturna plošča, vozilo ali dokument ter klasifikaciji strani avtomobilov, pri čemer bomo vodili skozi postopek integracije naučenega modela v aplikacijo, zgrajeno na ogrodju Flutter (Singh & Bhadani, 2020).

MobileNetV3 predstavlja vrhunski klasifikacijski model naučen s konvolucijskimi nevronskimi mrežami (CNN). Medtem ko obstaja več algoritmov strojnega učenja za klasifikacijo slik, kot so logistično regresijo, odločitvena drevesa, naključni gozdovi ipd., smo se odločili za uporabo CNN, saj raziskava v področju strojnega učenja modele CNN izloči kot najbolj zanesljivi, saj pričakujemo visoko stopnjo natančnosti modela za klasifikacijo slik (Wang et al., 2021).

Podatki predstavljajo temeljno komponento sodobnega sveta. V našem članku bomo podrobno opisali postopek pridobivanja podatkov, nato pa razkrili, kako smo jih skrbno obdelali, upoštevajoč specifične potrebe našega projekta. Pri tem smo se srečali z raznolikimi izzivi, ki smo jih sistematično naslovili in premagali, kar je ključno za zagotavljanje kakovostnih rezultatov našega modela. Slednje smo dosegli, ko smo skupaj s slikami, ki smo jih pridobili preko naše aplikacije vključili še slike iz zunajnih relevantnih virov.

Pomembno je poudariti, da uspešno obvladovanje podatkovne faze prispeva k celotni zanesljivosti in natančnosti modela ter posledično izboljšuje uporabniško izkušnjo. Z našim pristopom k pridobivanju in obdelavi podatkov želimo bralcem ponuditi vpogled v kompleksnost ter pomembnost pravilnega ravnanja s podatki v procesu razvoja in izpopolnjevanja modelov umetne inteligence.

MobileNetV3, ki deluje kot naš osnovni model, gre skozi proces fino prilagajanja (fine-tuning), da bi izboljšal svojo učinkovitost pri specifičnem opravilu klasifikacije strani avtomobilov. Ko je model enkrat prilagojen, sledi ključen korak brezhibne integracije v aplikacijo Flutter (Singh, 2020). Poglobili se bomo v zapletenosti te integracije, poudarili korake, ki smo jih sprejeli, da bi model brez težav vključili v arhitekturo aplikacije.

2 Metodologija

Kot osnovni programski jezik smo izbrali Python, ki velja za klasično izbiro pri globokem in strojnem učenju (angl. deep learning in machine learning). Pomagala nam je knjižnica TensorFlow (Pang et al., 2020), s katero smo naučili in testirali model, obdelali podatki ter konvertirali naučeno model v format TensorFlow Lite, optimiziran za uporabo na mobilnih napravah.

2.1 Priprava podatkov

Na začetku našega projekta smo imeli v mislih razvoj modela za napoved strani vozila. Zato smo se odločili uporabiti nabor podatkov CompCars (Yang et al., 2015), ki predstavlja obsežno kolekcijo slik avtomobilov in njihovih delov. Naša prvotna zahteva je bila, da morajo biti slike avtomobilov v naboru podatkov označene glede na stran vozila, kot so spredaj, levo, desno ali zadaj. Na žalost smo ugotovili, da ta zbirka ne vsebuje potrebnih označb. Z namenom, da bi ohranili učinkovitost projekta, smo se soočili z izzivom, saj bi ročno označevanje kar 136 tisoč slik predstavljalo nerealno obsežno opravilo. Zato smo sprejeli odločitev, da poskusimo drug nabor podatkov, ki bi ustrezal našim zahtevam.

Posledično smo se odločili tudi klasificirati kategorijo slik, ki jih lahko uporabnik posname, in sicer ali gre za armaturno ploščo, vozilo ali dokument. Za ta namen smo uporabili slike iz produkcijskega strežnika, saj projekt že deluje približno dve leti in vsebuje zadostno količino slik. Prednost te izbiri je v tem, da posebne lastnosti in format teh slik bo povečata točnost modela, ker so podatki za treniranje in za nadaljnjo uporabo podobni. Te surove slike so za naš primer zelo ustrezne, vendar sploh niso označene in zato smo se lotili dodatnega označevanja.

Za ta namen smo razvili aplikacijo z grafičnim vmesnikom (GUI), ki omogoča ročno klasifikacijo slik. Pri razvoju smo mislili na povečanje hitrosti ročne klasifikacije, zato smo implementirali klasifikacijo z uporabo hitrih tipk (angl. hotkeys) in dinamično izbiro kategorij in map slik. Ta rešitev je poenostavila ročni postopek klasifikacije slik in zagotovila ustvarjanje prilagojenega podatkovnega nabora iz surovih slik.

Takrat smo že začeli delati z MobileNetV3, in na srečo ta model ni zahteval velikega nabora podatkov pri finem prilagajanju in prav tako ni bilo vpliva na točnosti. Zato smo označili manj slik in trenirali model na majhnem naboru podakov, preostali slike pa smo označili že z uporabo modela.

Prav tako smo za implementacijo funkcionalnosti validacije morali pripraviti dve kategoriji v naboru podatkov: ustrezne in neustrezne slike. Kot ustrezne slike smo uporabili slike iz prejšnjega nabora podatkov, medtem ko smo za neustrezne slike uporabili nabor podatkov iz spletne strani Kaggle (Sinha, 2024), ki vsebuje več odprtodostopnih slik poljubnih kategorij.

Smo ustvarili tri vrste podatkovnih zbirk za učenje modela za klasifikacijo strani avtomobila, model za klasifikacijo tipa slik in validacijski model.

Za vsako kategorijo imamo med 3000 in 5000 slik, pri čemer pri treniranju uravnotežimo količino. Kljub velikemu številu slik lahko pri finem prilagajanju dejansko uporabimo manjše število, saj razlika v točnosti ni tako velika. Smo dosegli natančnost več kot 90% ali 95% pri 100 slikah na kategorijo.

2.3 Naloga strojnega učenja

Naloga modela strojnega učenja je vključevala predvsem večciljno klasifikacijo vrste slike, tj. dodelitev slike v določeno kategorijo iz nabora različnih kategorij. Za izpolnitev te naloge smo razvili arhitekturo modela, ki temelji na MobileNetV3 in je bila fino prilagojena specifičnim zahtevam našega projekta. Enoten model je bil oblikovan za izvajanje natančnih napovedi, pri čemer smo se osredotočili na prilagajanje končnih slojev glede na značilnosti naših podatkov. S tem smo ustvarili model, ki je sposoben učinkovito klasificirati slike v različne kategorije, kar je ključno za uspešno izvajanje naše aplikacije.

2.3.2 Večciljna klasifikacija

Večciljna klasifikacija (angl. multi-label classification) je algoritem strojnega učenja, ki se uporablja za napoved več kot ene kategorije oz. oznake za dano vhodno instanco (de Carvalho & Freitas, 2009).

V primeru večciljne klasifikacije imamo opravka s posebnim izzivom, imenovanim binarna relevantnost (Szymański & Kajdanowicz, 2017; Binary Relevance, 2024). Ta koncept se nanaša na napovedovanje verjetnost prisotnosti posamezne kategorije na vhodni sliki, pri čemer model vrača vrednosti med 0 in 1. Kot prag napovedane vrednosti se pogosto nastavlja 0.5, da lahko določimo, ali slika spada v posamezno kategorijo ali ne. Ta pristop omogoča osredotočanje na posamezne kategorije neodvisno ena od druge, kar je ključno pri nalogah, kjer želimo ločeno obravnavati več kategorij za dano instanco. Slika 1 prikazuje nabor slik, opisanih z vizualnimi značilnostmi, izluščenimi iz slik, kot so število črt različnih tipov (ukrivljene, horizontalne, diagonalne itd.), in ciljne atribute, ki označujejo prisotnost (T) ali odsotnost (⊥) objektov različnih tipov.

Image	Descriptive attributes				Target attributes						
	\	∫	—	...	*traffic light*	*car*	*truck*	*building*	*traffic sign*	*bridge*	*tree*
	32	3	54	...	T	T	T	T	T	⊥	T
	55	43	1	...	T	T	T	T	T	T	T
	23	4	5	...	T	T	⊥	T	T	⊥	T
	23	4	5	...	T	T	T	T	⊥	⊥	T
	21	2	4	...	T	T	⊥	⊥	T	⊥	T
...	...	...	...	...	...	...	...	...	...	...	...

Slika 1: Ilustrativni primer podatkovne zbirke z večciljno klasifikacijo.
Vir: lasten

2.4 Globoko učenje

Globoko učenje (angl. deep learning) predstavlja področje strojnega učenja, ki se osredotoča na uporabo kompleksnih umetnih nevronskih mrež z več plastmi, znanih kot globoke nevronske mreže (LeCun et al., 2015). Glavni cilj globokega učenja je avtomatsko pridobivanje in predstavljanje kompleksnih vzorcev v podatkih. Ključna značilnost tega pristopa je njegova sposobnost samodejnega učenja hierarhičnih in abstraktnih značilnosti iz podatkov na različnih ravneh kompleksnosti.

V primerjavi s tradicionalnimi metodami strojnega učenja, kjer so modeli običajno plitvi, globoko učenje omogoča učenje predstavitev podatkov na več ravneh. To omogoča boljše zajemanje in razumevanje hierarhične strukture podatkov. Temeljni gradniki globokih nevronskih mrež vključujejo vhodni sloj, skrite sloje in izhodni sloj. Skriti sloji igrajo ključno vlogo, saj omogočajo modelu, da samodejno izpelje značilnosti iz podatkov na bolj abstraktnih ravneh, kar vodi do učinkovitega reprezentiranja kompleksnih vzorcev. Ključna pomankljivost modele globokega učenja je njihova interpretivnost. Zato so modele globokega učenja imenujejo tudi »black-box« modele. Vendar so zelo uporabni in aplikativni zaradi njihove velike natančnosti.

2.4.1 Konvolucijska nevronska mreža

Konvolucijska nevronska mreža (angl. Convolutional Neural Network ali CNN) je vrsta nevronske mreže zasnovana obdelave podatkov, ki imajo strukturo mreže, kot so slike (O'Shea & Nash, 2015). CNN se pogosto uporablja v nalogah računalniškega vida, kot so klasifikacija slik, detekcija objektov in segmentacija slik. Glavna značilnost CNN je uporaba konvolucijskih plasti, ki omogočajo učinkovito prepoznavanje lokalnih vzorcev v vhodnih podatkih.

Pri tradicionalnih algoritmih strojnega učenja za večciljno klasifikacijo se pomembne značilke izbirajo v predpripravi podatkov in postanejo vhodni podatki v učnem procesu. Po drugi strani pa je ključna prednost konvolucijske nevronske mreže v tem, da namesto ročnega izbiranja značilk iz vhodnih podatkov v fazi predpriprave podatkov, model samodejno izpelje pomembne značilke s pomočjo konvolucijskih plasti, kar poteka skozi proces, imenovan inženiring značilk (ang. feature engineering), ki je vključen v samem učnem procesu. Konvolucijske plasti

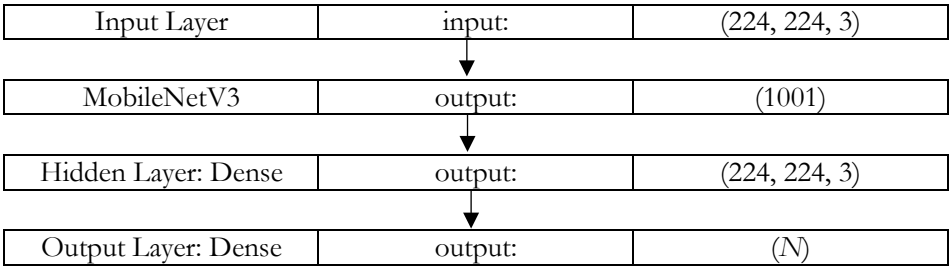
uporabljajo jedra (filtri), ki se pomikajo po vhodnih podatkih in izvajajo konvolucijo, kar omogoča prepoznavanje lokalnih vzorcev, kot so robovi, tekstura in oblike (Slika 2) (Siebel & Rice, 2019).

Poleg konvolucijskih plasti CNN običajno vključuje tudi plasti združevanja (pooling), ki zmanjšajo dimenzionalnost podatkov in povečajo prejemanje globalnih značilnosti. CNN ima običajno tudi polno povezane sloje, ki se uporabljajo za končno klasifikacijo ali regresijo.

Zaradi njihove uspešnosti pri obdelavi slik in drugih mrežnih podatkov so CNN postale ključna tehnologija na področju računalniškega vida.

Pri prvih poskusih smo probali zgraditi modelo z lastno napisano arhitekturo, podobno kot v dokumentaciji TensorFlow-a (TensorFlow, 2024), ki je vsebovala več slojev in je bila prazna. Po treniranju in prilagojenju parametrov modela nismo dosegli točnosti več kot 65%. Zato smo prenehali z uporabo tega modela in začeli iskati že modele naučene na velikem datasetu.

V našem primeru model za napoved več kategorij sledi naslednji arhitekturi: na vhodu sprejme podatke sliki z ločljivostjo 224 na 224 pikslov, slika je v RGB formatu. Uporablja se MobileNetV3 kot osnovni model, ki je že predhodno treniran in ima lastne uteži. Sledi skrit sloj Dense, ki ga nadalje treniramo. Končno, kot izhod služi zadnji sloj Dense, ki ima N enot, kjer je N število kategorij v naboru podatkov, kot je prikazano na Slika 2.



Slika 2: Arhitektura modela za večciljno klasifikacijo.

Vir: lasten.

2.4.2 MobileNetV3

MobileNetV3 je model nevronske mreže, zasnovan za klasifikacijo slik in druge naloge v področju računalniškega vida (Qian et al., 2021). Gre za tretjo iteracijo MobileNet modelov, ki so znani po svoji sposobnosti doseganja visoke natančnosti klasifikacije ob hkratni ohranitvi nizke računske zahtevnosti. Glavni cilj MobileNetV3 je izboljšati učinkovitost in natančnost predhodnih modelov.

Model uporablja arhitekturo nevronske mreže, ki vključuje konvolucijske plasti, aktivacijske funkcije, plasti normalizacije po serijah (angl. Batch Normalization), in globalno povprečenje. Značilnost MobileNetV3 je tudi uporaba blokov Inverted Residuals, ki omogočajo boljše zajemanje informacij in povečanje zmogljivosti.

MobileNetV3 je bil posebej zasnovan za mobilne naprave in aplikacije z omejenimi računskimi viri, kjer je pomembno ohraniti ravnovesje med natančnostjo in hitrostjo delovanja. Zato je priljubljen v aplikacijah, ki zahtevajo hitro obdelavo slik ob minimalni porabi energije.

2.5 Uporabniški primeri

Sedaj, ko imamo arhitekturo modela na osnovi MobileNetV3, lahko te modele treniramo z različnimi podatki. V našem primeru želimo ustvariti tri različne modele, vsakega posebej treniranega na ločenih datasetih, z namenom uporabe v specifičnih situacijah.

2.5.1 Validacijski model

Validacijski model se osredotoča na napovedovanje ustreznosti slike in vrača vrednosti med 0 in 1. Pri odločanju o ustreznosti uporabljamo prag, v našem primeru je to 0.5, kar predstavlja koncept binarne relevantnosti. Ta model je posebej prilagojen, kadar želimo napovedati le eno kategorijo.

2.5.2 Model za klasifikacijo tipa slike

V tem primeru imamo tri kategorije, in sicer armaturno ploščo, vozilo in dokumente. Model vrne verjetnost za vsako kategorijo, kar predstavlja klasičen primer večciljni klasifikacije.

2.5.3 Model za klasifikacijo strani avta

V tej situaciji imamo štiri kategorije, ki predstavljajo različne strani avtomobila, spredaj, levo, desno, zadaj. Model prav tako vrne verjetnost za vsako kategorijio, kar predstavlja še en primer večciljne klasifikacije.

2.6 Treniranje modelov

V Jupyter notebook smo implementirali fino prilagajanje, podporo spremembam parametrov, spremembo datasetov, prikazovanje rezultatov treniranja, testiranje, shranjevanje modela in konvertacijo v format TensorFlow Lite.

Učili smo modele na lokalnih računalnikih, opremljenih z NVIDIA GPUjev in podporo za CUDA. S pravilnimi nastavitvami TensorFlow zazna GPU in trenira nanj. V tem primeru je celoten proces zelo hiter, in en model se lahko izuči v 5 do 10 minutah.

Naš pristop omogoča tudi prilagajanje parametrov, kjer lahko izbiramo med velikostjo osnovnega modela (small ali large) ter vrednostjo za depth multiplier (0.75 ali 1). Vsi naši modeli so bili trenirani z uporabo modela small in depth multiplier 0.75, saj nismo zaznali bistvenih razlik med drugimi kombinacijami. Nato sledi izbira dataseta in razdelitve, kjer smo določili 70% za učni nabor in 30% za validacijo. Število epoh smo nastavili na 10. Uporabili smo Adam optimizer in Binary Crossentropy loss za model enojne klasifikacije (validacijski model), medtem ko smo za večciljno klasifikacijo uporabili Sparse Categorical Crossentropy.

2.7 Postavitev modela v mobilno aplikacijo

Naša aplikacija je napisana na platformi Flutter, okviru za izdelavo večplatformskih aplikacij. Flutter ima repozitorij paketov, kje razvijalci lahko dodelijo svojo kodo. Za delo z Tensorflow modeli smo se odločili uporabiti paket tflite_flutter (Flutter, 2024), ki prinaša številne prednosti. Ena izmed njih je hitrost, saj paket uporablja kompilirano C kodo, kar prispeva k izjemni učinkovitosti pri obdelavi slikovnih napovedi.

Z uporabo tega paketa in osnov objektno orientiranega programiranja smo ustvarili vmesnik, kjer lahko enostavno dodamo model in ga uporabimo v aplikaciji. Ta pristop nam zelo olajša delo, saj imamo tri različne modele z različnimi kategorijami.

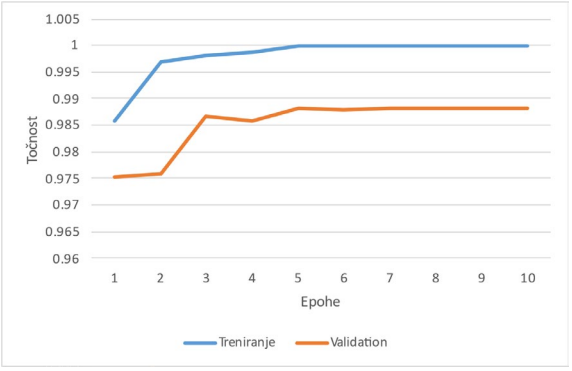
Pri izdelavi vmesnika smo pazili na pravilen prenos slikovnih podatkov v model, saj mora biti enak kot pri učenju. Zato smo poudarili testiranje pretvorbe sliki v vhodne podatke. Preverjali smo, ali je pretvorjena slika enaka sliki pri učenju, če so bile originalno podane.

Model uporabimo pri fotografiranju sliki avtomobila. Ko uporabnik klikne na gumb, aplikacija posname sliko, jo spremeni v pravo velikost in konvertira v vektor pikslov. Ta vektor je vhod za naš model, ki nam vrne rezultat inference.

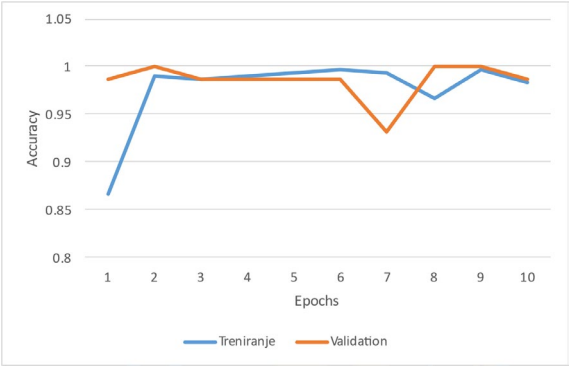
Ta model nam omogoča izboljšanje uporabniškega vmesnika v dveh situacijah. Prvič, validira sliko in preveri, ali je ustrezna; če ne, obvesti uporabnika. Če je slika ustrezna, jo klasificira, doda oznako in prikaže v uporabniškem vmesniku.

3 Rezultati

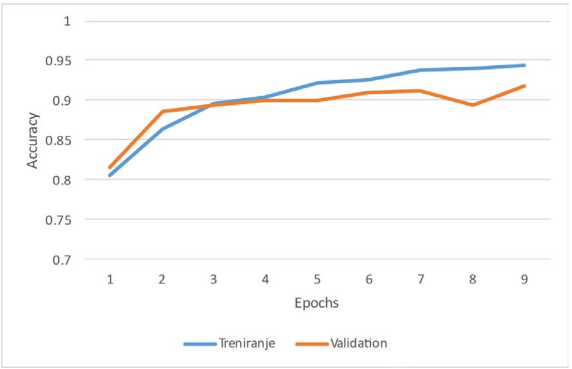
Po uspešnem učenju modela na osnovi MobileNetV3 in izvedbi fino prilaganja za klasifikacijo slik smo dosegli opazen napredek v več pogledih. Rezultati učenja treh modelov so ustrezno prikazani na Slika 4, Slika 5 in Slika 6.



Slika 4: Grafični prikaz rezultatov validacijskega modela.
Vir: lasten.



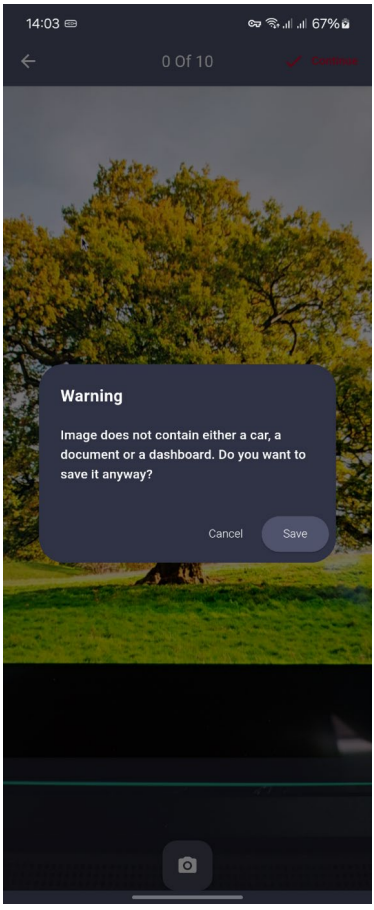
Slika 5: Grafični prikaz rezultatov modela za klasifikacijo tipa slike.
Vir: lasten.



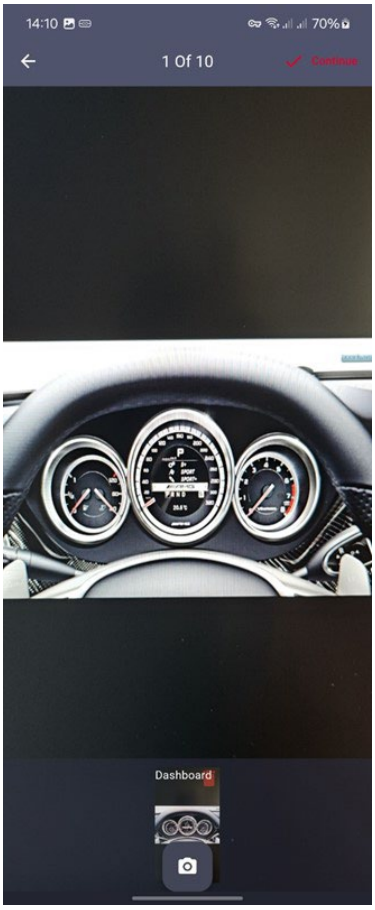
Slika 6. Grafični prikaz rezultatov modela za klasifikacijo strani vozila.
Vir: lasten.

3.1 Implementacija modelov v mobilni aplikaciji

Dodana funkcionalnost klasifikaciji v mobilni aplikaciji je bistveno izboljšala izkušnjo uporabnika. Uporabniki lahko zdaj enostavno fotografirajo avtomobile in prejmejo takojšnjo klasifikacijo, ki vključuje informacije o ustreznosti slike (Slika 7), tipu slike (vozilo, armaturna plošča ali dokument) (Slika 8) ter strani avtomobila (spredaj, levo, desno, zadaj). Napoved ne vpliva na hitrost in odzivnost aplikaciji, ker MobileNetV3 je optimiziran za mobilni napravi in srednja napoved je dolg 60 ms.



Slika 7. Slika iz mobilne aplikaciji, primer preverjanja ustreznosti sliki.
Vir: lasten.



Slika 8. Slika iz mobilne aplikaciji, primer klasificiranja vrste sliki.
Vir: lasten.

4 Zaključek

V tem članku smo uspešno predstavili uporabo modela MobileNetV3 za klasifikacijo slik v mobilni aplikaciji, zgrajeni v ogrodju Flutter. Razvili smo tri različne modele za klasifikacijo avtomobilskih slik glede na ustreznost, tip slike in stran avtomobila.

Pomemben poudarek smo namenili pridobivanju in obdelavi podatkov, ki sta ključna koraka pri razvoju modelov umetne inteligence. S sistematičnim pristopom smo premagali izzive, povezane z označevanjem slik in ustvarili kakovosten podatkovni nabor za učenje.

MobileNetV3 se je izkazal kot učinkovit model za klasifikacijo slik, saj dosega visoko natančnost ob ohranjanju nizke računske zahtevnosti. Rezultati učenja modelov so bili obetavni, kar kaže na uspešnost našega pristopa.

Implementacija modelov v mobilno aplikacijo je prinesla izboljšanje uporabniške izkušnje pri klasifikaciji avtomobilskih slik v realnem času. Hitrost delovanja modela MobileNetV3 je omogočila odzivno in učinkovito uporabo v mobilni aplikaciji.

Skupno gledano smo s tem projektom potrdili potencial uporabe umetne inteligence, zlasti globokega učenja, v praksi. Kombinacija zmogljivega modela, ustrezno pripravljenih podatkov ter integracije v uporabniški vmesnik mobilne aplikacije odpira vrata za različne aplikacije v industriji, kjer je klasifikacija slik ključnega pomena.

Viri in literatura

Binary Relevance (31.1.2024). Pridobljeno iz Scikit-multilearn: http://scikit.ml/api/skmultilearn.problem_transform.br.html (2024).

de Carvalho, A. C. & Freitas, A. A. "A tutorial on multi-label classification techniques". Foundations of Computational Intelligence Volume 5: Function Approximation and Classification, (2009): 177-195.

Flutter. (31. 1. 2024). tflite_flutter. Pridobljeno iz Pub.dev: https://pub.dev/packages/tflite_flutter (2024)

LeCun, Y., Benagio, Y. & Hinton, G. "Deep learning". Nature (2015): 436-444.

Nikoloski, S. "Structured Output Prediction and Modeling Soil Functions". Doctoral Dissertation. Jozef Stefan International Postgraduate School, Ljubljana Slovenia (2020).

O'Shea, K. & Nash, R. "An introduction to convolutional neural networks". arXiv preprint arXiv:1511.08458 (2015).

Pang, B., Nijkamp, E. & Wu, Y.N. "Deep learning with Tensorflow: A review". Journal of Educational and Behavioral Statistics (2020): 227-248.

Qian, S. N. (2021). "MobileNetV3 for Image Classification". 2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE) (str. 490-497). IEEE.

Siebel, T. & Rice, C. "Digital Transformation: Survive and Thrive in an Era of Mass Extinction". New York: Rosetta Books (2019).

Singh, A. & Bhadani, R. "Mobile Deep Learning with TensorFlow Lite, ML Kit and Flutter: Build scalable real-world projects to implement end-to-end neural networks on Android and iOS". Packt Publishing Ltd (2020).

Sinha, A. "Image Dataset". Pridobljeno iz Kaggle: https://www.kaggle.com/datasets/starktony45/image-dataset (2024).

Szymański, P. & Kajdanowicz, T. "A scikit-based Python environment for performing multi-label classification". ArXiv preprint arXiv:1702.01460 (2017).

TensorFlow. "Image classification". Pridobljeno iz TensorFlow: https://www.tensorflow.org/tutorials/images/classification (2024).

Wang, P., Fan, E. & Wang, P. "Comparative analysis of image classification algorithms based on traditional machine learning and deep learning". Pattern Recognition Letters (2021): 61-67.

Yang, L., Luo, P., Change, L.C. & Tang, X. "A large-scale car dataset for fine-grained categorization and verification". Proceedings of the IEEE conference on computer vision and pattern recognition (2015): 3973-3981.