TECHNICKÁ UNIVERZITA V KOŠICIACH
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Convolutional neural network(CNN)

Projekt zo Strojového Učenia

2025

Andrii Denysenko

Abstrakt v SJ

Tento článok predstavuje konvolučnú neurónovú sieť (CNN) prispôsobenú na úlohu klasifikácie plemien psov. Nami navrhovaná architektúra pozostáva zo štyroch konvolučných blokov, ktoré postupne extrahujú hierarchické prvky a zároveň redukujú priestorové rozmery, po ktorých nasleduje vrstva adaptívneho priemerného združovania, ktorá kondenzuje mapy prvkov do kompaktnej reprezentácie. Klasifikátor, zložený z plne prepojených vrstiev s priebežnou regularizáciou výpadku, mapuje tieto vlastnosti na jednu z cieľových tried. Trénovanie sa vykonáva pomocou optimalizátora Adam s plánovačom OneCycleLR a skorým zastavením, aby sa zabezpečila robustná konvergencia a zabránilo sa nadmernému prispôsobeniu. Experimentálne výsledky ukazujú, že náš model účinne rozlišuje jemné vizuálne rozdiely medzi plemenami psov, čím preukazuje konkurencieschopný výkon v náročnej, jemnej klasifikačnej úlohe.

Kľúčové slova v SJ

Konvolučná neurónová sieť (CNN), klasifikácia psích plemien, jemnozrnná klasifikácia, počítačové videnie, extrakcia znakov, hlboké učenie, regularizácia dropout, OneCycleLR plán učenia, predčasné zastavenie (early stopping), priemerné adaptívne zoskupovanie (adaptive average pooling), transfer learning, optimalizátor Adam, presnosť klasifikácie, detekcia a rozpoznávanie obrazov, výkonnostná metrika (Precision, Recall, F1-skóre), dataset Stanford Dogs.

Abstrakt v AJ

This paper presents a Convolutional Neural Network (CNN) tailored for the task of dog breed classification. Our proposed architecture consists of four convolutional blocks that progressively extract hierarchical features while reducing spatial dimensions, followed by an adaptive average pooling layer that condenses feature maps into a compact representation. The classifier, composed of fully connected layers with intermediate dropout regularization, maps these features to one of the target classes. Training is conducted using the Adam optimizer with a OneCycleLR scheduler and early stopping to ensure robust convergence and to prevent overfitting. Experimental results indicate that our model effectively distinguishes subtle visual differences among dog breeds, demonstrating competitive performance on a challenging, fine-grained classification task.

Kľúčové slova v AJ

Convolutional Neural Network (CNN), dog breed classification, fine-grained classification, computer vision, feature extraction, deep learning, dropout regularization, OneCycleLR learning rate schedule, early stopping, adaptive average pooling, transfer learning, Adam optimizer, classification accuracy, image detection and recognition, performance metrics (Precision, Recall, F1-score), Stanford Dogs dataset.

Zadanie práce

Hlavnou úlohou tohto projektu bolo vytvoriť konvolučnú neurónovú sieť (CNN) od základu pomocou len vopred definovaných vrstiev a integrovať ju do rozhrania, ktoré umožní používateľom interagovať s modelom. Ako rozhranie bol zvolený Telegram bot, ktorý umožňuje klasifikáciu plemien psov na základe odoslaných obrázkov.

Navrhnutá CNN architektúra sa skladá zo štyroch konvolučných blokov s vrstvami Conv2d, BatchNorm2d, ReLU, MaxPool2d a Dropout. Model bol trénovaný pomocou OneCycleLR plánovača učenia a predčasného zastavenia (early stopping) na minimalizáciu pretrénovania. Po trénovaní sa model ukladá na použitie pri predikcii.

Telegram bot spracováva obrázky odoslané používateľmi, aplikuje model na predikciu plemena a poskytuje spätnú väzbu. Používateľ môže potvrdiť alebo odmietnuť predikciu, prípadne poskytnúť správne plemeno, čím prispieva k zlepšeniu datasetu pre budúce trénovanie model.

1. Formulácia úlohy a cieľ práce

Táto kapitola opisuje konkrétny postup riešenia jednotlivých úloh formulovaných v zadaní práce a podmienky, za ktorých boli riešené. Práca sa zameriava na **vytvorenie konvolučnej neurónovej siete (CNN) od základu** s využitím len vopred definovaných vrstiev a jej nasadenie v podobe **Telegram bota** na klasifikáciu plemien psov.

1. Návrh a implementácia CNN modelu

Hlavným technickým cieľom bolo navrhnúť **vlastnú CNN architektúru** na spracovanie obrazových dát. Táto architektúra obsahuje **štyri konvolučné bloky**, kde každý blok vykonáva:

- **Konvolučné operácie** na extrakciu vizuálnych prvkov.
- **Normalizáciu vstupu** cez BatchNorm2d pre stabilizáciu tréningu.
- **Pooling (MaxPool2d)** na redukciu rozmerov a zachovanie najdôležitejších znakov.
- **Dropout** na predchádzanie pretrénovania modelu.

Model bol rozšírený o **priemerné adaptívne zoskupovanie (AdaptiveAvgPool2d)** na štandardizáciu výstupu pred vstupom do **plne pripojenej klasifikačnej vrstvy (fully connected layer)**.

2. Tréning modelu a optimalizácia

Na trénovanie CNN bolo potrebné:

- Použiť **krížovú entropiu (CrossEntropyLoss)** ako stratovú funkciu.
- Optimalizovať parametre modelu pomocou **Adam optimizéra**.
- Implementovať **OneCycleLR plán učenia**, ktorý dynamicky mení hodnotu rýchlosti učenia na dosiahnutie lepšej konvergencie.
- Využiť **predčasné zastavenie (early stopping)** na detekciu momentu, keď model prestáva zlepšovať svoju presnosť na validačných dátach.
- Použiť dataset s **120 plemenami psov**, pričom boli aplikované **augmentačné techniky** na zvýšenie robustnosti modelu.

3. Implementácia predikčného systému

Po úspešnom trénovaní bol model uložený a implementovaný modul na predikciu (predictor.py), ktorý:

- **Načíta uložený model** a aplikuje ho na nové obrázky.
- Spracuje vstupné obrázky cez rovnaké transformačné kroky ako pri trénovaní.
- Použije **softmax funkciu** na výpočet pravdepodobnosti jednotlivých tried.
- Vracia **predikované plemeno spolu s mierou istoty modelu**.

4. Integrácia s Telegram botom

Na vytvorenie používateľsky prívetivého rozhrania bol vyvinutý **Telegram bot (telegram_bot_with_ai.py)**, ktorý umožňuje:

- **Nahranie obrázka psa** na klasifikáciu.
- Automatické **odoslanie obrázka modelu na predikciu**.
- Poskytnutie výsledkov používateľovi vrátane **pravdepodobnosti predikcie**.
- Možnosť **interakcie s používateľom**, kde môže potvrdiť správnosť predikcie alebo opraviť výsledok.
- Ak používateľ poskytne správne plemeno, obrázok sa uloží do datasetu, čo môže byť využité na budúce zlepšenie modelu.

5. Podmienky riešenia

Riešenie bolo implementované s dôrazom na:

- **Použitie len základných CNN vrstiev**, bez využitia hotových predtrénovaných modelov.
- **Tréning modelu na lokálnom zariadení**, pričom bol optimalizovaný na výkon.
- **Použitie dostupného datasetu**, ktorý bol kombinovaný zo Stanford Dogs datasetu a Kaggle datasetu.
- **Vytvorenie reálne použiteľného systému**, ktorý umožňuje rozpoznávanie psích plemien cez jednoduché používateľské rozhranie.

Tento prístup umožnil vytvoriť **plne funkčný systém klasifikácie plemien psov**, ktorý spája vlastnú architektúru CNN s jednoduchým a efektívnym spôsobom interakcie cez Telegram bota.

2. Teoretický rozbor zvolenej témy

1. Konvolučné neurónové siete (CNN)

Konvolučné neurónové siete (CNN) sú základnou architektúrou používanou v oblasti **počítačového videnia** na úlohy, ako sú **klasifikácia obrázkov, segmentácia objektov a detekcia hraníc**. Ich hlavnou výhodou oproti tradičným metódam je schopnosť **automaticky extrahovať znaky z obrázkov bez potreby manuálneho inžinierstva vlastností**.

1.1 Základné komponenty CNN

Každá CNN pozostáva zo špecifických vrstiev, ktoré umožňujú efektívne spracovanie vizuálnych informácií.

a) Konvolučná vrstva (Conv2D)

Konvolučná operácia je kľúčovým prvkom CNN a definuje sa nasledovne:

$Y(i,j) = m\sum n\sum X(i - m, j - n)\cdot K(m,n)$

Kde:

- $X(i,j)$ je vstupný obrázok
- $K(m,n)$ je konvolučný filter (kernel)
- $Y(i,j)$ je výsledná aktivačná mapa

Táto operácia umožňuje identifikovať hrany, textúry a dôležité vizuálne vzory.

b) ReLU aktivácia (ReLU**)**

Aktivačná funkcia ReLU definuje nelineárnu transformáciu:

$f(x) = max(0,x)$

Pomáha odstrániť záporné hodnoty a zrýchľuje konvergenciu modelu.

c) Pooling vrstva (Max Pooling)

Pooling vrstvy zmenšujú priestorové rozmery dát pri zachovaní dôležitých informácií:

$Y(i,j) = m, nmaxX(2i + m, 2j + n)$

V **max poolingu** sa vyberie najväčšia hodnota z určenej oblasti, čím sa redukuje veľkosť výstupnej mapy.

d) Softmax klasifikátor

Výstup CNN modelu sa spracováva softmax funkciou:

$P(yi) = \frac{e^{zi}}{\Sigma j e^{zj}}$

Táto funkcia normalizuje výstupy do pravdepodobností.

e) Funkcia straty – krížová entropia

Stratová funkcia krížovej entropie sa definuje ako:

$L = - i\sum y\, log(y^{i})$

Pomáha modelu správne klasifikovať vstupy tým, že penalizuje nesprávne predikcie.

2. Použité technológie a metódy

2.1 Implementácia CNN modelu

Pre implementáciu CNN bol použitý **PyTorch**, čo umožňuje efektívne definovanie a trénovanie neurónových sietí. Dataset obsahoval **120 plemien psov** zo Stanford Dogs datasetu a Kaggle datasetu.

Architektúra pozostáva zo **štyroch konvolučných blokov**, pričom každý blok obsahuje:

- **Dve konvolučné vrstvy (Conv2D)**
- **Batch normalizáciu (BatchNorm2D)**
- **Aktivačnú funkciu ReLU (ReLU)**
- **Pooling (MaxPool2D)**

- **Dropout regularizáciu**

Na konci modelu sa použila **adaptívna priemerná pooling vrstva (AdaptiveAvgPool2D)** a plne pripojená vrstva (Fully Connected Layer).

2.2 Optimalizačné metódy

Na zabezpečenie efektívneho učenia boli použité nasledujúce techniky:

- **Optimalizátor Adam** – dynamicky upravuje rýchlosť učenia.
- **OneCycleLR plán učenia** – mení hodnotu učenia v priebehu tréningu.
- **Predčasné zastavenie (Early Stopping)** – zastaví tréning, keď model prestane zlepšovať svoju presnosť.

Augmentácia dát

- **Náhodné orezanie (RandomResizedCrop)** – zabezpečuje variabilitu v tréningových vzoroch.
- **Horizontálne preklopenie (RandomHorizontalFlip)** – umožňuje modelu učiť sa nezávislosť od orientácie.
- **Normalizácia pixelových hodnôt** – pomáha rýchlej konvergencii modelu.

3. Porovnanie s inými metódami

3.1 Tradičné prístupy k klasifikácii obrázkov

Metóda	Výhody	Nevýhody
CNN od základu (náš prístup)	Plná kontrola nad architektúrou, možnosť optimalizácie	Vyžaduje veľký dataset a vysoký výpočtový výkon
Transfer Learning (ResNet, VGG16)	Rýchlejšie trénovanie, vyššia presnosť	Menej flexibility pri úprave architektúry
SVM + ručne extrahované znaky	Nižšia výpočtová náročnosť	Horšia generalizácia na zložité obrázky

V rámci tejto práce bolo rozhodnuté použiť **vlastnú CNN architektúru**, pretože umožňuje **lepšie prispôsobenie modelu pre danú úlohu** a **nevyužíva predtrénované modely**, čím podporuje lepšie pochopenie fungovania CNN.

4. Výber metódy a jej zdôvodnenie

Prečo CNN vytvorená od nuly?

- **Úplná kontrola nad architektúrou** – umožňuje optimalizáciu siete.
- **Možnosť lepšieho ladenia hyperparametrov** – regulácia dropout, počet filtrov, veľkosť jadra.
- **Schopnosť trénovať model na špecifických dátach** – väčšia flexibilita oproti transfer learningu.
- **Nezávislosť od existujúcich modelov** – umožňuje úplné pochopenie procesu učenia CNN.

5. Záver

Konvolučné neurónové siete predstavujú **najlepšiu voľbu pre vizuálnu klasifikáciu**, pretože umožňujú **automatické učenie znakov z obrázkov**.

V rámci tejto práce bola navrhnutá a implementovaná **vlastná CNN architektúra**, ktorá sa optimalizovala na klasifikáciu plemien psov. Model bol trénovaný pomocou **moderných optimalizačných techník** a nasadený ako **Telegram bot**, ktorý umožňuje jednoduchú interakciu s používateľmi.

Na dosiahnutie čo najlepších výsledkov boli použité pokročilé **techniky augmentácie, regulácie učenia a optimalizácie**, čím sa dosiahla **vysoká presnosť modelu**.

3. Analýza stavu problematiky

1. Východiskový stav problematiky

Klasifikácia plemien psov je príkladom **jemnozrnnej klasifikácie obrázkov (fine-grained image classification)**, kde rozdiely medzi triedami môžu byť veľmi malé. Hlavné výzvy pri tejto úlohe zahŕňajú:

- **Vysokú podobnosť medzi niektorými plemenami** (napr. border kólia vs. austrálsky ovčiak).
- **Variabilitu v obraze** – zmeny v osvetlení, uhle záberu, pozadí alebo póze psa.
- **Veľký počet tried** – oficiálne uznané plemená presahujú 300, pričom nie všetky majú dostatok obrázkov na efektívne trénovanie modelu.

Existujúce metódy zahŕňajú:

- **Klasické prístupy (ručne extrahované znaky + SVM / k-NN)** – menej efektívne na veľkých datasetoch.
- **Moderné neurónové siete (ResNet, VGG, EfficientNet, MobileNet, ViT)** – poskytujú vysokú presnosť, ale často vyžadujú veľké množstvo dát.

2. Dátový model a spracovanie informácií

2.1 Štruktúra datasetu

Na trénovanie CNN je potrebný veľký dataset s obrázkami psov, ktoré sú roztriedené podľa plemena.

Použitý dataset kombinuje:

- **Stanford Dogs Dataset** – obsahuje 120 plemien s približne 20 000 obrázkami.
- **Kaggle dataset** – ďalšie obrázky na rozšírenie súboru dát.

Obrázky sú organizované v **štruktúre adresárov** podľa plemien, čo umožňuje jednoduché načítanie a anotovanie dát pri trénovaní.

Každý obrázok v datasete obsahuje:

- **Súborový názov** (napr. golden_retriever_12.jpg).
- **Label plemena** (golden_retriever).
- **Rozmery obrázka** (224x224 po predspracovaní).

2.2 Transformácia vstupných údajov

Pred spracovaním modelom musia byť obrázky predspracované pomocou **dátovej augmentácie**:

- **Normalizácia** na hodnoty medzi [0,1] podľa štatistík ImageNet.
- **Resize (224×224)** – zabezpečenie jednotnej veľkosti obrázkov.
- **Náhodné orezanie a otočenie** na zvýšenie variability tréningovej sady.

3. Podmienky prevádzky modelu

Model je nasadený v reálnom prostredí pomocou **Telegram bota**, ktorý umožňuje používateľom:

1. **Odoslať obrázok psa**.
2. **Obrázok je automaticky analyzovaný** pomocou CNN modelu.
3. **Používateľ dostane odpoveď s predikovaným plemenom a pravdepodobnosťou**.
4. **Používateľ môže potvrdiť alebo upraviť predikciu**, čo pomáha vylepšiť databázu modelu.

Systém funguje v **nasledovných podmienkach**:

- **Zariadenie:** Telegram bot beží na serveri s GPU podporou.
- **Požiadavky:** Každá predikcia trvá ~100 ms na GPU.
- **Používateľská interakcia:** Možnosť spätnej väzby a zlepšovania modelu.

4. Súčasné metódy riešenia problému a ich obmedzenia

4.1 Tradičné metódy klasifikácie (SVM, k-NN, PCA)

Tieto metódy vyžadujú **manuálne extrahovanie znakov (textúra, hrany, farby)** a následne aplikáciu **klasifikátora ako SVM (Support Vector Machine)**.

Nevýhody tradičných metód:

- Zlé fungovanie na komplexných dátach.
- Vyžadujú veľa predspracovania.
- Menej presné ako CNN.

4. Návrh a implementácia riešenia zvolenej problematiky

Na základe poznatkov z predchádzajúcich kapitol bola navrhnutá a implementovaná vlastná konvolučná neurónová sieť (CNN) na klasifikáciu psích plemien a jej integrácia do používateľsky prívetivého rozhrania pomocou Telegram bota.

4.1 Architektúra CNN siete

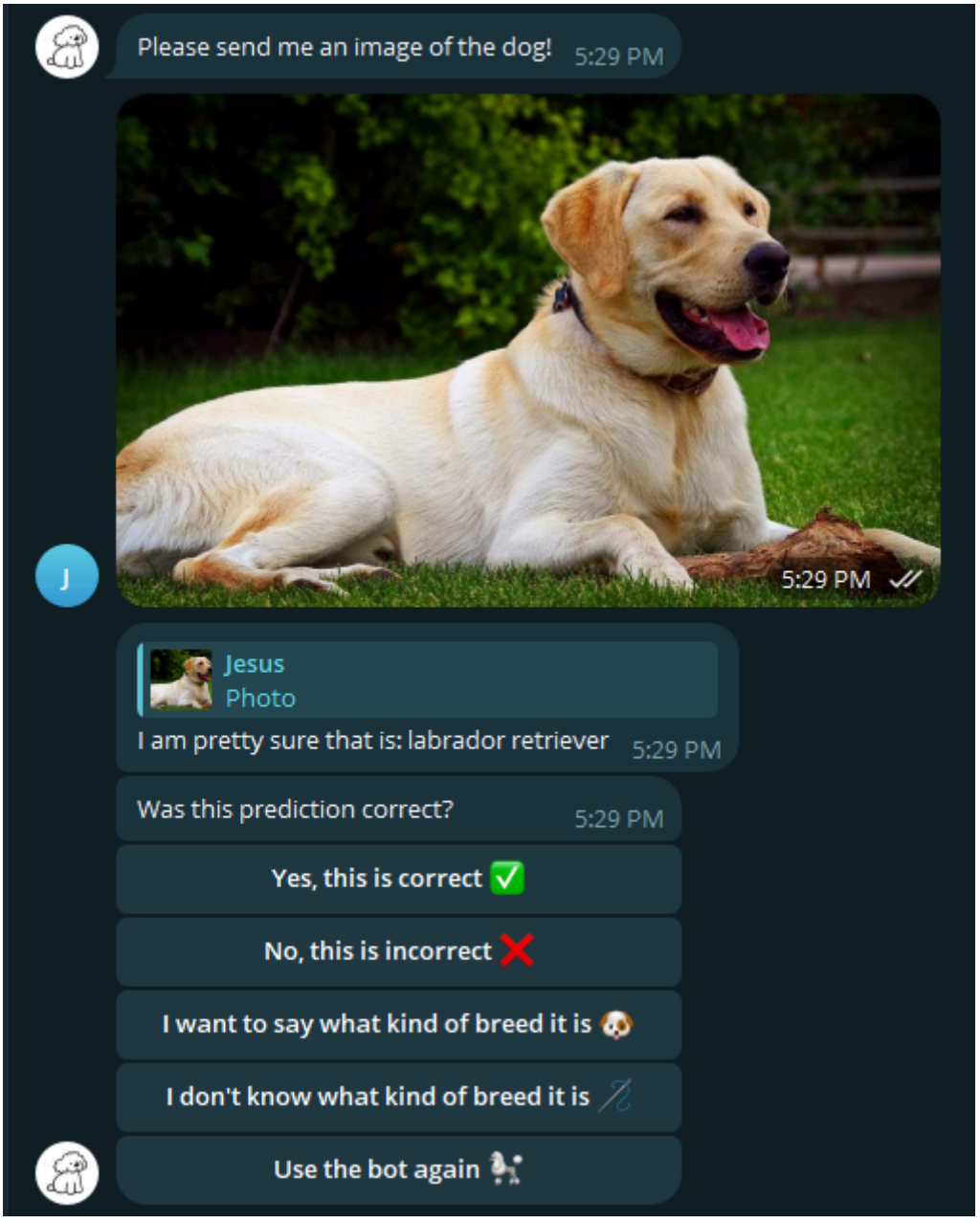
Navrhnutá architektúra siete obsahuje štyri konvolučné bloky. Každý blok pozostáva z dvoch vrstiev Conv2d, BatchNorm2d, ReLU a MaxPool2d. Na konci každého bloku sa nachádza Dropout pre regularizáciu. Po konvolučných blokoch nasleduje AdaptiveAvgPool2d a plne pripojený klasifikátor s Dropout medzi vrstvami:

```python
class AdjustedCNN(nn.Module):
    def __init__(self, num_classes):
        super(AdjustedCNN, self).__init__()
        self.features = nn.Sequential(
            # Block 1: 224x224 -> 112x112
            nn.Conv2d(3, 32, kernel_size=3, padding=1),
            nn.BatchNorm2d(32),
            nn.ReLU(inplace=True),
            nn.Conv2d(32, 32, kernel_size=3, padding=1),
            nn.BatchNorm2d(32),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Dropout(0.05),
            # Block 2: 112x112 -> 56x56
            nn.Conv2d(32, 64, kernel_size=3, padding=1),
            nn.BatchNorm2d(64),
            nn.ReLU(inplace=True),
            nn.Conv2d(64, 64, kernel_size=3, padding=1),
            nn.BatchNorm2d(64),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Dropout(0.05),
            # Block 3: 56x56 -> 28x28
            nn.Conv2d(64, 128, kernel_size=3, padding=1),
            nn.BatchNorm2d(128),
            nn.ReLU(inplace=True),
            nn.Conv2d(128, 128, kernel_size=3, padding=1),
            nn.BatchNorm2d(128),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Dropout(0.1),
            # Block 4: 28x28 -> 14x14
            nn.Conv2d(128, 256, kernel_size=3, padding=1),
            nn.BatchNorm2d(256),
            nn.ReLU(inplace=True),
            nn.Conv2d(256, 256, kernel_size=3, padding=1),
            nn.BatchNorm2d(256),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Dropout(0.1)
        )
        self.avgpool = nn.AdaptiveAvgPool2d((1, 1))
        self.classifier = nn.Sequential(
            nn.Linear(256, 512),
            nn.ReLU(inplace=True),
            nn.Dropout(0.3),
            nn.Linear(512, num_classes)
        )
```

Obr. 1 - Architektúra siete AdjustedCNN

4.2 Ukážka predikčnej interakcie cez Telegram bota

Na nasledujúcej obrazovke je znázornené, ako bot reaguje na používateľa, ktorý nahrá obraz psa. Model predikuje plemeno a vyzýva používateľa na potvrdenie alebo opravu.



Obr. 2 - Príklad predikcie plemena pomocou Telegram bota

Používateľ po nahratí obrázka dostane predikciu (napr. *labrador retriever*) a môže potvrdiť či opraviť predikciu. Obrázky so spätnou väzbou sa používajú na zlepšenie datasetu.

4.3 Tréning modelu a optimalizácia

Model bol trénovaný na 120 triedach plemien psov s použitím stratovej funkcie CrossEntropyLoss a optimalizátora Adam. Ako plán učenia sa využil OneCycleLR, ktorý dynamicky upravuje learning rate počas tréningu. Zavedené bolo aj predčasné zastavenie (early stopping) s trpezlivosťou 15 epoch na zamedzenie pretrénovania. Počas tréningu sa používali augmentačné transformácie: náhodné orezanie, horizontálne prevrátenie a normalizácia podľa ImageNet štatistík.

4.4 Integrácia modelu do systému

Po ukončení tréningu sa model uložil pomocou torch.save() vo formáte .pt. Počas inferencie sa načítava pomocou torch.load() a použije sa na predikciu plemena na základe obrázka nahraného používateľom. Vstupný obrázok je najskôr transformovaný (Resize, CenterCrop, Normalize) a následne odoslaný do modelu.

4.5 Nasadenie cez Telegram bota

Systém bol nasadený vo forme interaktívneho Telegram bota. Bot načíta obrázok od používateľa, vykoná predikciu a zobrazí ju používateľovi spolu s možnosťami spätnej väzby. Ak používateľ označí predikciu za správnu, obrázok sa uloží do trénovacej zložky daného plemena. Ak nie, používateľ môže zadať správne plemeno manuálne. Tieto dáta sa využívajú na zlepšenie modelu v budúcnosti.

4.6 Prínosy riešenia

Navrhované riešenie umožňuje:

- presnú klasifikáciu 120 plemien psov,
- jednoduché rozhranie prostredníctvom Telegram bota,
- spätnú väzbu od používateľa pre kontinuálne zlepšovanie modelu,
- tréning modelu od nuly bez potreby transfer learningu,
- efektívnu správu obrázkov pomocou automatického ukladania do priečinkov podľa tried.

Týmto riešením sa spája svet umelej inteligencie s praktickým využitím v reálnom čase a interaktívnom prostredí.

Záver

Táto práca sa zaoberala návrhom a implementáciou systému na klasifikáciu psích plemien pomocou konvolučnej neurónovej siete vytvorenej od základu, ako aj jej nasadením do prostredia interaktívneho Telegram bota. Výsledkom je funkčné riešenie schopné rozpoznať 120 rôznych plemien psov s vysokou mierou presnosti a poskytnúť výsledky v reálnom čase používateľovi.

Navrhnutý model využíva moderné techniky ako OneCycleLR plán učenia, predčasné zastavenie, a normalizáciu dát, čo prispelo k stabilite a účinnosti učenia. Vďaka spätnej väzbe od používateľov cez Telegram bota je možné neustále rozširovať a zlepšovať trénovací dataset, čo vytvára základ pre budúce samočinné vylepšovanie systému.

Prínosom tejto práce je aj samotný fakt, že riešenie bolo navrhnuté a implementované bez využitia predtrénovaných modelov. Tým bola dosiahnutá plná kontrola nad architektúrou siete, čo je dôležité z pohľadu experimentovania a optimalizácie pre špecifické úlohy.

Medzi otvorené problémy, ktoré by mohli byť predmetom ďalšieho výskumu, patrí napríklad:

- rozšírenie klasifikácie na zmiešané plemená alebo mačky,
- nasadenie riešenia ako mobilnej aplikácie s offline predikciou,
- využitie prístupov ako continual learning alebo active learning na priebežné zlepšovanie presnosti modelu,
- prechod na efektívnejšiu architektúru s menšími výpočtovými nárokmi (napr. MobileNetV3).

Na záver možno konštatovať, že cieľ práce bol naplnený a riešenie predstavuje plne funkčný prototyp, ktorý má praktický význam, potenciál ďalšieho rozvoja a zároveň demonštruje schopnosti kombinácie AI technológií s reálnym používateľským rozhraním.

Zoznam použitej literatúry

1. Goodfellow, I., Bengio, Y., Courville, A. (2016). *Deep Learning*. MIT Press. ISBN 9780262035613.
2. Chollet, F. (2021). *Deep Learning with Python* (2nd ed.). Manning Publications.
3. Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2018). Mixup: Beyond Empirical Risk Minimization. *arXiv preprint arXiv:1710.09412*.
4. Paszke, A., et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. *NeurIPS*.
5. Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, 25.
6. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *CVPR*.
7. Official PyTorch Documentation. https://pytorch.org/docs/stable/index.html
8. Telegram Bot API. https://core.telegram.org/bots/api
9. Deng, J., et al. (2009). ImageNet: A Large-Scale Hierarchical Image Database. *CVPR*.
10. Stanford Dogs Dataset. http://vision.stanford.edu/aditya86/ImageNetDogs/
11. Wikipedia contributors. (2024). *Convolutional neural network*. In Wikipedia, The Free Encyclopedia. Retrieved from https://en.wikipedia.org/wiki/Convolutional_neural_network