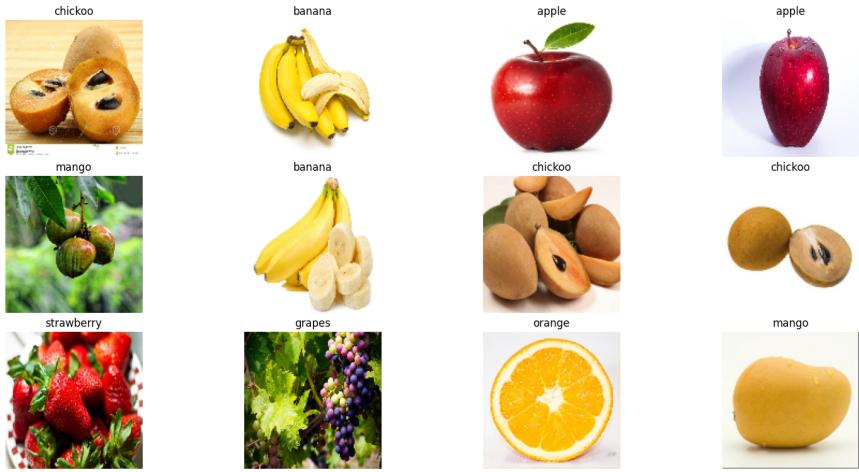


Fakulta matematiky, fyziky a informatiky
Univerzita Komenského v Bratislave

Projekt zo strojového učenia



Obr. 1: Ukážka vstupných dát.

1 Úvod

V tomto projekte sa zameriavame na návrh architektúry založenej na konvolučných neurónových sietiach, ktorá dokáže z obrázka automaticky určiť druh ovocia a zároveň odhadnúť jeho množstvo. Počet ovocia je klasifikovaný do kategórií: *one*, *two*, *three* alebo *many*. Dáta pochádzajú zo stránky

<https://www.kaggle.com/datasets/shreyapm/her/fruits-dataset-images>. Týmto priznávam, že som použil chatgpt 5.2 na tvorbu doc-stringov a niektorých funkcií v súbore *vizualize.py*.

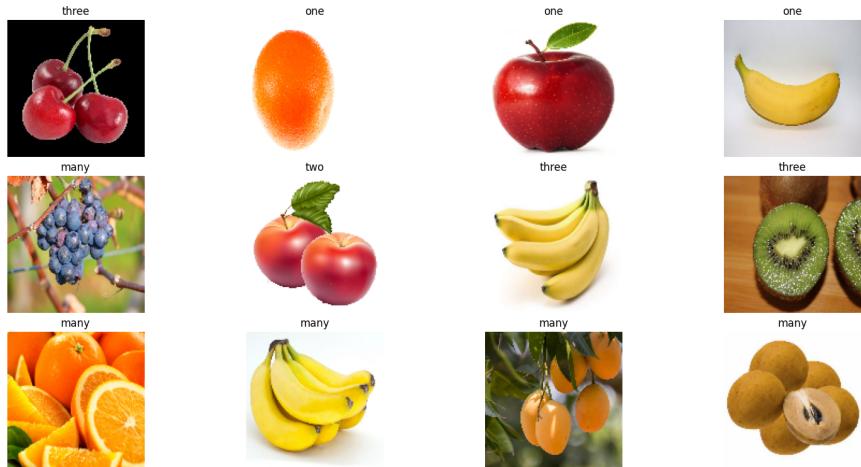
2 Dáta

Pôvodný dataset bol organizovaný hierarchicky do štruktúry, kde každý podpriečinok reprezentoval jednu triedu ovocia. Štruktúra dát je znázornená v nasledujúcim výpisom:

```
data/
|-- fruits_category/
|   |-- apple/
|   |-- banana/
|   |-- cherry/
|   |-- chickoo/
|   |-- grapes/
|   |-- kiwi/
|   |-- mango/
|   |-- orange/
|   |-- strawberry/
```

Dataset obsahoval celkovo 9 rôznych druhov ovocia, pričom pre každý druh bolo pôvodne k dispozícii 40 obrázkov. Celkový počet pozorovaní tak predstavoval 360 snímok. Ukážka vstupných dát je zobrazená na obrázku 1.

Po detailnejšej kontrole datasetu sa ukázalo, že niektoré obrázky boli takmer identické (duplicítne) alebo neboli vhodné pre účely tohto projektu, najmä v kontexte predikcie



Obr. 2: Ukážka manuálne klasifikovaných počtov ovocia.

počtu objektov na obrázku. Z tohto dôvodu boli problematické vzorky z datasetu odstránené. Po tejto filtrácii zostalo v datasete 343 obrázkov.

Kedže cieľom projektu nebola len predikcia druhu ovocia na obrázku, ale aj odhad počtu jednotlivých kusov, pôvodný dataset bolo potrebné rozšíriť o túto informáciu. Kedže pôvodné dátá neobsahovali anotácie týkajúce sa počtu objektov, bolo nutné tieto anotácie vytvoriť manuálne. Vzhľadom na výraznú variabilitu počtu ovocia na jednotlivých obrázkoch sme sa rozhodli úlohu zjednodušiť a rozdeliť obrázky do štyroch diskrétnych kategórií: **one**, **two**, **three** a **many**. Výsledná štruktúra dát teda vyzerá nasledovne:

```
data/
|-- fruits_category/
|   |-- apple/
|   |-- banana/
|   |-- cherry/
|   |-- chickoo/
|   |-- grapes/
|   |-- kiwi/
|   |-- mango/
|   |-- orange/
|   |-- strawberry/
|-- fruits_count/
    |-- one/
    |-- two/
    |-- three/
    |-- many/
```

Treba poznamenať, že na niektorých obrázkoch šlo o subjektívne rozhodnutie, do ktorej kategórie treba obrázok zaradiť. Príklad takto manuálne klasifikovaných snímok ovocia môžeme vidieť na obrázku 2.

3 Klasifikátor druhu ovocia

Cieľom je vytvoriť klasifikátor, ktorý ako vstup zoberie obrázok a ako výstup vyberie jeden prvok z množiny

$$\mathcal{C}_{\text{fruit}} = \{\text{apple}, \text{banana}, \text{cherry}, \text{chickoo}, \text{grapes}, \text{kiwi}, \text{mango}, \text{orange}, \text{strawberry}\}.$$

Našim hlavným nástrojom je knižnica Keras. Vstupné dátá najprv načítame pomocou knižnice cv2 a každý obrázok znormálizujeme na jednotný rozmer 128×128 pixelov. Po filtračii datasetu tak pracujeme s 343 pozorovaniami tvaru $128 \times 128 \times 3$ a s vektorom cieľových tried dĺžky 343. Jednotlivé kategórie z množiny $\mathcal{C}_{\text{fruit}}$ označíme celočíselne $0, \dots, 8$ (v abecednom poradí).

Ako základ modelu používame *transfer learning*, teda predtrénovanú konvolučnú neurónovú sieť. Konkrétnie využívame architektúru MobileNetV2.

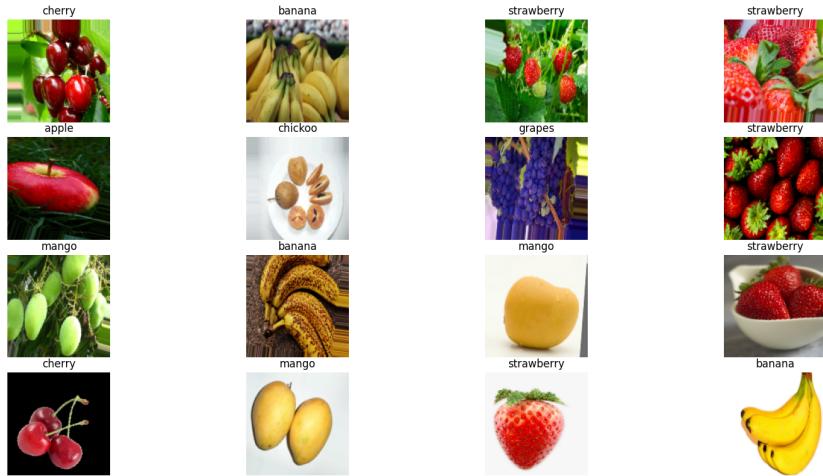
```
base_model = keras.applications.MobileNetV2(
    input_shape=(128, 128, 3),
    include_top=False
)
base_model.trainable = False

model = Sequential([
    base_model,
    GlobalAveragePooling2D(),
    Dense(
        128,
        activation="tanh",
        kernel_regularizer=keras.regularizers.l1_l2(1e-3, 1e-3)
    ),
    Dropout(0.3),
    Dense(9, activation="softmax"),
])
```

Pomocou parametra *input_shape* definujeme veľkosť obrázku do vstupnej vrstvy. Ďalej pomocou *include_top=False* vynechávame pôvodnú klasifikačnú vrstvu a ponechávame len „telo“ tejto architektúry ako „features extractor“. Keďže náš dataset je pomerne malý, nadstavujeme aj parameter *.trainable=False*, teda zamrazíme už predtrénované časti tejto siete a trénujeme len nami pridanú časť. Na výstup konvolučnej časti nadvázuje vrstva *GlobalAveragePooling2D*, ktorá agreguje priestorové príznaky do jedného vektora. Následne používame plne prepojenú vrstvu s 128 neurónmi a aktivačnou funkciou *tanh*; jej váhy sú regularizované pomocou kombinácie L1/L2 regularizácie, čo znižuje riziko overfittingu. Ďalej aplikujeme *Dropout* s hodnotou 0.3, teda počas tréningu vypneme 30% neurónov ako ďalší nástroj ako predísť overfittingu. Posledná vrstva je *Dense(9, softmax)*, ktorej výsledok je pravdepodobnostné rozdelenie nad 9 triedami ovocia; ako výslednú predikciu následne zoberieme argument maxima týchto pravdepodobností.

Taktiež počas tréningu používame aj early stopping, teda kontrolujeme validačnú accuracy nasledovne:

```
early_stop: EarlyStopping = EarlyStopping(
    monitor="val_accuracy",
    patience=5,
    restore_best_weights=True,
```



Obr. 3: Ukážka augmentovaného ovocia.

```

mode="max",
start_from_epoch=20,
verbose=1)

model.compile(
    optimizer=keras.optimizers.Adam(1e-3),
    loss=keras.losses.SparseCategoricalCrossentropy(),
    metrics=["accuracy"])

```

Teda od 20-tej epochy začíname sledovať validačnú accuracy a ak sa 5 epôch nezlepšila, tréning ukončíme. Model následne skompilujeme a ako optimalizačný algoritmus nadstavíme *Adam* s hodnotou learning rate-u 0.001.

Dáta najprv predspracujeme pomocou funkcie `preprocess_input` z modulu `keras.applications.mobilenet_v`. Teda pôvodné hodnoty z intervalu [0, 255] naškálujeme na hodnoty [-1, 1], aby to zodpovedalo tréningu pôvodnej *MobilNetV2* siete. Ďalej ich rozdelíme na TRAIN/VAL/TEST pomerom 60/20/20 stratifikované, teda zachováme pôvodné zastúpenie jednotlivých tried, tak ako v celom datasete.

Trénovacie dáta počas tréningu augmentujeme pomocou *ImageDataGenerator* takto:

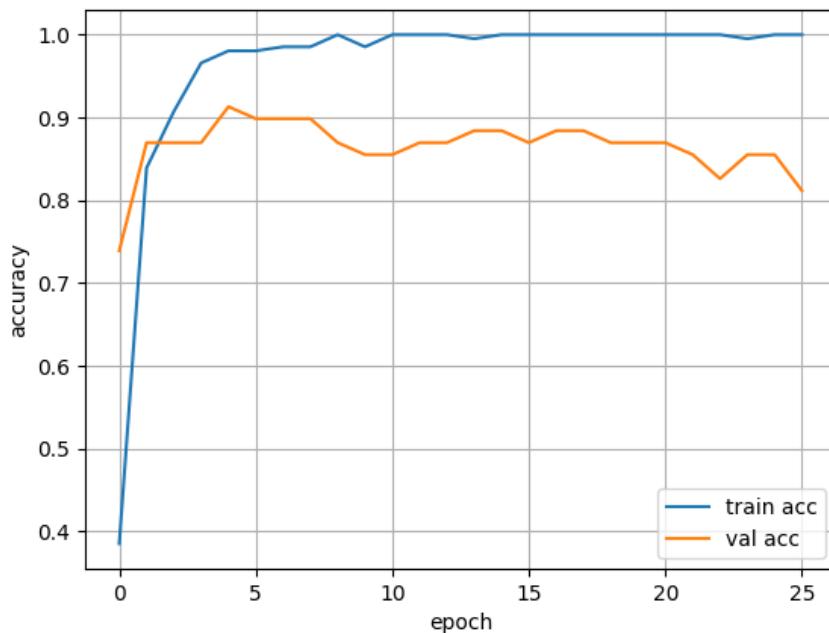
```

train_datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.1,
    height_shift_range=0.1,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode="nearest"
)

```

Teda každý obrázok náhodne zrotujeme o $\pm 20^\circ$, posunieme až o 10% do šírky a výšky a priblížime alebo oddialime o 20%. Príklad takto augmentovaných dát môžeme vidieť na obrázku 3.

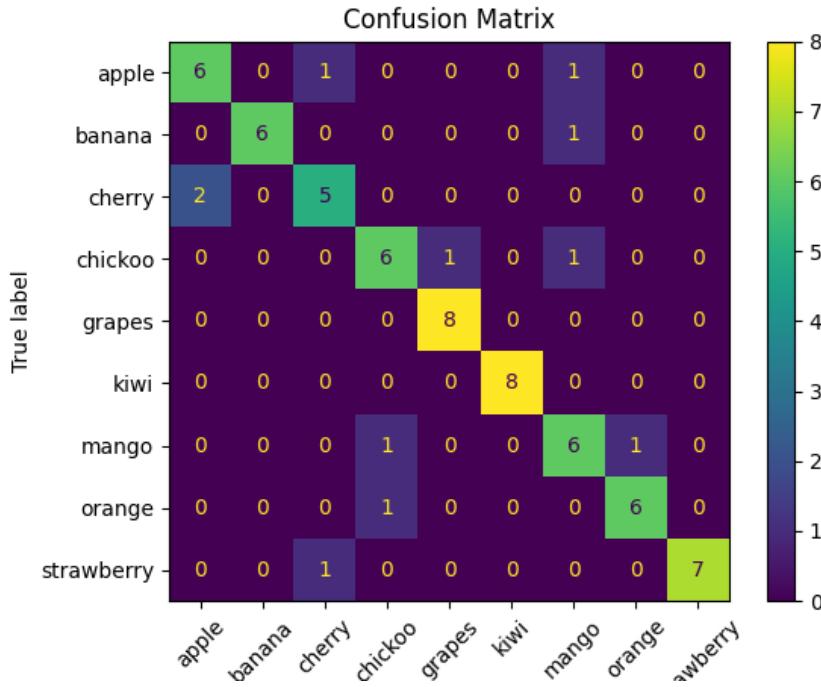
S takto upravenými dátami môžeme spustiť finálne trénovanie a model uložiť a otestovať na testovacích dátach. Maximálny počet epoch sme nadstavili na 100, avšak trénovanie prestalo už 26-tou epochou. Finálna testovacia accuracy je 0.84. Priebeh trénovacej a validačnej accuracy počas tréningu môžeme vidieť na obrázku 4.



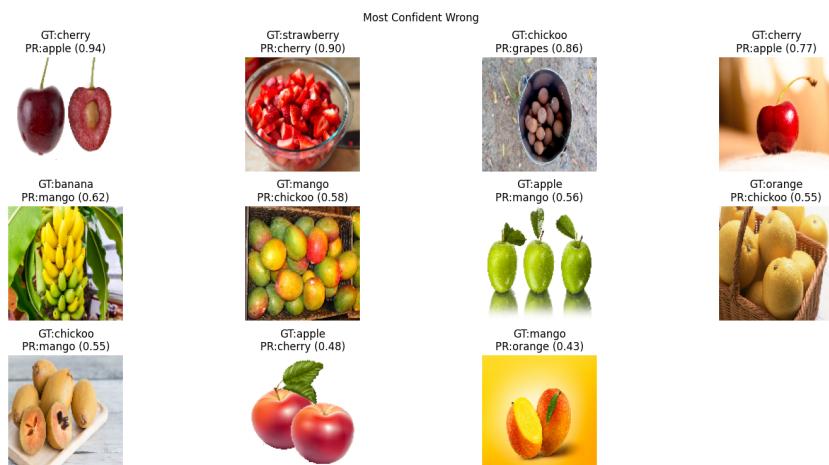
Obr. 4: Ukážka priebehu accuracy počas tréningu.

Okrem hlavnej metriky accuracy sa môžeme pozrieť aj na tzv. confusion matrix na obrázku 5, ktorá nám ukazuje, ktoré hodnoty sa najčastejšie medzi sebou zamienali. Napríklad určitý typ podobnosti môžeme vidieť medzi triedou *apple* a *cherry*. Na druhú stranu podľa tejto matice si náš model vie dobre poradiť s triedou *grapes* a aj s triedou *kiwi*, keďže ani jeden obrázok neklasifikoval zle.

Ako posledné sa môžeme pozrieť na to, ktorými obrázkami si náš model bol najviac istý a neklasifikoval ich správne. Toto vidíme na obrázku 6. Podobný trend ako nám ukázal obrázok 5, vidíme aj tu, keďže medzi prvými 4 obrázkami sa dvakárat nachádza trieda *cherry* a náš model si mysel, že ide o triedu *apple*. Môžeme sa domnievať, že ide hlavne o relatívne podobný tvar týchto druhov ovocia.



Obr. 5: Confusion matrix pre klasifikátor druhu ovocia.



Obr. 6: Zle klasifikované obrázky s najväčšou istotou.

4 Klasifikátor počtu ovocia

Analogicky k predchádzajúcej kapitole je aj úloha klasifikácie počtu ovocia riešená pomocou transfer learning-u. Opäť využívame predtrénovanú konvolučnú neurónovú sieť MobileNetV2 ako extraktor príznakov, pričom architektúra modelu sa lísi iba v definícii výstupnej vrstvy. Tá je prispôsobená tak, aby predikovala jednu zo štyroch tried reprezentujúcich počet objektov na obrázku, definovaných množinou

$$\mathcal{C}_{\text{count}} = \{\text{one}, \text{two}, \text{three}, \text{many}\}.$$

Formálne je architektúra modelu zapísaná nasledovne:

```
base_model = keras.applications.MobileNetV2(
    input_shape=(128, 128, 3),
    include_top=False
)
base_model.trainable = False

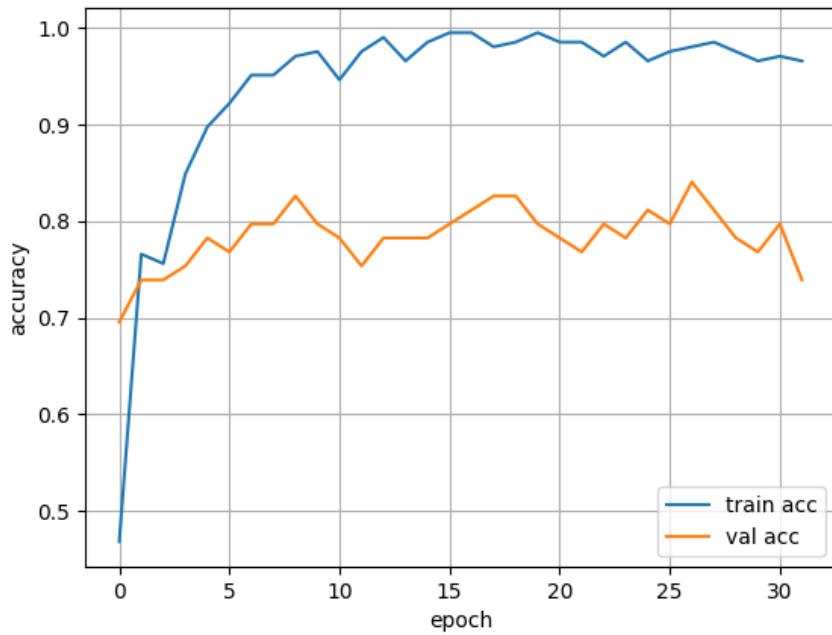
model = Sequential([
    base_model,
    keras.layers.GlobalAveragePooling2D(),
    Dense(
        128,
        activation="tanh",
        kernel_regularizer=keras.regularizers.l1_l2(1e-3, 1e-3)
    ),
    Dropout(0.3),
    Dense(4, activation="softmax"),
])
```

Aj v tomto prípade zachovávame stratifikované rozdelenie dát na trénovaciu, validačnú a testovaciu množinu, ako aj augmentáciu aplikovanú výhradne na trénovacie dátu. Výsledný model dosahuje testovaciu accuracy približne 0.71, čo je mierne nižšia hodnota v porovnaní s modelom určeným na klasifikáciu druhu ovocia. Pre detailnejšiu analýzu výkonu modelu sú opäť použité rovnaké diagnostické grafy ako v predchádzajúcej kapitole.

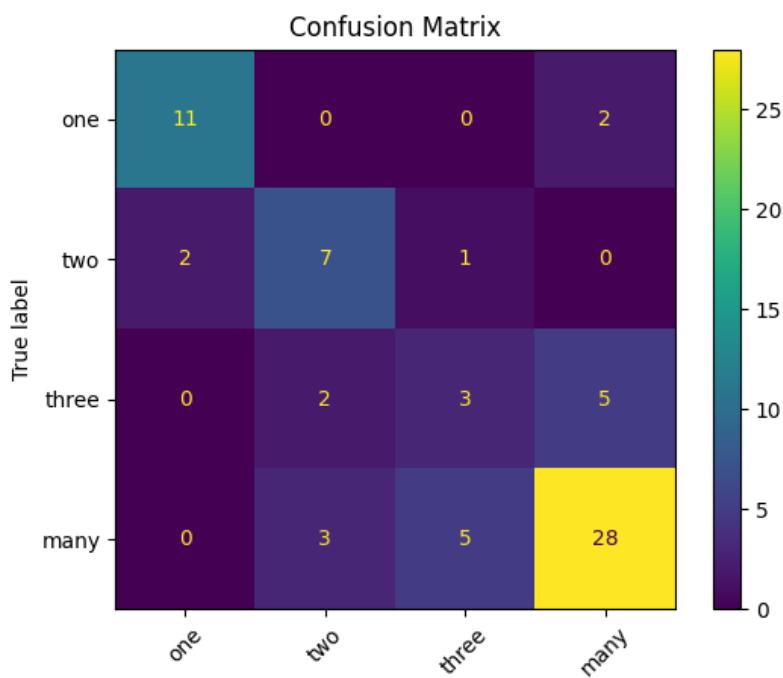
Na obrázku 7 je zobrazený vývoj trénovacej a validačnej accuracy v priebehu učenia. Je možné pozorovať výraznejší rozdiel medzi týmito krivkami, pričom trénovacia accuracy sa približuje k hodnote 1. Tento jav naznačuje miernu mieru overfittingu.

Na obrázku 8 je znázornená confusion matrix výsledného modelu. Najvýraznejší zdroj chýb možno pozorovať pri triede *three*, ktorú si model často zamieňa s triedou *many*. Pravdepodobnou príčinou tohto správania je nevyváženosť datasetu, keďže trieda *three* obsahuje najmenší počet pozorovaní. Zvýšenie množstva dát pre túto triedu by pravdepodobne mohlo viesť k zlepšeniu schopnosti modelu.

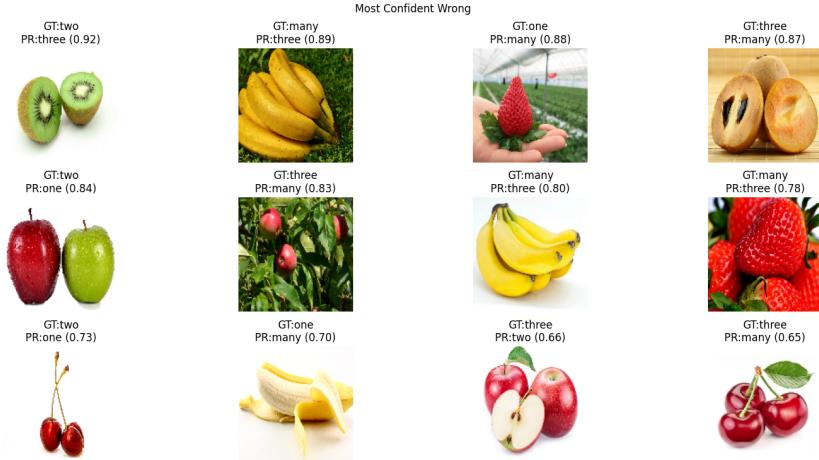
Na záver uvádzame niekoľko ukážok obrázkov, pri ktorých si bol model vo svojej predikcii najistejší, avšak výsledná klasifikácia bola nesprávna. Z obrázka 9 je zrejmé, že trieda *three* predstavuje pre model problematickú kategóriu. Zvlášť zaujímavý je prvý zobrazený príklad, v ktorom model predikuje triedu *three*, hoci na obrázku sú zreteľne viditeľné dve polovice jedného kiwi. V tomto prípade nie je úplne jednoznačné, prečo model dospel k takto nesprávnemu a zároveň vysoko sebavedomému odhadu.



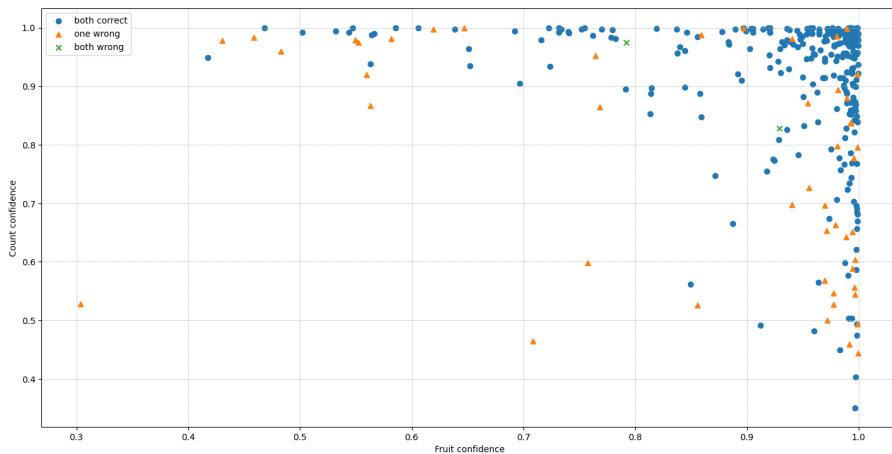
Obr. 7: Vývoj trénovacej a validačnej accuracy počas tréningu modelu.



Obr. 8: Confusion matrix pre klasifikátor počtu ovocia.



Obr. 9: Najistejšie nesprávne odhady modelu.



Obr. 10: Rozdelenie spoločných predikcií.

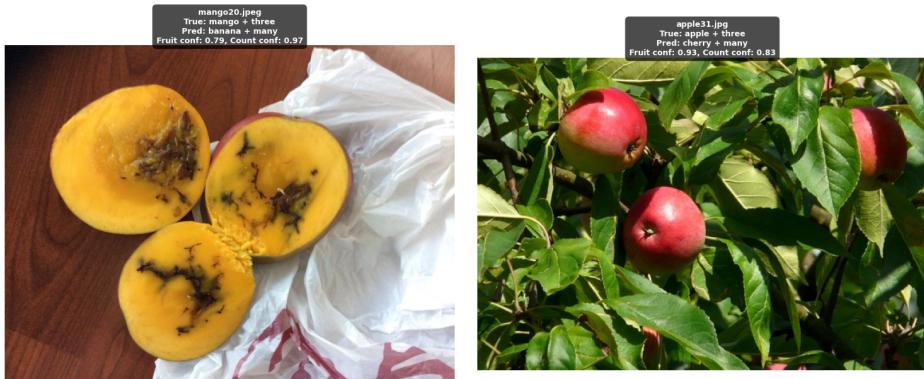
5 Spojenie do jednej predikcie

V záverečnej kapitole spájame oba natrénované modely a analyzujeme ich spoločné správanie pri klasifikácii dát. Postup je priamočiary – oba uložené modely načítame a pre celý dataset sú vypočítané ich predikcie.

Najskôr sa zameriavame na analýzu správnosti spoločných predikcií, konkrétnie na rozdelenie prípadov, v ktorých oba modely zlyhali, zlyhal iba jeden z nich, alebo boli obe predikcie správne. Tento vzťah je znázornený na obrázku 10, kde osi reprezentujú maximálne predikované pravdepodobnosti jednotlivých modelov.

Z výsledkov vyplýva, že simultánne zlyhanie oboch modelov je veľmi zriedkavé a vyskytuje sa iba v dvoch prípadoch. Zároveň je možné pozorovať očakávaný trend, podľa ktorého sa nesprávne predikcie koncentrujú v oblastiach s nižšími hodnotami pravdepodobností, teda tam, kde si modely nie sú svojím rozhodnutím dostatočne isté. Na obrázku 11 sú zobrazené konkrétnie pozorovania, pri ktorých zlyhali oba modely súčasne. V oboch prípadoch klasifikátor počtu ovocia predikoval triedu *many*, zatiaľ čo skutočná hodnota bola *three*. Zaujímavý je aj nesprávny odhad druhu ovocia, kde si model pomýlil *banana* s *mango*.

Ak vynásobíme maximálne predikované pravdepodobnosti oboch modelov, môžeme sa



Obr. 11: Pozorovania, pri ktorých zlyhali oba modely.



Obr. 12: Najistejšie, no nesprávne spoločné predikcie modelov.

na obrázku 12 zamierať na inštancie, pri ktorých si boli modely vo svojej predikcii najistejšie, avšak výsledok bol nesprávny. Pri detailnejšom pohľade na tieto chyby možno konštatovať, že väčšina z nich je z hľadiska vizuálnej podobnosti objektov pomerne pochopiteľná, napríklad zámena *kiwi* a *chickoo*. Opäť sa tu však výrazne prejavuje častá zámena tried *three* a *many*.

Celkovú accuracy oboch modelov, ako aj ich spoločnej predikcie na celom datasete, uvádzame v tabuľke 5.

Accuracy	
Klasifikátor druhu ovocia	0.9417
Klasifikátor počtu ovocia	0.9067
Oba modely správne	0.8542

Tabuľka 1: Presnosť jednotlivých modelov a ich spoločnej predikcie.