

**SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY**

AIS ID: 92320

**KONVOLUČNÉ SIETE A PRENOS VEDOMOSTÍ
ZADANIE**

Predmet: I-SUNS – Strojové učenie a neurónové siete
Prednášajúci: prof. Dr. Ing. Miloš Oravec
Cvičiaci: Ing. Marián Šebeňa

Bratislava 2024

Bc. Lukáš Patrnčiak

Obsah

Úvod	1
1 Spracovanie dát	2
2 Konvolučná sieť	4
3 Transfer Learning	9
3.1 Dotrénovanie modelu	20
3.2 Neurónová sieť pre zhľuky	24
Záver	31

Zoznam obrázkov a tabuliek

Obrázok 1	Zobrazení reprezentanti tried v trénovacej množine	2
Obrázok 2	Zobrazení reprezentanti tried v validačnej množine	3
Obrázok 3	Zobrazení reprezentanti tried v testovacej množine	3
Obrázok 4	Zobrazene priebehu trénovania a testovania	5
Obrázok 5	Konfúzna matica pre trénovaciu množinua	6
Obrázok 6	Konfúzna matica pre validačnú množinu	7
Obrázok 7	Konfúzna matica pre testovaciu množinu	8
Obrázok 8	1. zhluk obrázkov	10
Obrázok 9	Priemerný obrázok pre zhluk 1	10
Obrázok 10	2. zhluk obrázkov	11
Obrázok 11	Priemerný obrázok pre zhluk 2	11
Obrázok 12	3. zhluk obrázkov	12
Obrázok 13	Priemerný obrázok pre zhluk 3	12
Obrázok 14	4. zhluk obrázkov	13
Obrázok 15	Priemerný obrázok pre zhluk 4	13
Obrázok 16	5. zhluk obrázkov	14
Obrázok 17	Priemerný obrázok pre zhluk 5	14
Obrázok 18	6. zhluk obrázkov	15
Obrázok 19	Priemerný obrázok pre zhluk 6	15
Obrázok 20	7. zhluk obrázkov	16
Obrázok 21	Priemerný obrázok pre zhluk 7	16
Obrázok 22	8. zhluk obrázkov	17
Obrázok 23	Priemerný obrázok pre zhluk 8	17
Obrázok 24	9. zhluk obrázkovv	18
Obrázok 25	Priemerný obrázok pre zhluk 9	18
Obrázok 26	10. zhluk obrázkov	19
Obrázok 27	Priemerný obrázok pre zhluk 10	19
Obrázok 28	Vizualizácia zhlukov pomocou t-SNE	20
Obrázok 29	Trénovací a testovací priebeh transfer learing modelu	21
Obrázok 30	Konfúzna matica pre trénovaciu množinu	22
Obrázok 31	Konfúzna matica pre validačnú množinu	23
Obrázok 32	Konfúzna matica pre testovaciu množinu	24

Obrázok 33	Trénovací a testovací priebeh neurónovej siete pre vybrané zhľuky	26
Obrázok 34	Konfúzna matica pre trénovaciu množinu	27
Obrázok 35	Konfúzna matica pre validačnú množinu	28
Obrázok 36	Konfúzna matica pre testovaciu množinu	29
Obrázok 37	Grafické znázornenie vyhodnotenia úspešnosti pre jednotlivé zhľuky	30
Tabuľka 1	Konfigurácia jednotlivých experimentov	4
Tabuľka 2	Konfigurácia jednotlivých experimentov (pokračovanie)	5
Tabuľka 3	Vyhodnotenie úspešnosti pre jednotlivé experimenty	5

Úvod

Cieľom tohto zadania je v programovacom jazyky Python implementovať program, ktorý bude klasifikovať vtáčikov na obrázku z poskytnutého datasetu, ktorý obsahuje približne 7200 obrázkov v trénovacej, testovacej a validačnej množine. Triedy sú určené podľa priečinka, v ktorom sa nachádzajú.

Najprv je potrebné spracovať a analyzovať poskytnuté dátá a zobraziť reprezentanta z triedy.

Potom je potrebné vytvoriť a natrénovať konvolučnú neurónovú sieť na klasifikáciu vtáčikov. Na tento účel vytvoríme konvolučnú neurónovú sieť s aspoň dvoma vrstvami, vrátane vytvorenia experimentov, kde boli zmenené niektoré hyperparametre.

Na záver bolo potrebné zlepšenie výsledkov použitím princípov transfer learningu.

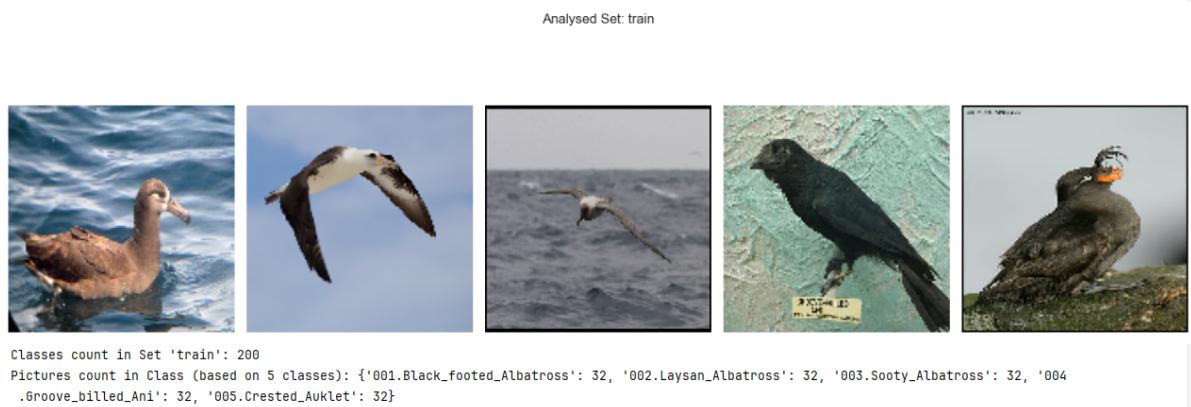
1 Spracovanie dát

V tejto kapitole bude popísaný spôsob spracovania dát, ktorých štruktúra je:

- **dataset** - hlavný priečinok
 - **množiny** - trénovacia (train), testovacia (test) a validačná (validation)
 - * **tryedy** - jednotlivé triedy (skupiny obrázkov - v tomto prípade vtáčikov)
 - **obrázky** - niekolko obrázkov (vtáčikov) prislúchajúcim k danej triede

Na spracovanie dát boli použité knižnice NumPy¹, Pandas², Seaborn³, Matplotlib.pyplot⁴, skLearn⁵, Tensorflow (Keras)⁶.

Pomocou funkcie *analyze_data()* sme vypísali názov množín, ktoré obsahuje dataset (trénovacia - train, testovacia - test, validačná - validation) a počet tried, nachádzajúcich sa v týchto množinách a celkový počet reprezentantov pre vybraný počet tried. Funkcia ďalej zobrazí reprezentanta z ľubovoľného počtu tried.



Obrázok 1: Zobrazení reprezentanti tried v trénovacej množine

¹<https://numpy.org/>

²<https://pandas.pydata.org/>

³<https://seaborn.pydata.org/>

⁴https://matplotlib.org/stable/api/pyplot_summary.html

⁵<https://scikit-learn.org/stable/>

⁶<https://www.tensorflow.org/guide/keras>

Analysed Set: validation



Classes count in Set 'validation': 200

Pictures count in Class (based on 5 classes): {'001.Black_footed_Albatross': 4, '002.Laysan_Albatross': 4, '003.Sooty_Albatross': 4, '004.Groove_billed_Ani': 4, '005.Crested_Auklet': 4}

Obrázok 2: Zobrazení reprezentanti tried v validačnej množine

Analysed Set: test



Classes count in Set 'test': 200

Pictures count in Class (based on 5 classes): {'001.Black_footed_Albatross': 4, '002.Laysan_Albatross': 4, '003.Sooty_Albatross': 4, '004.Groove_billed_Ani': 4, '005.Crested_Auklet': 4}

Obrázok 3: Zobrazení reprezentanti tried v testovacej množine

2 Konvolučná sieť

Pre trénovaciu, testovaciu a validačnú množinu bol pripravený generátor dát (`ImageDatasetGenerator`), pomocou ktorého boli dátá (okrem iného) normalizované do rozsahu [0, 1] (pomocou škály 1.0/255) a pre rýchlejšie spracovanie im bola aj zmenšená veľkosť. Trénovacím a validačným dátam bolo nastavené shuffle (premiešanie) (zlepšenie učenia neurónovej siete, aby sa model naučil vzory špecifické pre poradové usporiadanie, pri validačných dátach nie je nutné, ale bolo pridané pre väčšiu robustnosť).

Navrhnutá konvolučná sieť je tvorená funkciou `create_cnn_model()` a trénovaná funkciou `train_cnn_model()`, pričom je architektúra je:

- **Počet konvolučných vrstiev:** 3
- **Počet neurónových vrstiev:** 1 (výstupná vrstva)
- **Regularizačné techniky:**
 - *Dropout*: zníženie pretrénovania vypínaním určitého náhodného vypnutia neurónov
 - *BatchNormalization*: zlepšuje tréning a stabilitu
- **EarlyStopping**: V prípade, že sa sieť začne pretrénovať, zastaví sa jej trénovanie
- **Checkpoint**: Počas trénovania boli vytvárané checkpointy (kvôli náročnosti výpočtov)

Hyperparametre siete, ako napríklad počet filtrov pre jednotlivé konvolučné vrstvy, dropout pre jednotlivé vrstvy modelu a počet neurónov na výstupnej vrstve, rýchlosť učenia je možné nastaviť. To sme použili pri vykonaní nasledovných 3 experimentov (ktoré boli vykonané pri 10 epochách):

Číslo	Rýchlosť učenia	1. vrstva	2. vrstva	3. vrstva	4. vrstva
1	0.001	32	32	64	256
2	0.001	32	64	64	128
3	0.001	32	128	128	128

Tabuľka 1: Konfigurácia jednotlivých experimentov

Číslo	1. dropout	2. dropout	3. dropout	4. dropout
1	0.1	0.1	0.1	0.25
2	0.25	0.1	0.1	0.25
3	0.1	0.1	0.25	0.25

Tabuľka 2: Konfigurácia jednotlivých experimentov (pokračovanie)

Číslo	Tréningová úspešnosť	Validačná úspešnosť	Testovacia úspešnosť
1	0.97969	0.09250	0.08375
2	0.068906	0.1	0.06625
3	0.073437	0.06875	0.06625

Tabuľka 3: Vyhodnotenie úspešnosti pre jednotlivé experimenty

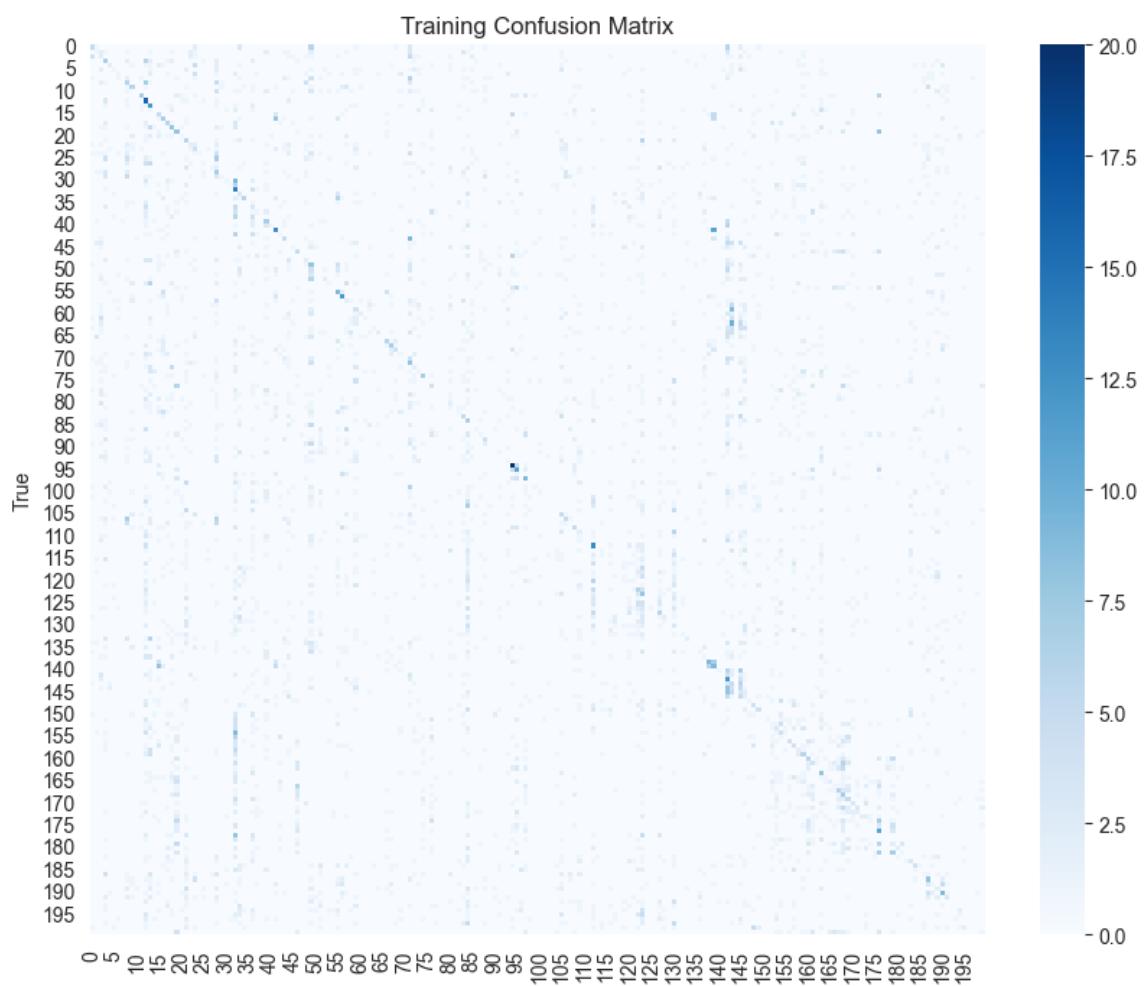
Na základe testovacej (a v podstate aj ostatných) úspešnosti sme dospeli k záveru, že najúspešnejšie obstál experiment číslo 1, pre ktorý vykreslím priebeh trénovania a testovania (pomocou funkcie `plot_training(history)`), vrátane konfúznych matíc, ktoré slúžia na vyhodnotenie výkonnosti klasifikačného modelu.



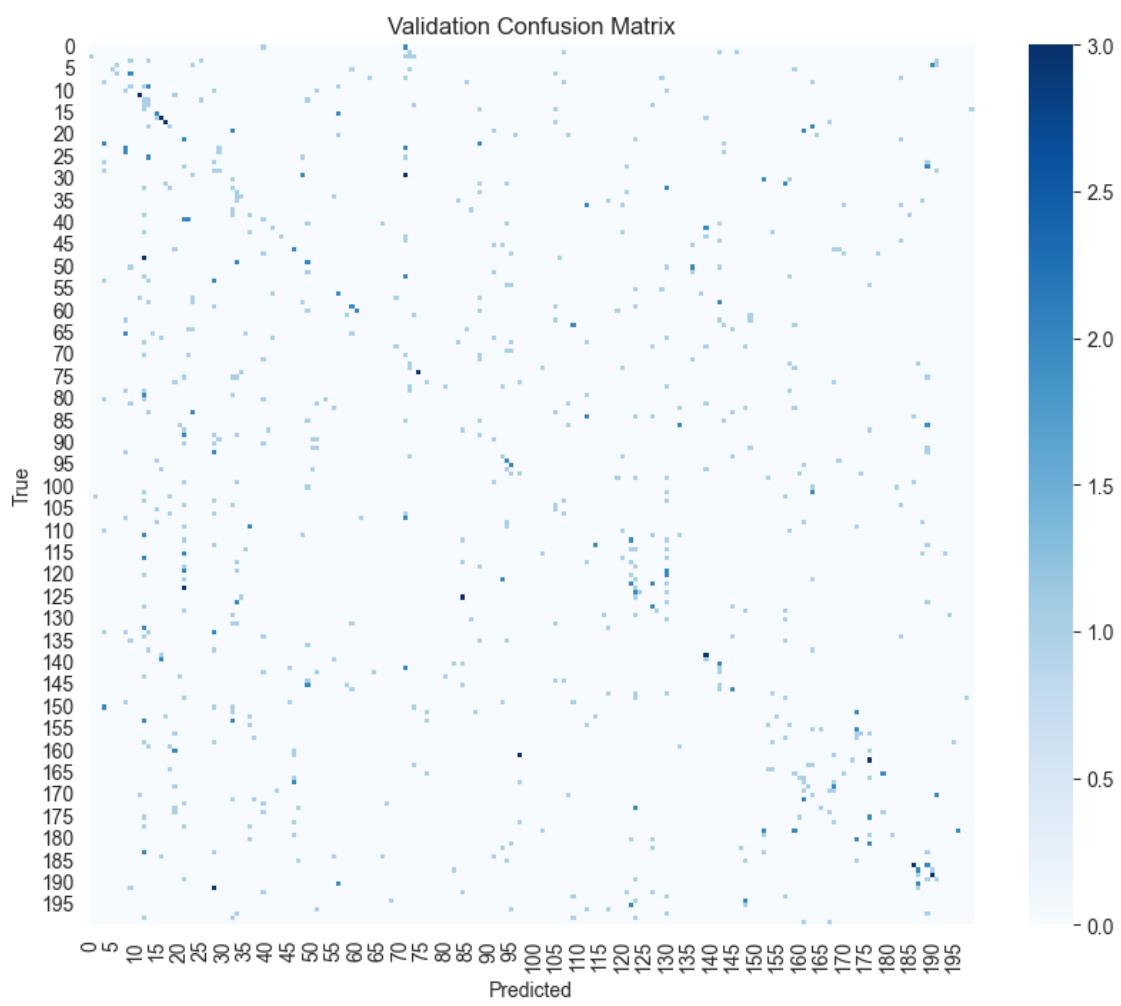
Obrázok 4: Zobrazene priebehu trénovania a testovania

Na X-ovej konfúznej matici osi sa nachádzajú predikované hodnoty a na Y-ovej osi sa nachádzajú skutočné hodnoty. Ako je možné si všimnúť, najviac dát je sústredených okolo hlavnej diagonály, čo zobrazuje správne predikcie, kde je predikovaná hodnota rovnaká ako skutočná hodnota. Mimo diagonály sú hodnoty, ktoré sú nesprávne predikované . Tieto hodnoty naznačujú, kde sa model pomýlil pri klasifikácii.

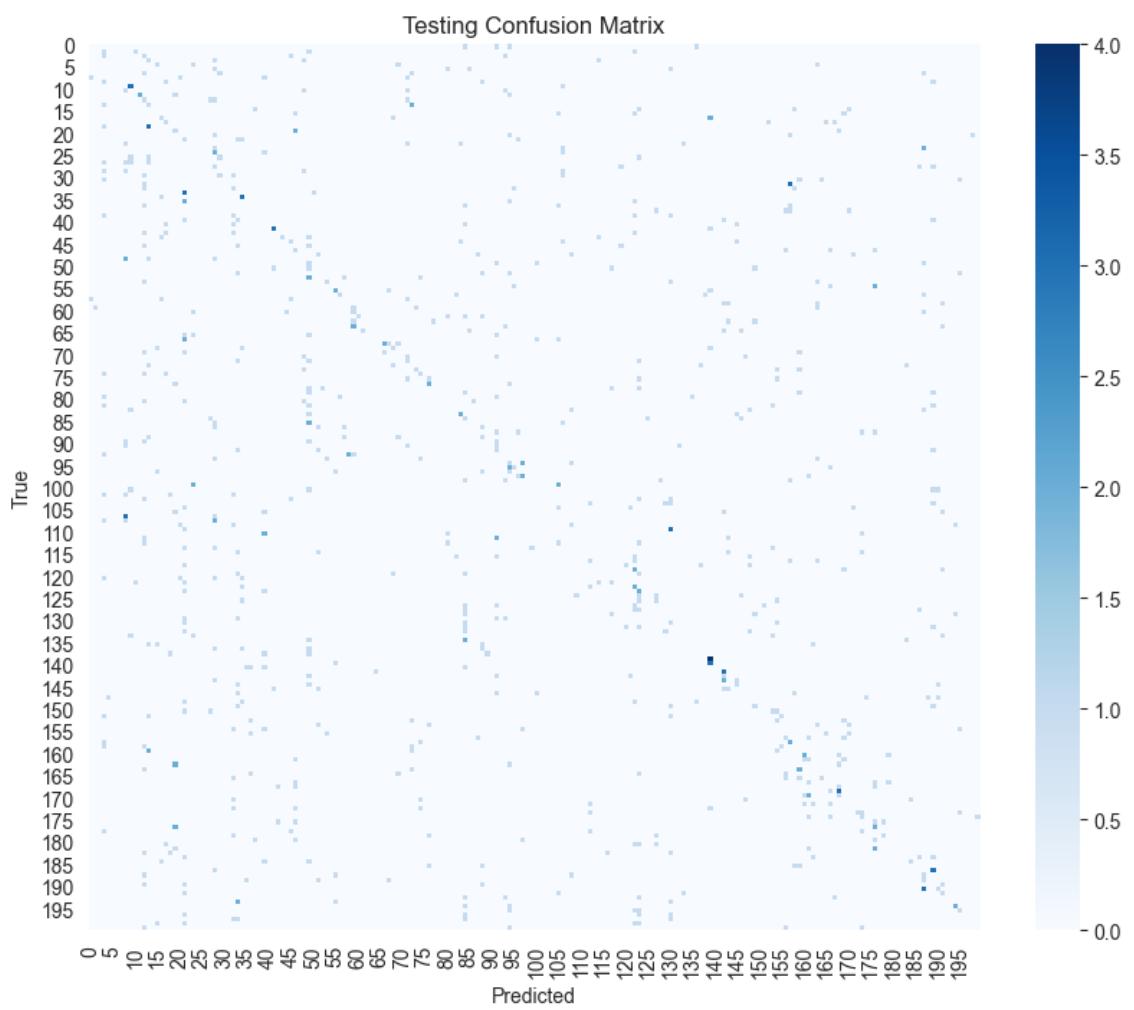
Konfúzne matice vykresluje funkcia `plot_confusion_matrix_shuffle()` v prípade trénovacích a validačných dátach (ktorá používa funkciu `get_prediction()` - táto funkcia bola poskytnutá na cvičení), resp. `plot_confusion_matrix_nonshuffle()` v prípade testovacích.



Obrázok 5: Konfúzna matica pre trénovaciu množinu



Obrázok 6: Konfúzna matica pre validačnú množinu



Obrázok 7: Konfúzna matica pre testovaciu množinu

3 Transfer Learning

Cieľom tejto časti bolo využitie princípov transfer learningu. Transfer learning je technika strojového učenia, pri ktorej sa poznatky získané z úlohy opäťovne používajú na zvýšenie výkonu súvisiacej úlohy.

Pre tento účet bola vybraná sieť **MobileNet**, ktorá je trénovaná na datasete ImageNet. Táto sieť bola vybraná z dôvodu jej rýchlosťi a efektívnosti. V tejto sieti boli zmrazené všetky vrstvy okrem jej posledných 20 vrstiev (aby sme si zachovali užitočné informácie, ktoré obsahujú - využívame tieto vrstvy bez rizika ich 'prepísania'.) Posledné vrstvy boli odmrazené hlavne preto, aby bolo možné vylepšovať presnosť siete na nových úlohách.

Táto sieť bola použitá vo funkcií *generate_features()*, ktorá vracia vektor týchto príznakov. O korektné ukladanie príznakov (každý do samostatného stĺpca) sa stará funkcia *dataset_to_dataframe()*. Táto funkcia okrem zhlukov pridáva aj informácie o množine ('set'), triedach ('class') a ceste k triedam ('path').

Pre zlepšenie výkonu bola zredukovaná dimenzia pomocou **PCA**, ktorá umožňuje znížiť počet rozmerov dát bez výraznejších strát.

Pre zhlukovanie vygenerovaných príznakov z obrázkov bol použitý algoritmus **K-Means**, ktorého cieľom je rozdeliť dátu do zhlukov, aby boli body v rámci jedného zhluku čo najpodobnejšie a body z rôznych zhlukov čo najodlišnejšie. Tieto zhluky boli taktiež pridané do dataframe (ako stĺpec 'cluster'), pričom následne bol taktiež vygenerovaný *.csv* súbor obsahujúci tieto stĺpce ('features', 'class', 'set', 'path', 'cluster').

Na vybraných obrázkoch (5) v zhlukoch (10) môžeme vidieť rozmanitosť vtákov, ktoré môžu zdieľať nejakú spoločnú vlastnosť (tvar tela, operenia, ...), farebnú konzistenciu obrázkov a variabilitu v pozadí/prostredí. Toto zabezpečuje funkcia *plot_cluster_image()*.

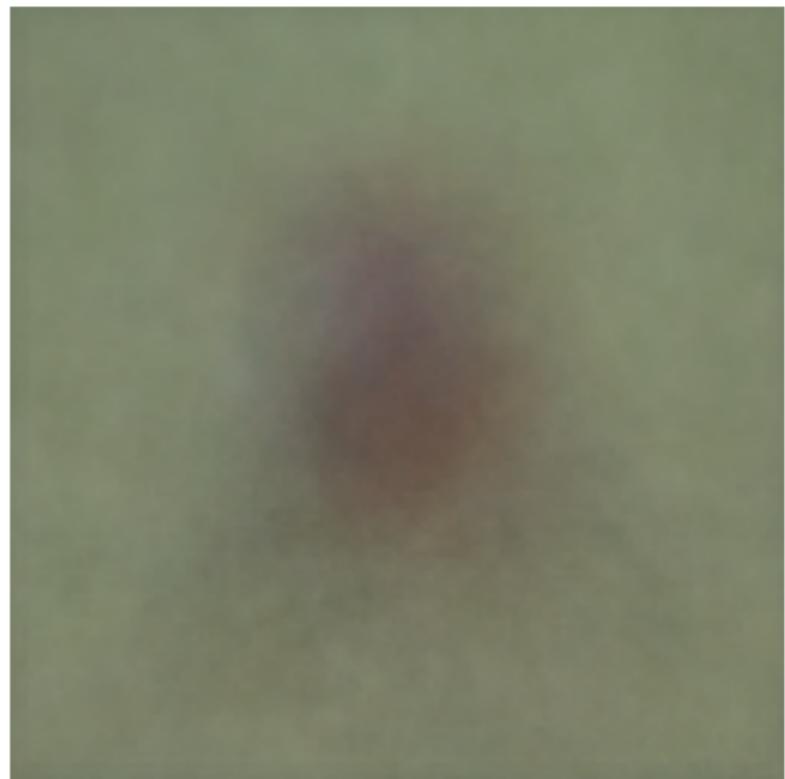
Priemerný obrázok znázorňuje zlúčenie viacerých charakteristík - ide o výsledok výpočtu priemerov pixelov zo všetkých obrázkoch v zhluku (dominancia jemných farebných prechodov). Vykreslenie tohto obrázku zabezpečuje funkcia *compute_cluster_average()*, ktorá vypočíta priemernú hodnotu pixelov, pričom dátu skonvertuje na formát *uint8* pre ich ďalšie spracovanie.

Cluster 0



Obrázok 8: 1. zhluk obrázkov

Average Image per Cluster 0



Obrázok 9: Priemerný obrázok pre zhluk 1

Cluster 1



Obrázok 10: 2. zhluk obrázkov

Average Image per Cluster 1



Obrázok 11: Priemerný obrázok pre zhluk 2

Cluster 2



Obrázok 12: 3. zhluk obrázkov

Average Image per Cluster 2



Obrázok 13: Priemerný obrázok pre zhluk 3

Cluster 3



Obrázok 14: 4. zhluk obrázkov

Average Image per Cluster 3



Obrázok 15: Priemerný obrázok pre zhluk 4

Cluster 4



Obrázok 16: 5. zhluk obrázkov

Average Image per Cluster 4



Obrázok 17: Priemerný obrázok pre zhluk 5

Cluster 5



Obrázok 18: 6. zhluk obrázkov

Average Image per Cluster 5



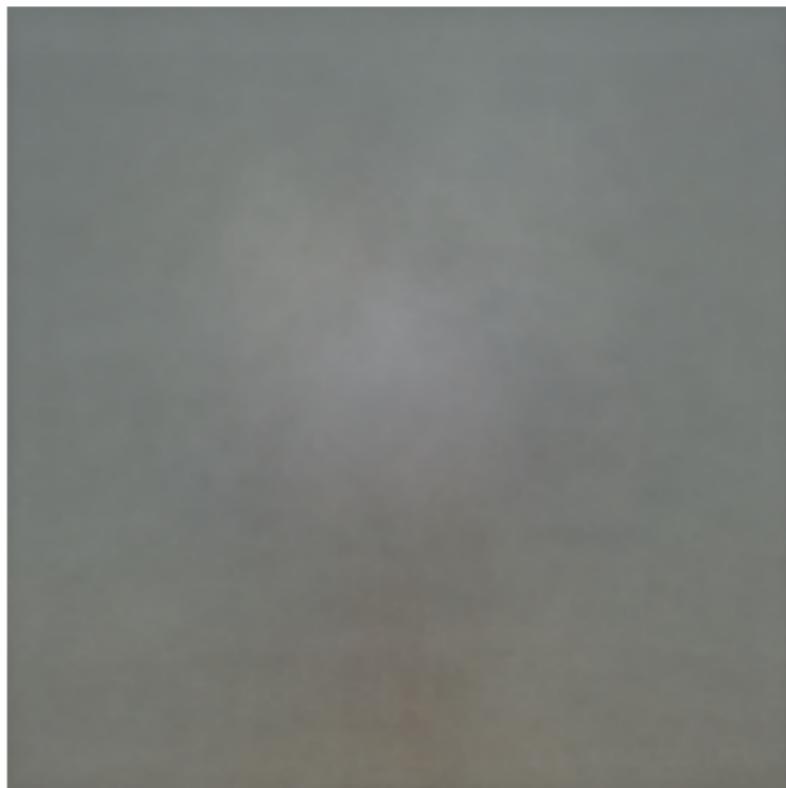
Obrázok 19: Priemerný obrázok pre zhluk 6

Cluster 6



Obrázok 20: 7. zhluk obrázkov

Average Image per Cluster 6



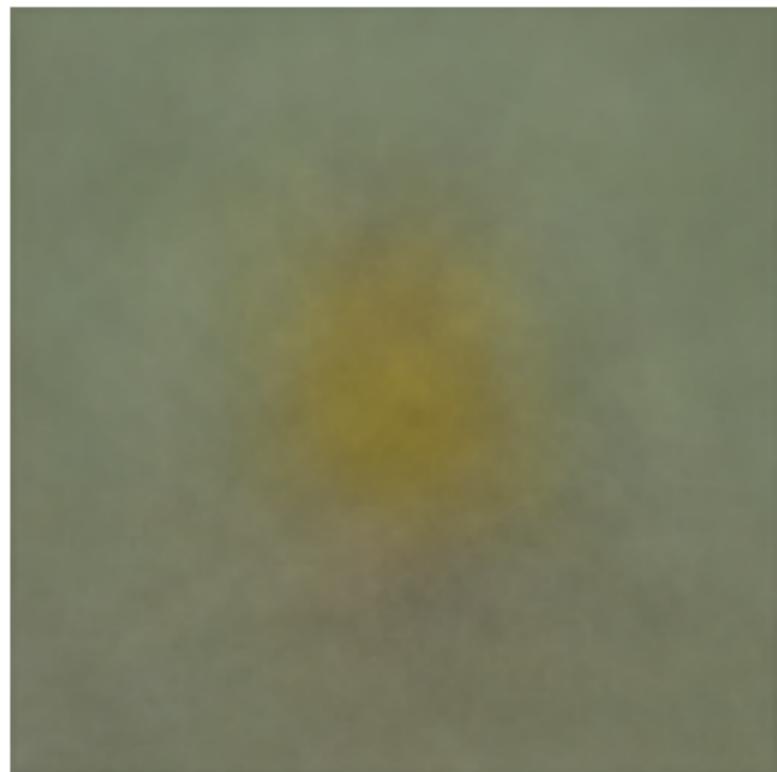
Obrázok 21: Priemerný obrázok pre zhluk 7

Cluster 7



Obrázok 22: 8. zhluk obrázkov

Average Image per Cluster 7



Obrázok 23: Priemerný obrázok pre zhluk 8

Cluster 8



Obrázok 24: 9. zhluk obrázkovv

Average Image per Cluster 8



Obrázok 25: Priemerný obrázok pre zhluk 9

Cluster 9



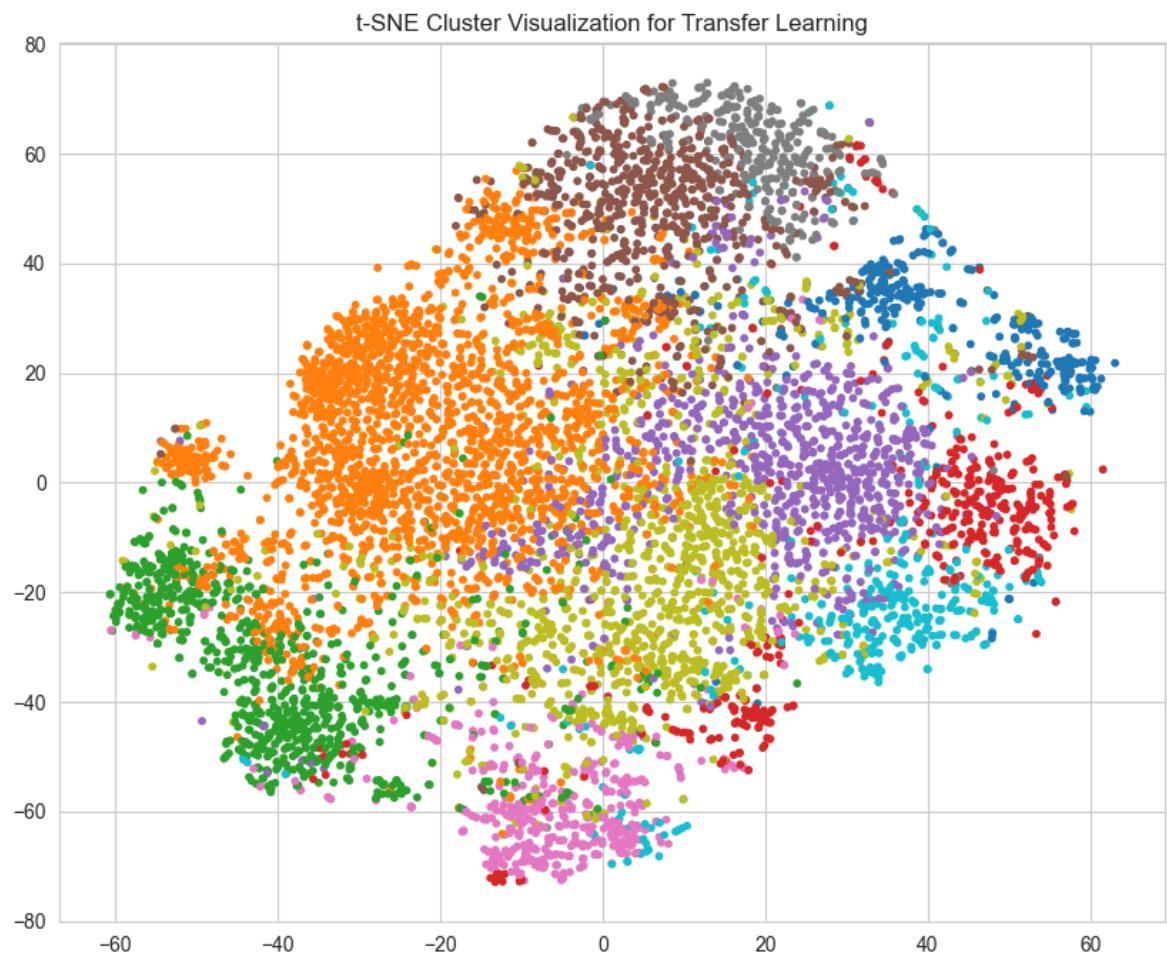
Obrázok 26: 10. zhluk obrázkov

Average Image per Cluster 9



Obrázok 27: Priemerný obrázok pre zhluk 10

Nasledujúci obrazok zobrazuje finálnu 2D reprezentáciu zhlukov pomoocu t-SNE, ktorý umožňuje efektívne vizualizovať zhluky vo vysokodimenzionálnych dátach. Každý zhluk má inú farbu, čo zodpovedá iným triedam identifikovaných algoritmom K-Means. Väčšia skupina zhlukov znázorňuje identifikovanú štruktúru. Prekrývanie zhlukov znázorňuje ich podobnosť.



Obrázok 28: Vizualizácia zhlukov pomocou t-SNE

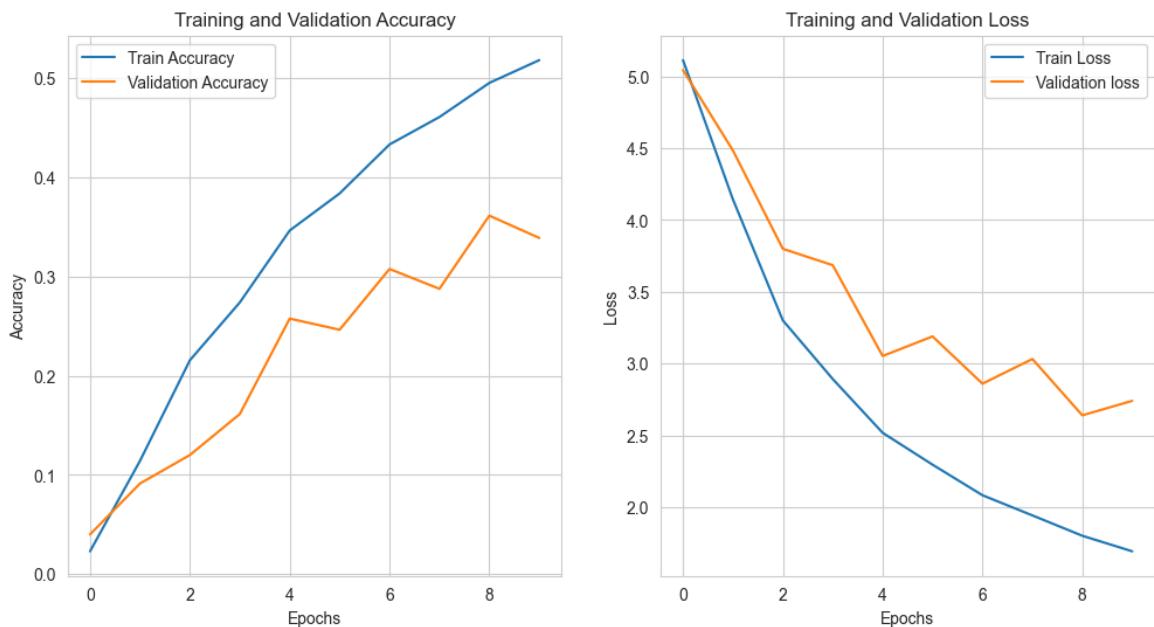
3.1 Dotréновanie modelu

ImageNet model bol dotrénovaný pre funkciu *transfer_learning_model()* - neurónovou sieťou, ktorého architektúru tvorí pridaných ďalších 64 neurónov s dropoutom 25% pri rýchlosťi učenia 0.001. Výsledky trénovania, testovania a validácie tejto siete su popísané nižšie.

Počet epoch bol nastavený na 10 (pre rýchlejšie spracovanie dát).

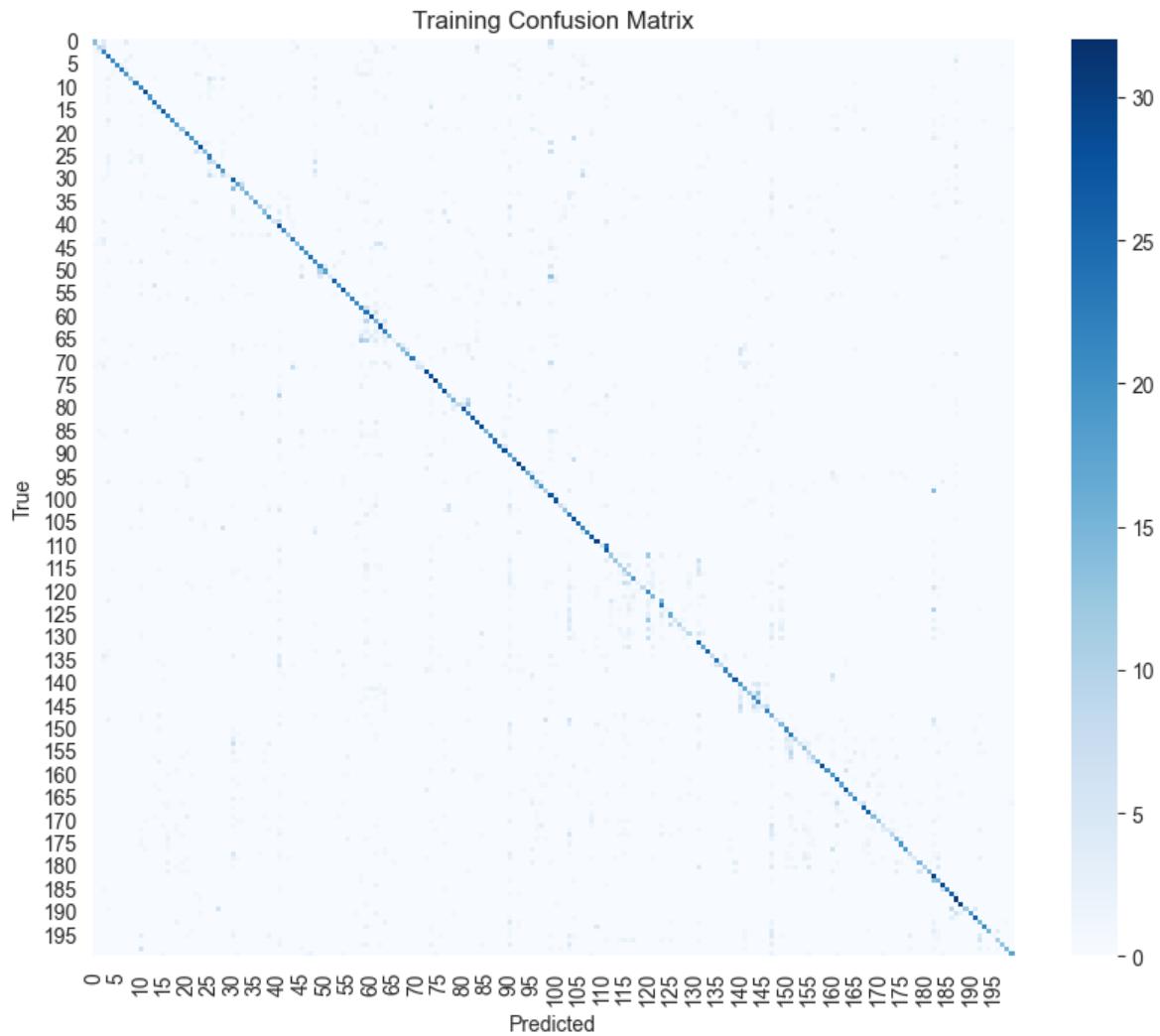
- Tréning:
 - *Presnosť*: 0.55890
 - *Strata*: 1.73381
- Validácia:
 - *Presnosť*: 0.36125
 - *Strata*: 2.64001
- Testovanie:
 - *Presnosť*: 0.35249
 - *Strata*: 2.6318

Priebeh trénovania pre trénovanie a testovanie a aj konfúzne matice pre trénovaciu testovaciu a validačnú množinu sú zobrazené nižšie. Ako je možné vidieť, v porovnaní s vlastným modelom konvolučnej siete nastalo zlepšenie (čo sa týka úspešnosti, zlepšenie je možné vidieť aj na konfúznych maticiach - väčšie zvýraznenie hlavnej diagonály.)

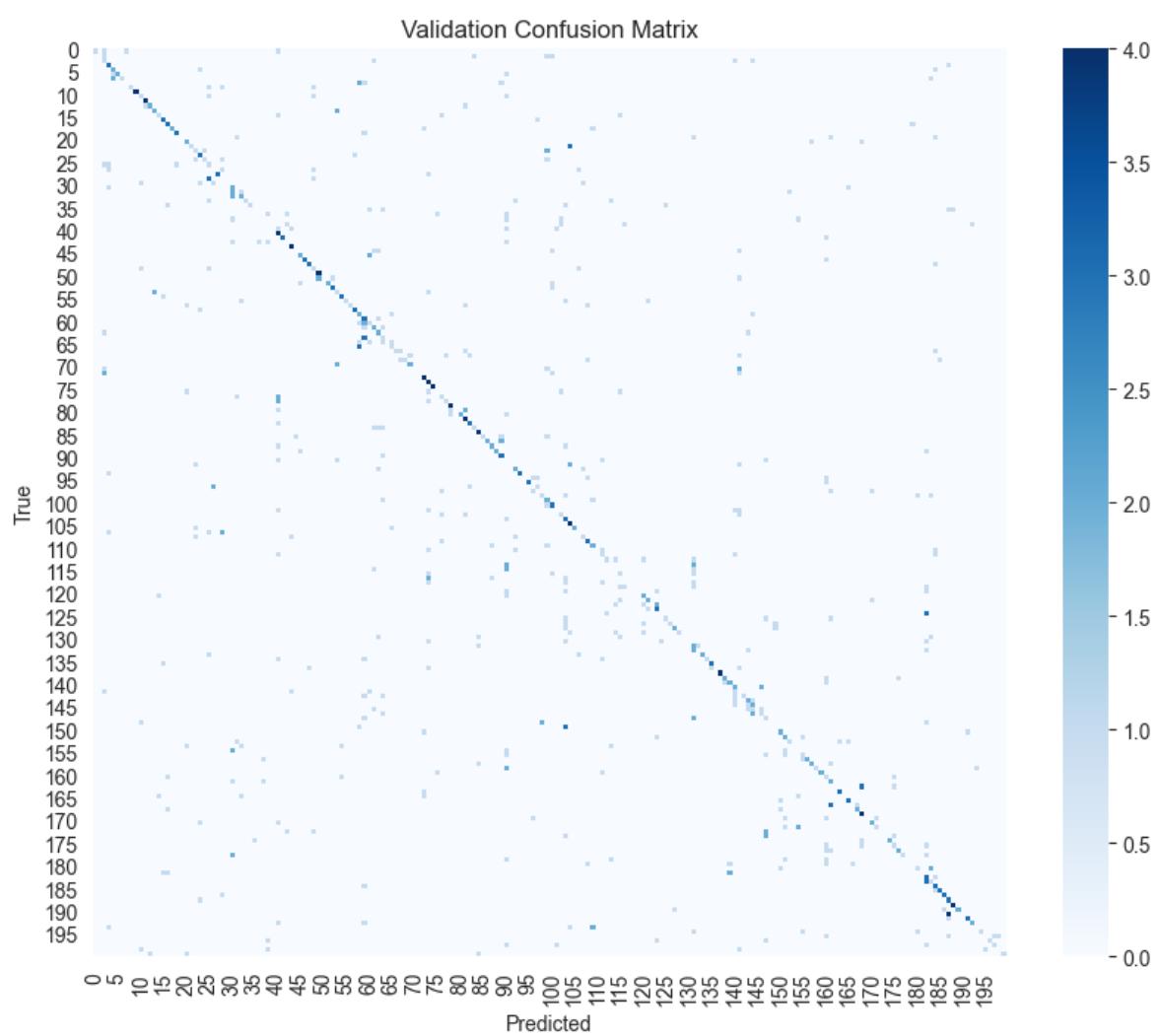


Obrázok 29: Trénovací a testovací priebeh transfer learning modelu

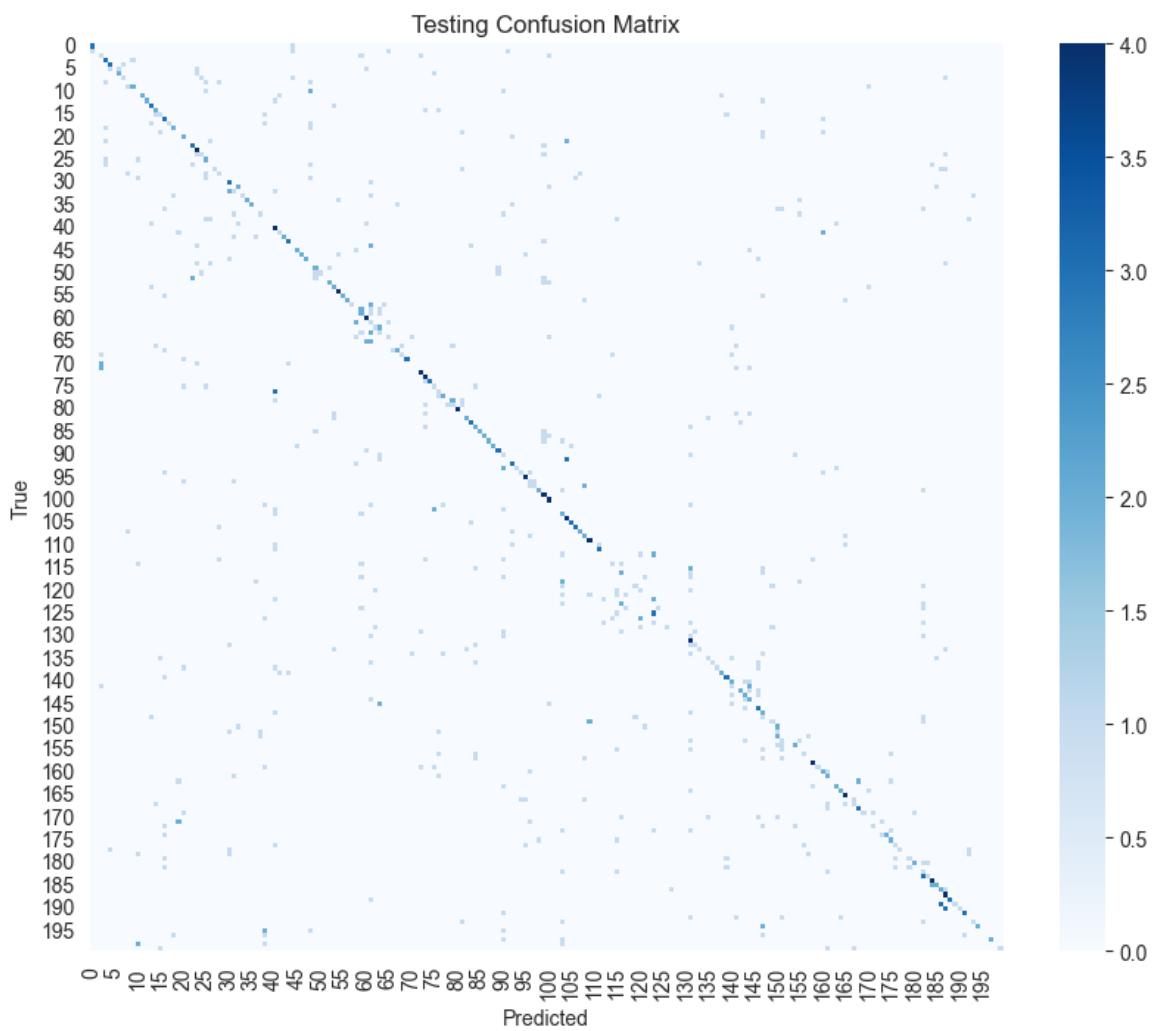
na vykreslenie týchto konfúznych matíc bola použitá funkcia `plot_confusion_matrix()`, ktorá pracuje s dátovými množinami a nie triedami dát.



Obrázok 30: Konfúzna matica pre trénovaciu množinu



Obrázok 31: Konfúzna matica pre validačnú množinu



Obrázok 32: Konfúzna matica pre testovaciu množinu

3.2 Neurónová sieť pre zhluky

Pre vytvorenie neurónovej siete pre 10 vybraných zhlukov je potrebné upraviť dataframe, resp. rozdeliť dátá na vstupnú (X) a výstupnú (Y) množinu.

Vstupná množina bude obsahovať všetky stĺpce s príznakmi (uložené ako features) a výstupná množina bude obsahovať zhluky. Rozdelenie dát medzi trénovacou a testovacou množinou je 20% (pre trénovaciu množinu) a rozdelenie medzi trénovaciou a validačnou množinou je 50% (pre validačnú množinu).

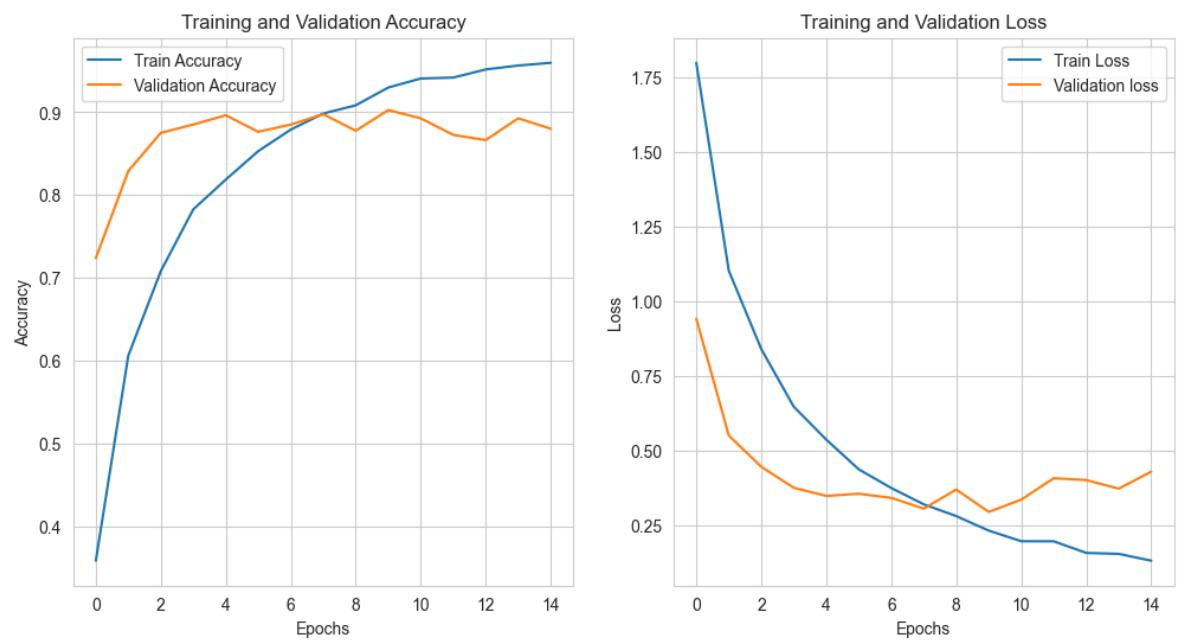
Táto stieť obsahuje vstupnú vrstvu, 3 skryté vrstvy a 1 výstupnú vrstvu:

- **Vstupná vrstva:** Obsahuje počet stĺpcov v množine X
- **1. skrytá vrstva:** 128 neurónov (dropout: 30%)
- **2. skrytá vrstva:** 64 neurónov (dropout: 20%)
- **3. skrytá vrstva:** 32 neurónov (dropout: 20%)
- **Výstupná vrstva:** Obsahuje počet hodnôt množiny Y

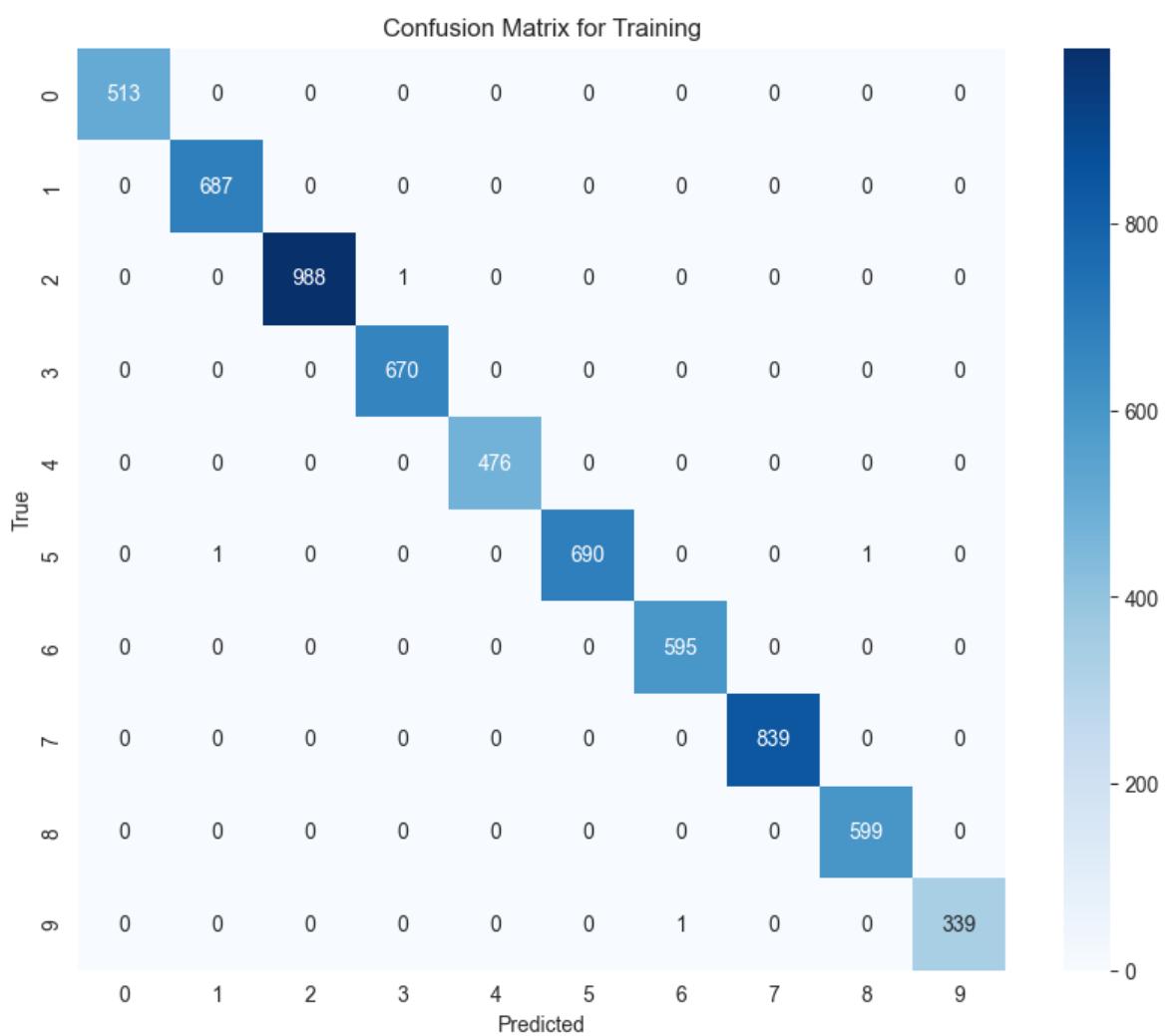
Nasledovný zoznam zobrazuje výsledky trénovania, testovania a validácie dát:

- **Tréning:**
 - *Presnosť:* 1.00
 - *Strata:* 0.02
- **Validácia:**
 - *Presnosť:* 0.88
 - *Strata:* 0.35
- **Testovanie:**
 - *Presnosť:* 0.88
 - *Strata:* 0.35

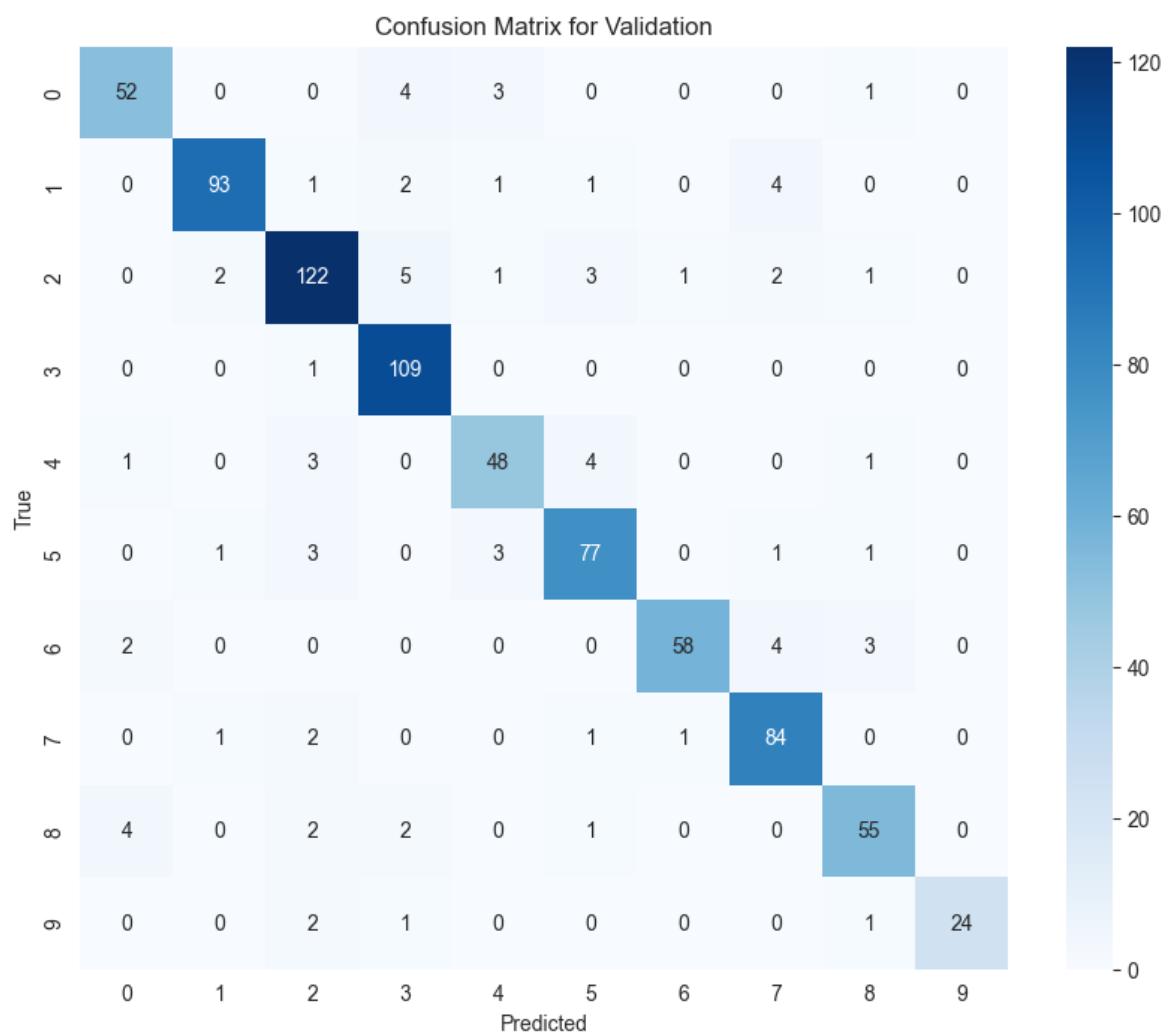
Nižšie taktiež zobrazíme priebeh trénovania a testovania, vrátane konfúznych matíc pre trénovaciu, validačnú a testovaciu množinu. Ako je možné vidieť nižšie na obrázkoch, trénovanie dopadlo najlepšie pre tento model.



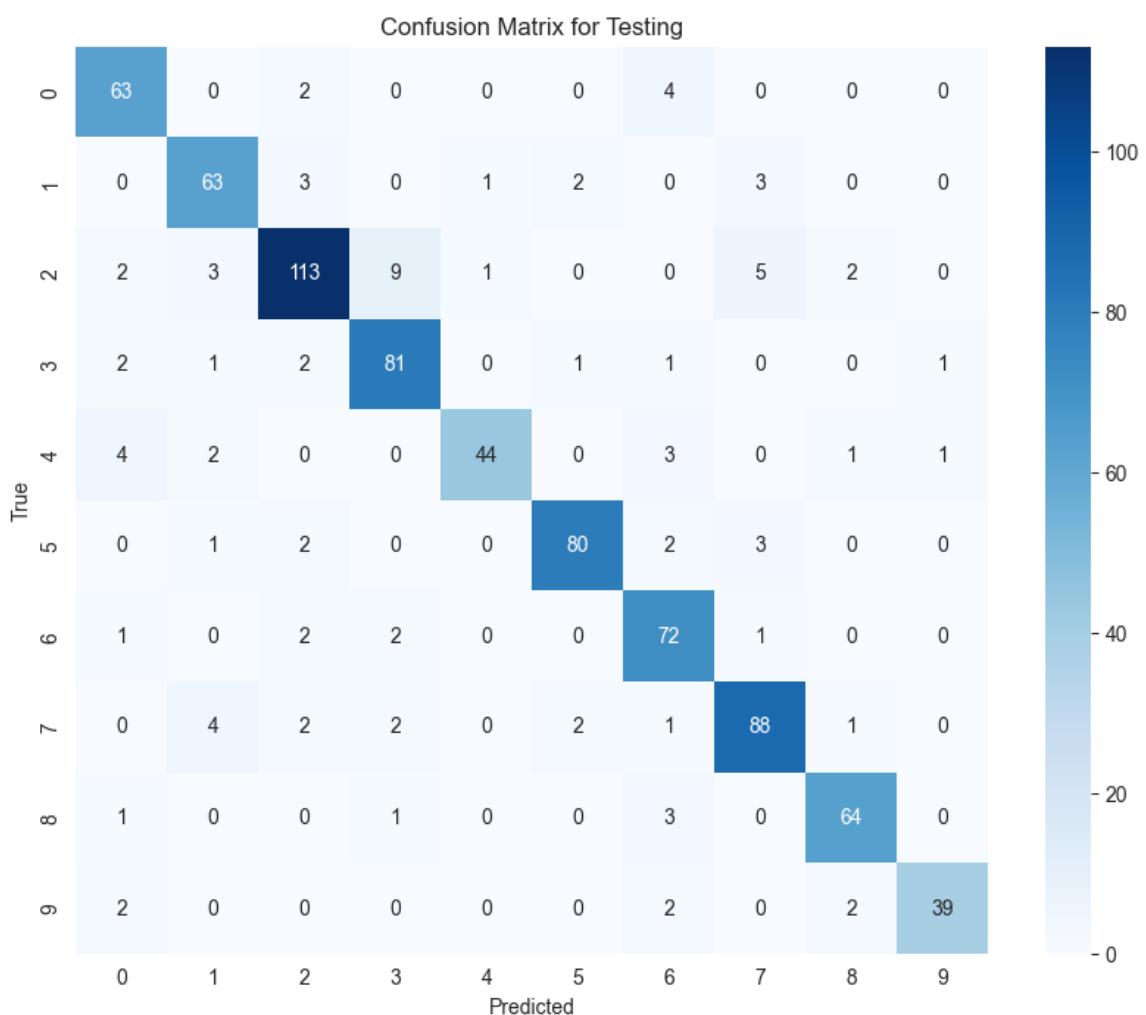
Obrázok 33: Trénovací a testovací priebeh neurónovej siete pre vybrané zhluky



Obrázok 34: Konfúzna matica pre trénovaciu množinu



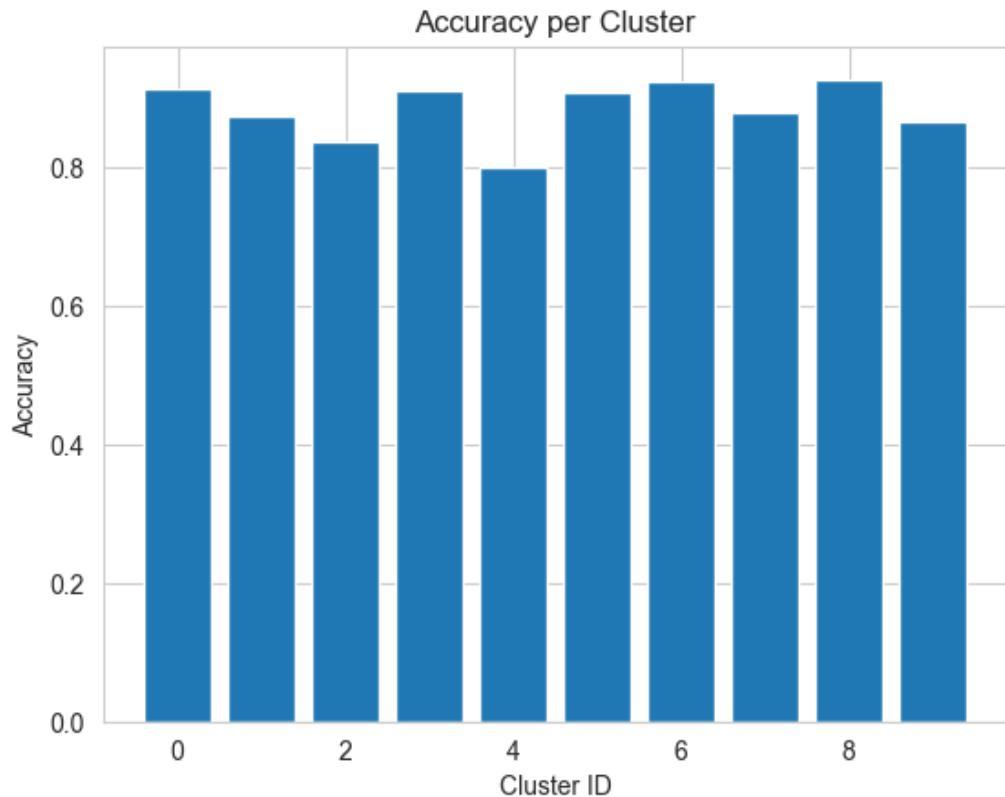
Obrázok 35: Konfúzna matica pre validačnú množinu



Obrázok 36: Konfúzna matica pre testovaciu množinu

Zoznam všetkých úspešností pre jednotlivé zhluky je nasledovný (zhluky sú očíslované od 1 - 10, avšak v tabuľke sú od 0 - 9):

1. **Úspešnosť pre zhluk č.1:** 0.9130
2. **Úspešnosť pre zhluk č.2:** 0.8750
3. **Úspešnosť pre zhluk č.3:** 0.8370
4. **Úspešnosť pre zhluk č.4:** 0.9101
5. **Úspešnosť pre zhluk č.5:** 0.8000 (*najhoršia úspešnosť*)
6. **Úspešnosť pre zhluk č.6:** 0.9091
7. **Úspešnosť pre zhluk č.7:** 0.9231
8. **Úspešnosť pre zhluk č.8:** 0.8800
9. **Úspešnosť pre zhluk č.9:** 0.9275 (*najlepšia úspešnosť*)
10. **Úspešnosť pre zhluk č.10:** 0.8667



Obrázok 37: Grafické znázornenie vyhodnotenia úspešnosti pre jednotlivé zhluky

Záver

Cieľom zadania bolo implementovať program, ktorý klasifikuje obrázky vtáčikov, ktoré boli poskytnuté v rámci datasetu.

Vtáčiky sa nám podarilo klasifikovať s už popísanými úspešnosťami, pričom bolo taktiež možné vidieť porovnanie jednotlivých modelov (vlastnej konvolučnej siete, ImageNet siete a taktiež aj neurónovej siete pre vybrané zhluhy). Tieto modely bolo možné porovnávať prostredníctvom výstupných priebehov trénovania a testovania, vrátane konfúznych matíc pre trénovaciu, validačnú a testovacku množinu.