ML Project

Klára Sládečková

Úvod

Vo svete dentálnej hygieny sa záznamy o pacientoch udržiavajú v rôznych podobách. Jednou z týchto možností sú fotografie ústnej dutiny pacienta z rôznych strán. Táto forma archivovania má viacero výhod, ako napríklad presnejšia a obsiahlejšia dokumentácia pacienta, možná opätovná evaluácia ústnej dutiny, či prípadné možné porovnanie stavu po nejakom čase alebo pred a po liečbe pacienta.

Na druhú stranu udržiavanie niekoľkých tisícok snímok ústnej dutiny pre každého pacienta nie je lacná záležitosť. Taktiež by bolo výhodné pre každú fotografiu poznať jej obsah, teda napríklad vedieť determinovať pohľad, z ktorého bol daný obraz generovaný. Motiváciou tohto projektu je práve určenie pohľadu fotografie zubov pacienta a to využitím strojového učenia.

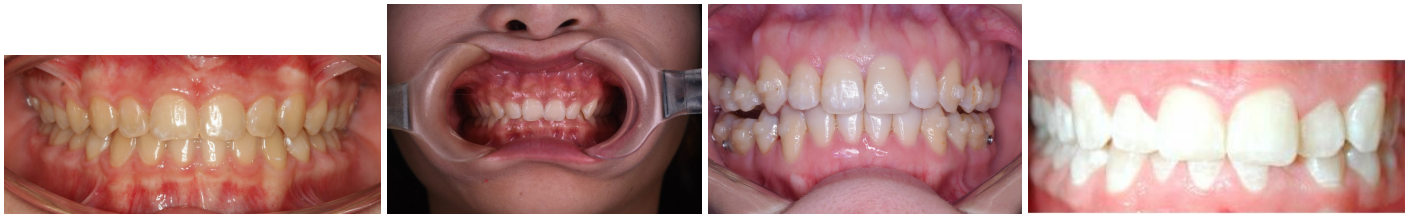
Dáta

Dáta pozostávajú z fotografií ústnej dutiny pacientov. Fotografie boli nasnímané z piatich pohľadov - spredu, zľava, sprava, zhora a zdola. Jednotlivé pohľady sú už utriedené v piatich priečinkoch ('front', 'left', 'lower', 'right', 'upper'). Jednému pacientovi prislúchajú občas aj viaceré snímky z toho istého pohľadu, sú označené ako pred a po liečbe prípadne iným popiskom rozdielu fotografií. Tieto snímky nie sú na prvý pohľad viac podobné ako iné, preto ich v riešení považujeme za nezávislé vzorky.

Dohromady máme teda 3926 záznamov "front", 3917 záznamov "left", 3915 záznamov "lower", 3913 "right" a 3904 záznamov "upper", čo je dokopy 19 575 vstupných dát.

Príklady vstupov môžme vidieť na nasledovnej strane. V každom riadku sú štyri vzorky pre všetky triedy pohľadov.

Front



Left

Lower

Right

Upper

Riešenie

Predspracovanie vstupu

Problém je zadaný nasledovne: pre vstupný obrázok identifikovať jeden z piatich pohľadov, z ktorého bol daný snímok odfotený.

Po načítaní obrázkov si vstupné dáta najprv vyžadovali predspracovanie - odstránenie nekonzistentných dát (tie už nie sú započítanie do konečnej sumy uvedenej vyššie), prestavenie rôznorodej veľkosti obrázkov na jednotnú (konkrétne 96×96×3) a preformulovanie jednotlivých snímkov do vektorovej podoby pomocou funkcie $flatten()$, ktorá je prístupná v balíčku `PIL.Image`.

Následne sa dáta rozdelili na trénovaciu a testovaciu množinu v pomere 6:1. Nakoniec sme vstupné dáta náhodne poprehadzovali.

Definovanie modelu

Ako model sme použili konvolučnú sieť. Konvolučná sieť je zostavená z neurónov, ktoré sú schopné učeniu váh a odchýlok. Každý neurón dostane určité vstupy a na nich vykoná operácie typu skalárny súčin, prípadne iné nelineárne operácie. Celkovo neurónová sieť vyjadruje jednu diferencovateľnú funkciu udávajúcu skóre. Na koniec konvolučnej siete prichádza plne prepojená sieť (napr. SVM/Softmax), ktorá tiež špecifikuje stratovú funkciu.

Výhodou tohto typu modelu je predpoklad, že vstup je obrázok, čo sa vzťahuje aj na naš prípad. Tento predpodklad umožňuje zakódovanie určitých vlastností vstupných dát do architektúry modelu, čo vedie k väčšej efektivite funkcií a tiež znižuje množstvo parametrov v sieti.

Pri budovaní modelu sme využili tri typy vrstiev konvolučnej siete a to konvolučnú, pooling a plne-prepojené vrstvy.

Konvolučná vrstva

Konvolučná vrstva vyprodukuje výstup neurónov, ktoré sú lokálne prepojené so vstupným regiónom. Každý neurón vypočíta skalárny súčin medzi váhami a malou vstupnou oblasťou, ku ktorej je pripojený.

Nastavením rôznej aktivácie tejto vrstvy dosiahneme aj nelineárne funkcie počítajúce neuróny (napr. aktivácia `RELU` vypočíta $max(0, x)$ pre threshold 0). Táto aktivácia nemení veľkosť rozmerov.

Pooling vrstva

Táto vrstva predstavuje redukujúcu operáciu, čo sa týka priestorových rozmerov (výška, šírka), pričom rozsah tejto redukcie záleží na veľkosti filtra. Čím väčšia miera filtra, tým redukujúcejšia je táto vrstva.

V našej aplikácii sme používali filtre veľkosti 2×2.

Plne-prepojená vrstva

Na záver konvolučnej siete sa pridávajú tzv. plne-prepojené vrstvy, ktorých výpočet uzatvorí celú sieť. Výsledkom je skóre pre každú výstupnú triedu. V tejto vrstve sú neuróny poprepájané každý s každým, ako to vyplýva aj z názvu.

Celý model teda vyzerá nasledovne.

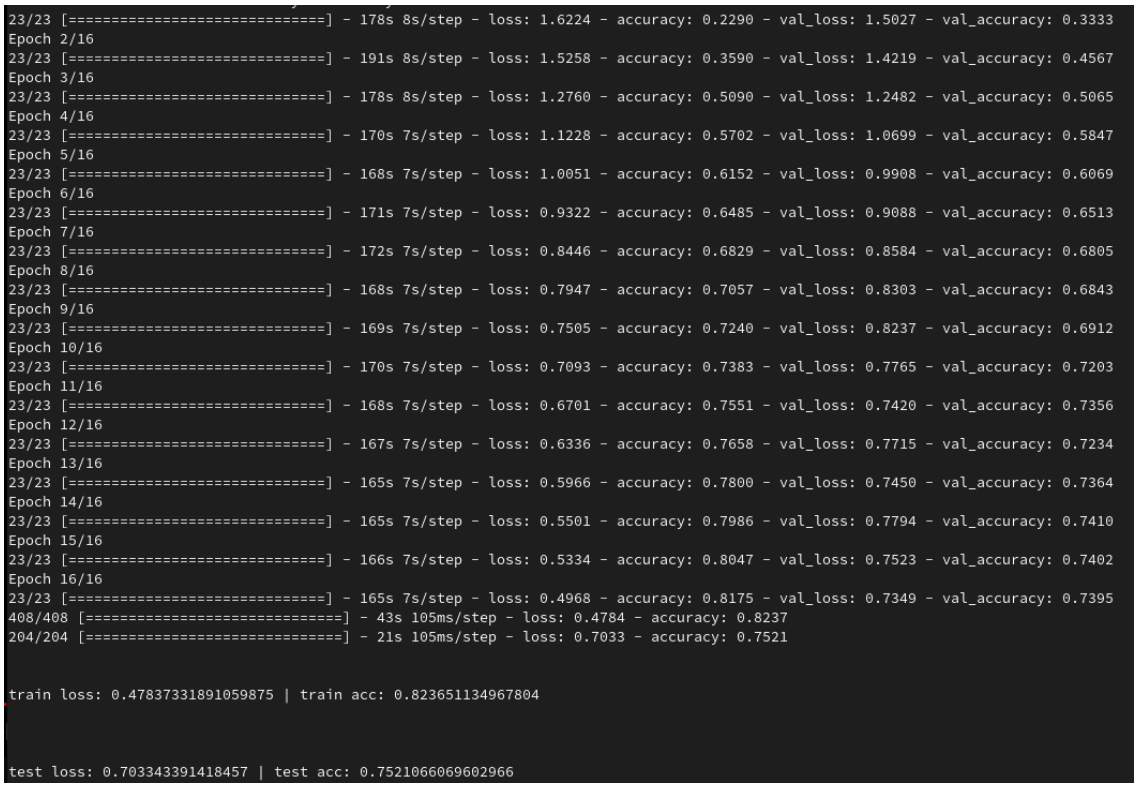
- konvolučná vrstva s aktiváciou `RELU`, 32 filtrov o veľkosti 3×3, padding
- konvolučná vrstva s aktiváciou `RELU`, 64 filtrov o veľkosti 5×5
- konvolučná vrstva s aktiváciou `RELU`, 128 filtrov o veľkosti 3×3, padding
- pooling vrstva s filtrom veľkosti 2×2
- konvolučná vrstva s aktiváciou `RELU`, 32 filtrov o veľkosti 3×3, padding
- konvolučná vrstva s aktiváciou `RELU`, 64 filtrov o veľkosti 5×5
- konvolučná vrstva s aktiváciou `RELU`, 128 filtrov o veľkosti 3×3, padding
- pooling vrstva s filtrom veľkosti 2×2
- dense vrstva s aktiváciou `RELU`, výstup veľkosti 256
- dense vrstva s aktiváciou `RELU`, výstup veľkosti 256
- dense vrstva s aktiváciou `SOFTMAX`, výstup veľkosti 5

Zvolili sme sekvenčný typ konvolučnej siete, čo umožnilo postupné pridávanie vrstiev do modelu. Ďalšie dva cykly vrstiev pozostávali z troch konvolučných vrstiev s aktiváciou `RELU`, každá z nich s iným počtom filtrov rôznej veľkosti. Dve z týchto vrstiev aplikovali na vstup vypchávku, čím sa zachovalo viac informácie. Na koniec týchot cyklov prišla pooling vrstva, ktorá zredukovala priestorové rozmery. Tým sa výpočet urýchlil a zovšeobecnil.

Nakoniec sme pridali niekoľko hustých vrstiev s aktiváciou `RELU` a plne-prepojenú vrstvu s aktiváciou `SOFTMAX`, ktorej výsledkom bolo 5 rôznych tried pre klasifikáciou (pre každý pohľad jedna trieda).

Analýza výsledkov

Zobrazením histórie fitovania modelu môžme nahliadnuť na jeho kvalitu.



Pre lepší náhľad tiež uvádzame výsledky niektorých mierne pozmenených modelov.

1. rozdelenie train:test dát v pomere 2:1, počet epoch=16, height, weight = 48
   val_loss = 0.4783, val_accuracy = 0.8236, test_loss = 0.7033, test_accuracy = 0.7521

2. rozdelenie train:test dát v pomere 2:1, počet epoch=30, height, weight = 48
   val_loss = 0.2364, val_accuracy = 0.9373, test_loss = 1.5063, test_accuracy = 0.7352

3. rozdelenie train:test dát v pomere 3:1, počet epoch=30, height, weight = 48
   val_loss = 0.2685, val_accuracy = 0.9215, test_loss = 1.2651, test_accuracy = 0.7224

4. rozdelenie train:test dát v pomere 6:1, počet epoch=30, height, weight = 96
   val_loss = 0.3123, val_accuracy = 0.9056, test_loss = 1.2585, test_accuracy = 0.7426

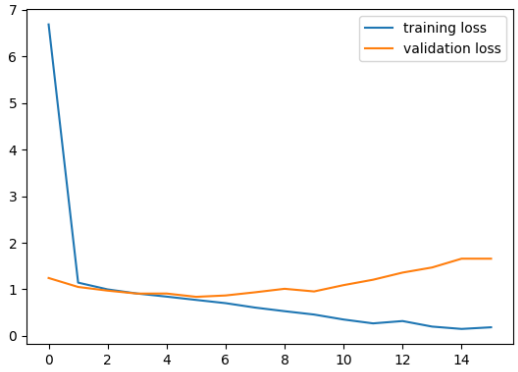
5. rozdelenie train:test dát v pomere 9:1, počet epoch=30, height, weight = 96
   val_loss = 0.3358, val_accuracy = 0.9411, test_loss = 2.5820, test_accuracy = 0.6229

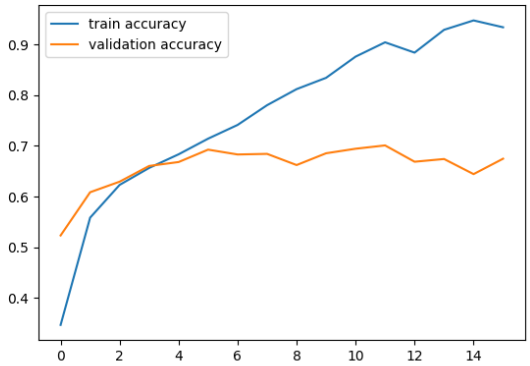
6. rozdelenie train:test dát v pomere 7:1, počet epoch=30, height, weight = 96, bez prvej dense layer
   val_loss = 1.6091, val_accuracy = 0.1990, test_loss = 1.6095, test_accuracy = 0.1891

7. rozdelenie train:test dát v pomere 7:1, počet epoch=36, height, weight = 96, s oboma dense layer
   val_loss = 0.0608, val_accuracy = 0.9798, test_loss = 2.6471, test_accuracy = 0.6229

Väčší testovací dataset prináša presnejšiu testovaciu chybu, ale horšiu trénovaciu presnosť. Zväčšenie rozmerov obrázkov pomohlo k presnejšiemu modelu. Zvýšenie počtu epoch sa nejako zásadne neodzrkadlilo, v posledných etapách sa výsledné skóre menilo len nebadateľne. Odstránenie dense layer prinieslo očividné zhoršenie kvality modelu.

Ku koncu sme pridali možnosť nahliadnuť kvalitu modelu so šumivým vstupom. Funkcia `generuj_sum` vygeneruje náhodný šum na testovacích dátach, na ktorých potom prebehne analýza testovacej chyby. Postupné skóre možno vidieť na obrázkoch.





Záver

Vo všeobecnosti je pravdepodobnosť, že sa model trafí približne 20%, čo je značný skok oproti výsledkom nášho modelu. Taktiež pri náhľade na vstupné dáta je badateľná náročnosť rozoznať triedu "left" a triedu "right" a niekedy aj triedu "front". Vzhľadom na tieto fakty si model počínal celkom dobre.

Na druhú stranu má model značne vysokú testovaciu stratu a pri niektorých variantoch modelu vidíme veľký rozdiel medzi validačnou stratou a testovacou stratou, ako aj medzi validačnou a testovacou presnosťou. Toto poukazuje na overfittovanie modelu, čo môže byť spôsobené viacerými faktormi. Napríklad zvoleným pomerom trénovacích a testovacích dát alebo príliš komplikovanou štruktúra konvolučnej siete. Komplikovanosť modelu sa dá riešiť rôznymi spôsobmi - zmenou počtu vrstiev alebo počtu filtrov, počiatočným nastavením váh, regularizáciou, atď. Konkrétne sme vyskúšali nastavenie počiatočných váh využitím `He initialization`. Táto zmena ale nemala priaznivé účinky, testing presnosť sa znížila a test error sa výrazne zvýšil.

Súhrnom, model prejavuje pomerne dobré výsledky, prípadné zlepšenia by boli v budúcnosti zamerané na zjednodušenie modelu a tak zvýšenie testovacej presnosti.

Existujúci kód nájdete na https://github.com/SladeckovaKlara/ML-project.