Vysoké učení technické
v Brně
Fakulta informačních
technologií

Strojové učení a rozpoznávání
Dokumentace projektu

30. dubna 2022

xhrani02

1 Úvod

Cílem projektu bylo vytvořit a natrénovat detektor jedné osoby z obrázku obličeje a hlasové nahrávky. Pro tento účel byly vytvořeny a natrénovány dva klasifikátory.

2 Obecné požadavky pro spuštění projektu

Projekt byl implementován v jazyce Python 3.7.4. Knihovny potřebné k zprovoznění projektu jsou uvedeny v souboru `requirements.txt`, ve složce se zdrojovými soubory. Složka projektu obsahuje zároveň několik složek, jejichž význam je následující:

- `data/` – předpokládá se, že složka obsahuje složky jednotlivých datasetů, tak jak byly postkytnuty, tzn. složky `eval`,`non_target_dev`, `non_target_train`, `target_dev` a `target_train`
- `src/` – složka se zdrojovými soubory
  - `models/` – složka obsahuje námi natrénované modely. Dále slouží pro ukládání nových modelů.
  - `features/` – složka obsahuje objekty Numpy.Array uložené na disk, které obsahují příznaky zvukových nahrávek
  - `out/` – složka slouží pro ukládání výstupu evaluace

3 Detekce ze snímku obličeje

Pro detekci obličeje jsme vytvořili konvoluční neuronovou síť pomocí knihovny PyTorch.

3.1 Průběh implementace

Prvotní snahou bylo vytvořit konvoluční neuronovou síť s třemi konvolučními a dvěma plně propojenými vrstvami, na výstupu této síťe byla umístěna aktivační funkce sigmoida, kde její hodnota je interpretována jako pravděpodobnost, že obličej patří hledané osobě. Architekturou jsme se inspirovali u sítí jako VGG-16 [4] a AlexNet [3]. Jakožto loss funkci jsme zvolili Binární cross entropii. Jako učicí algoritmus jsme zvolili Adam.

Síť s touto architekturou však při tréninku nesnižovala hodnotu loss funkce, byly proto přidány normalizační vrstvy.

I přes přidání normalizačních vrstev a úpravu parametrů sítě nebylo dosaženo požadované generalizace modelu, síť si vždy perfektně zapamatovala trénovací data. Změnili jsme proto architekturu sítě na síť ze článku [5].

Zároveň jsme zvětšili trénovací sadu pomocí transformací rotace, zvětšení a také jsme přidali gaussovský šum, tyto úpravy jsou implementovány ve skriptu

augment_image_dataset.py. Při trénovaní této sítě jsme navíc využili techniky early stopping, kdy jsme již při trénování modelu zároveň testovali na validačním datasetu a jakmile se přesnost přestala zvyšovat ukončili jsme trénování. Tímto způsobem jsme nakonec natrénovali model, který dosahuje přesnosti 88% na validačním datasetu.

3.2 Zprovoznění implementace

Detektor obličeje je implementován v jediném souboru, a to `image_nn.py`. Ten se nachází v adresáři `src/` projektu. Skript přijímá několik parametrů, všechny jsou vypsány na standardní výstup při spuštění skriptu s přepínačem `-h`. Jsou to parametry `--mode [MODE]`, který může nabývat hodnot:

- "tv"– defaultní hodnota tohoto paremetru, který spustí trénování a poté validaci.
- "t"– skript provede pouze trénování modelu
- "v"– skript provede pouze validaci modelu, tento parametr musí doprovázet parametr `--loadmodel`, viz. dále.

parameter `--loadmodel [MODEL]`, kde `[MODEL]` je název modelu bez koncovky. Skript hledá modely ve složce `/src/models`. A parametr `--savemodel [MODEL]`. Pokud je tento parametr uveden, je na konci trénování model uložen s názvem `[MODEL]` do složky `/src/models`.

Pro natrénování modelu a jeho uložení je skript možno spustit následovně
`python image_nn.py --savemodel=first_model`

4 Detekce z hlasové nahrávky

Pro detekci hlasu byla vytvořena neuronová síť s dvěma konvolučními a čtyřmi plně propojenými vrstvami pomocí knihovny Pytorch v jazyce Python.

4.1 Průběh implementace

Pro detekci hlasu bylo z nahrávek potřeba extrahovat příznaky. Kromě MFCC koeficientů, které extrahuje funkce `wav16khz2mfcc` knihovny `ikrlib.py` používáme navíc následjící příznaky: chromagram, mel-spektrogram, spektrální kontrast a příznaky tonálního těžiště. Extrakce těchto příznaků je převzata z článku [1].

Po trénovaní a validaci na několika architekturách byla zvolena architektura obsahující dvě 1D konvoluční vrstvy na začátku sítě a čtyři plně propojené vrstvy síť byla trénovaná na originálním targev_train a non_target_train datasetu v průběhu padesáti epoch. Jakožto optimalizační algoritmus byl zvolen stochastic gradient descent a v průběhu trénování byla kontrolována jak hodnota loss funkce, tak přesnost na validačním datasetu. Takto natrénovaná síť dosahovala přesnosti 94 % na validačním datasetu.

4.2 Zprovoznění implementace

Detektor hlasu je implementován v jediném souboru, a to `voice_nn.py`. Ten se nachází v adresáři `src/` projektu. Skript obsahuje totožné parametry jako detekor obličeje, ty je popřípadě taktéž možno vypsat přepínačem `-h`.

Pro natrénování modelu a jeho uložení je skript možno spustit následovně:
`python voice_nn.py --savemodel=first_model`

Narozdíl od detekce z obrázku je potřeba u hlasu extrahovat příznaky. Tato operace může být v závislosti na počtu dat časově náročná. Po prvním spuštění skriptu je do adresáře `src/features/` uložen `Numpy.Array` obsahující extrahované příznaky. Při dalším spuštění si skript příznaky jednoduše načte.

5 Detekce na evaluačního datasetu

Výstupem projektu jsou dva detektory, jeden pro obličej a druhý pro hlas hledané osoby. Pro detekci na evaluačním datasetu byla použita kombinace obou detektorů. Pro jejich spojení bylo využito sum rule z článku [2], ve kterém jsou shrnuty metody kombinace klasifikátorů. Mimo jiné v článku řeší téměř totožný problém, ve kterém mají autoři k dispozici fotku osoby z dvou různých úhlu a její hlasovou nahrávku, přičemž se snaží o detekci hledané osoby. Na každou úlohu používají jiný klasifikátor, podobně jako v našem případě. Ve výsledcích autoři uvádějí, že nejlepší výsledky vykazovalo použití sum rule.

V projektu se nám ale nepovedlo natrénovat dobře generalizující detekor obličeje a kombinací klasifikátorů se přesnost spíše zhoršila. Odevzdáváme proto i výsledky klasifikátorů samostatně.

5.1 Zprovoznění implementace

Detekce na evaluačním datasetu je implementována ve skriptu validate.py. Přijímá následující parametry. Volba módu `--mode [v,i,vi,all]`, kde `v` je zkratka pro `voice` a `i` pro image. Volba `vi` provede detekci na kombinaci klasifikátorů. Možnost `all` pak provede všechny tři zmíněné volby. Výsledky jsou, ve formátu specifikovaném v zadání, uloženy do složky `src/out`. Volba modelu pro detekci hlasu `--voice=[VOICE]` a volba modelu pro detekci obličeje `--image=[IMAGE]`, kde `[VOICE]` a `[IMAGE]` jsou názvy modelů ve složce `src/models`.

6 Limity

Natrénovaný detektor obrazu negeneralizoval příšil dobře. Větší dataset by problém vylepšil, jako jedno z možných vylepšení se nabízí využití trénovací techniky cross-validation, která dovoluje trénovat na celém datasetu. Model by se tak mohl lépe naučit generalizaci. Druhým možným vylepšením se nabízí použít techniku data augmentation pro zvukové nahrávky.

Reference

[1] Jurgen Arias. How to build a neural network for voice classification — by jurgen arias — towards data science. https://towardsdatascience.com/how-to-build-a-neural-network-for-voice-classification-5e2810fe1efa, Květen 2020. Zobrazeno 30.04 2022.

[2] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[4] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

[5] Fadhlan Zaman. Gender classification using custom convolutional neural networks architecture. *International Journal of Electrical and Computer Engineering (IJECE)*, 10:5758, Prosinec 2020.