Čtení textu z memů pomocí OCR

Vojtěch Jedlička
ČVUT–FIT
jedlivo1@fit.cvut.cz

7. ledna 2023

1 Úvod

Memy jsou ve 21. století velkou součástí naší kultury. Podle Wikipedie [12] se jedná o kulturní obdobu genu - replikující se kulturní informaci. Pokud mem chceme s někým sdílet, většinou to uděláme pomocí sociálních sítí. Co když ale chceme rychle nějaký mem najít? S tím si Google občas nemusí poradit. Nebo co když si chceme prohlížet memy s určitým obsahem, ať už textovým, nebo obrázkovým? Neexistuje žádná kvalitní služba, která by uměla vyhledávat memy pomocí textu, kategorií, nebo původních obrázkových šablon.

Tato semestrální práce je potřebnou součástí takové služby - pomocí OCR [10] bude schopna vyhledávat v databázi memů podle textu, který se na nich nachází, a zároveň poté bude možné memy klasifikovat podle textu do různých zájmových kategorií pomocí NLP algoritmů.

2 Očekávaný výstup

Výstupem by měl být Docker server, který pomocí webového API bude přijímat memy, ze kterých přečte text, který vrátí. Důvodem pro kontejnerizaci je možnost škálování na cloudových službách jako je AWS. Databáze memů, kterou jsem pomocí web scrapingu vytvořil, čítá přes milion záznamů a zdaleka není úplná. Na sociálních sítích, např. reddit.com je každý měsíc přes 60 000 nových příspěvků na několika nejznámějších kanálech čistě určených pro memy. Pro ilustraci - pokud by přečíst mem trvalo sekundu, pak by přečtení celé dosavadní databáze trvalo přes deset dnů.

3 Analýza OCR algoritmů

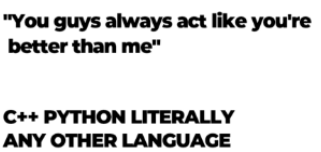
Na analýzu jsem zvolil dvě OCR knihovny: Tesseract [6] a EasyOCR [1]. Dále jsem testoval následující filtry:

- Cannyho hranový detektor [11]
- Cannyho hranový detektor s vyplněnými tvary (vlastní úprava)
- Zaostření (Sharpen, pomocí konvoluce)
- Rozostření (Gaussian blur) [9]
- Binarizaci [8]
- Vlastní filter

Obě knihovny jsou z velké části natrénované na vyfocených dokumentech buď ručně psaných, nebo vytisknutých. Jedná se tedy hlavně o černý text na bílém pozadí. Memy se od dokumentů liší tím, že text je většinou přímo vložený do obrázku, což negativně ovlivňuje úspěšnost správného přečtení textu. Viz 1 a 2

8 & (6 'You guys always act like youre better than me" Ct+ PYHOn LITERALLY ANYOTHER LANGUACE

Obrázek 1: Ukázkový meme (vlevo) a text získaný z EasyOCR po převedení do barev šedi (vpravo).



"You guys always act like youre better than me" C++PYTHON LITERALLY ANY OTHER LANGUAGE

Obrázek 2: Stejný text v obrázku (vlevo) a výsledek z EasyOCR (vpravo).

3.1 Metodika porovnávání

Pro porovnávání jsem použil testovací dataset o 100 memech a vstupní obrázky jsem převedl na černobílé.

Samotný dataset jsem vytvořil pomocí Google Vision API a ručně opravil, čítá 1000 otextovaných memů. Testování probíhalo na menším vzorku kvůli zdlouhavějším výpočtům.

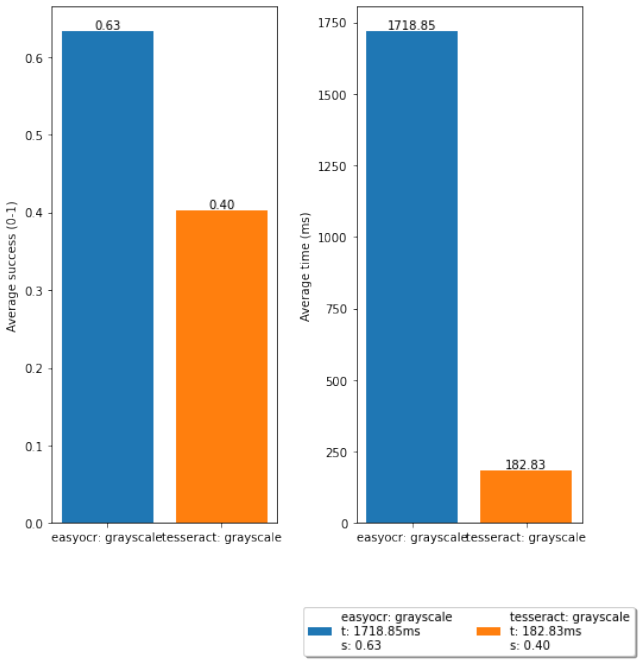
Úspěšnost algoritmu počítám jako procentuální obsazení stejných slov výstupního textu v očekávaném textu (procentuální velikost průniku množin očekávaných a výstupních slov ku velikosti množiny očekávaných slov).

EasyOCR narozdíl od Tesseractu využívá GPU.

Testování probíhalo na počítači s následujícími parametry:

- CPU: Ryzen 3600
- RAM: 32GB 3200MHz
- GPU: GTX 1070
- NVMe SSD

3.2 Porovnání OCR knihoven bez filtrů

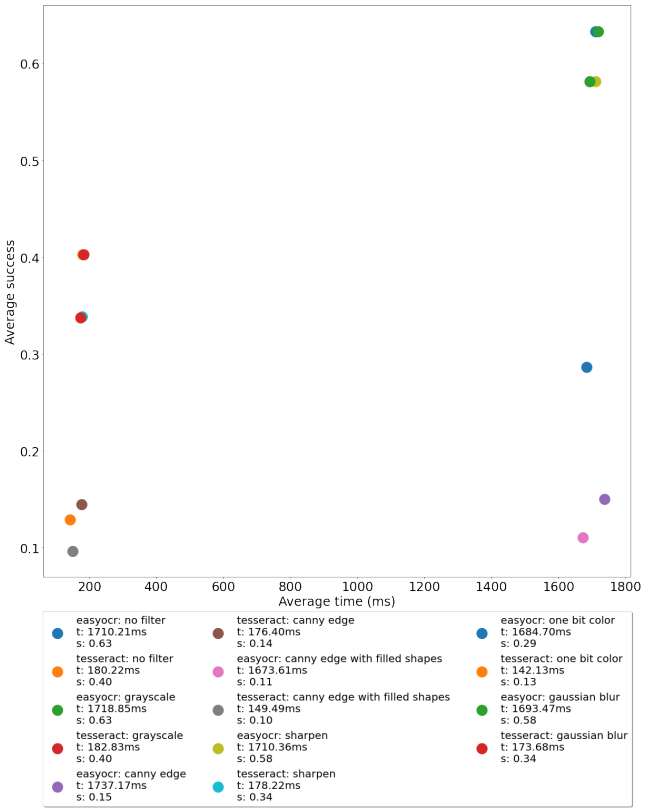


Obrázek 3: Tesseract vs. EasyOCR

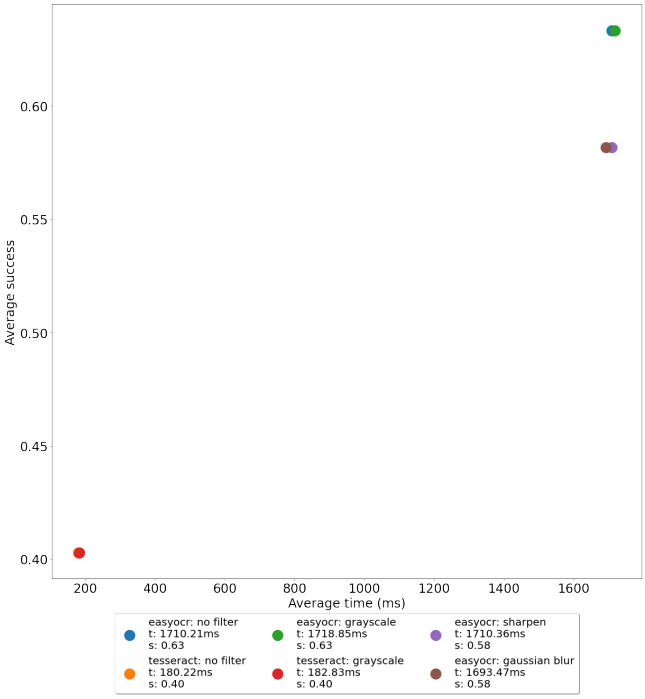
Z grafu 3 jsou patrné rozdíly jak v procentuální úspěšnosti (vlevo), tak rychlosti (v pravo). Tesseract je téměř 10x rychlejší, zato má o 23% menší úspěšnost než EasyOCR.

3.3 Porovnání s filtry

Na grafu 4 jsem vykreslil všechny kombinace OCR knihoven s jedním filtrem. Pojďme se ale zabývat



Obrázek 4: Graf úspěšnosti a rychlosti OCR knihoven se všemi filtry.



Obrázek 5: Původní graf s algoritmy úspěšnějšími než 40%.

grafem kombinací s větší úspěšností než 40%. 5

Co se týče Tesseractu, tak je vidět, že bez filtrů funguje stejně rychle i úspěšně jako s převedením vstupního obrázku do barev šedi. To samé platí o EasyOCR.

Filtry, kromě barev šedi, jako je zaostření, nebo

rozmazání mají na čtení textu negativní dopad. U Tesseractu natolik, že úspěšnost nepřekročí hranici 40% a u EasyOCR se jedná o -5% u zaostření a rozmazání. Ostatní filtry mají velmi malou úspěšnost.

4 Tvorba vlastního filtru

Vlastním filtrem, specificky vytvořeným na předzpracování memů, jsem si sliboval zvýšení úspěšnosti.

4.1 Odstranění zbytečných pixelů

Odstranění zaručeně netextových částí memu by mohlo algoritmům nejen zvýšit jejich úspěšnost, ale i zvýšit rychlost.

Samotný algoritmus pomocí Cannyho hrany s vyplněnými tvary vytvoří masku, která poté slouží k odstranění pixelů mimo vyplněné tvary. Tím se zachovají pixelově rozmanitější části obsahující text a odeberou části jako je výplň obličejů, předmětů nebo pozadí.

Bohužel z grafu 7 vidíme, že rychlost zůstává prakticky stejná a účinnost klesla o 8% u obou knihoven po odstranění zbytečných pixelů.

4.2 Použití dalších filtrů

Autoři Tesseract i EasyOCR ve svých implementacích používají vlastní filtry, které zlepšují jejich účinnost čtení textu pro obecně vypadající obrázky, a zřejmě odvedli dobrou práci. [2, 7]. Všechny další pokusy s kombinováním filtrů, odstraňováním pixelů nebo tvorbou vlastních filtrů dopadly neúspěšně. Při jejich tvorbě jsem čerpal z [4, 5, 3]

5 API Server

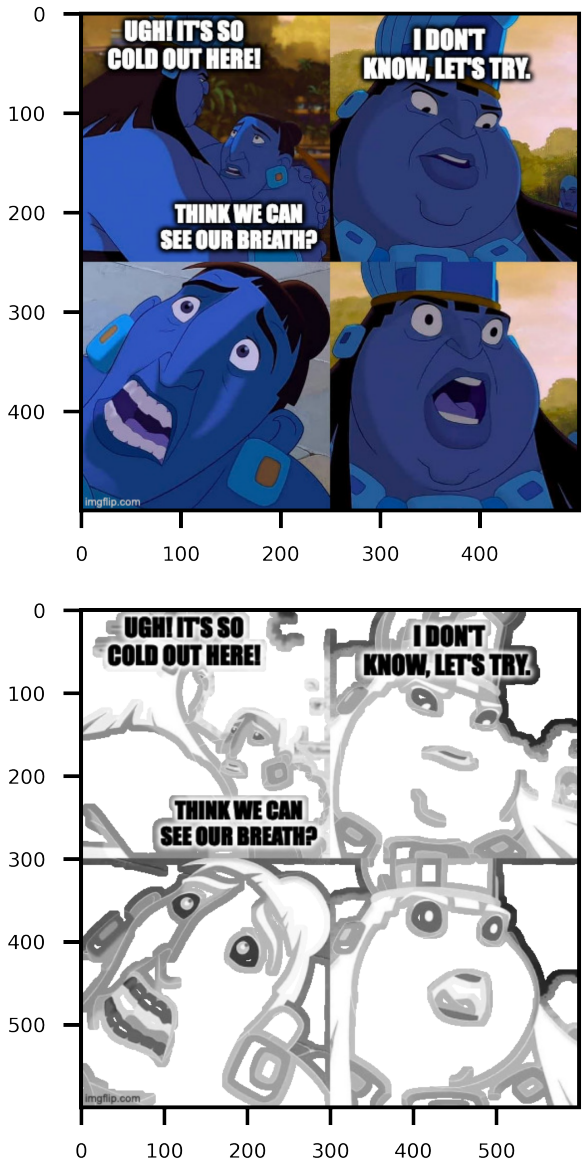
Výsledná kontejnerizovaná aplikace má dva endpointy pro použití EasyOCR i Tesseractu a nepoužívá žádné filtry, a to kvůli jejich malé úspěšnosti. Zároveň je vyžadován autentifikační klíč, který lze nastavit pomocí proměnné v prostředí pro navýšení bezpečnosti.

Oba endpointy jsou typu POST a vyžadují tělo požadavku ve formě ‘form-data‘ a klíč ‘file‘ s nahrávaným obrázkem.

Více o použití serveru je v dokumentu Readme.md

6 Další rozvoj - zlepšení přesnosti

Obě knihovny umožňují přetrénování na vlastních trénovacích datech, což by velmi pravděpodobně zvýšilo jejich úspěšnost. Testovací datová sada pro analýzu algoritmů momentálně čítá okolo 1000 přečtených memů, což by vytvořilo násobně více trénovacích dat pro jejich učení, protože je potřeba



Obrázek 6: Ukázka odstranění zbytečných pixelů.

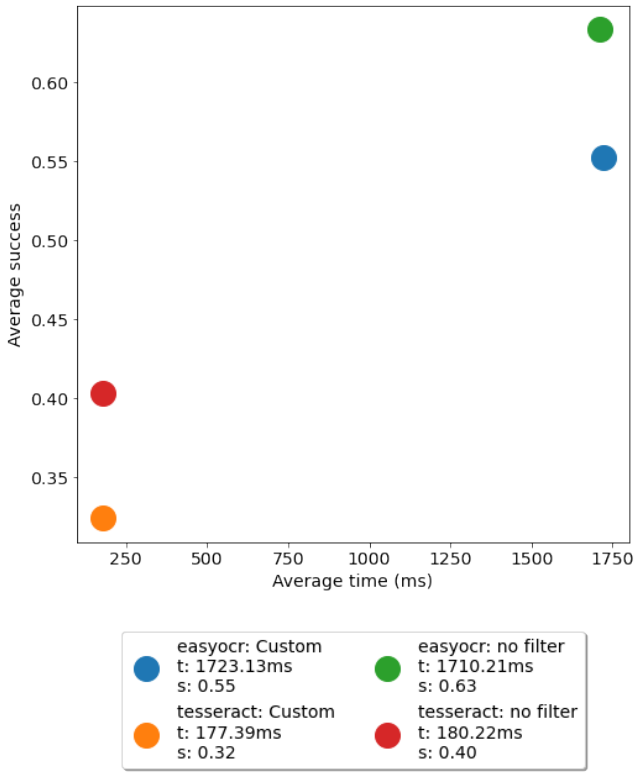
dodat pouze výřez obrázku, který obsahuje jedinou řádku textu, a těch je v každém memu (s textem) většinou více než jeden. Rozšíření datasetu, co se týče množství memů, je možné použitím stejného Jupyter notebooku, jakým byl původně vytvořen a dodáním Google API klíče.

7 Využití

Výsledná aplikace bude použita pro otextování velké databáze memů, díky čemuž bude možné vytvořit webovou aplikaci pro vyhledávání právě v této databázi pomocí klíčových slov nebo fulltextově.

8 Závěr

Vytvoření algoritmu, který by zvyšoval účinnost oproti samostatnému použití jedné z OCR knihoven,



Obrázek 7: Účinnost a rychlost vlastního algoritmu po odstranění zbytečných pixelů.

se nepodařilo. Ukázalo se, že obě knihovny obsahují na míru ušité filtry a používají lepší předzpracování obrázků, než jsem byl schopen vymyslet. Nicméně vytvořit server, který dokáže číst text z obrázků se povedlo i přes relativně malou úspěšnost (63% pro EasyOCR, 40% pro Tesseract) vůči datasetu vytvořenému pomocí Google Vision API. Server také nabízí možnost využití obou knihoven a s malou úpravou v nastavení Docker image lze použít i hardwarovou akceleraci na grafických kartách u EasyOCR, bez které je na mém počítači zhruba 2 - 3x pomalejší.

Reference

[1] Open Source JaidedAI. Easyocr knihovna. online, 2023. [cit. 2023–01–05] https://github.com/JaidedAI/EasyOCR.

[2] Open Source JaidedAI. Easyocr pipeline. online, 2023. [cit. 2023–01–05] https://github.com/JaidedAI/EasyOCR#implementation-roadmap.

[3] docparser.com Kate Meda. Improve ocr accuracy with advanced image preprocessing. online, 2020. [cit. 2023–01–07] https://docparser.com/blog/improve-ocr-accuracy.

[4] Stack overflow contributors. Advise filters to improve text visibility on photo — stack overflow. online, 2017. [cit. 2023–01–07] https://stackoverflow.com/questions/35276538/advise-filters-to-improve-text-visibility-on-

[5] Stack overflow contributors. Prepare image for ocr — stack overflow. online, 2020. [cit. 2023–01–07] https://stackoverflow.com/questions/55254270/prepare-image-for-ocr.

[6] Open Source Ray Smith. Tesseract. online, 2023. [cit. 2023–01–05] https://tesseract-ocr.github.io.

[7] Open Source Ray Smith. Tesseract pipeline. online, 2023. [cit. 2023–01–05] https://tesseract-ocr.github.io/docs/das_tutorial2016/2ArchitectureAndDataStructures.pdf.

[8] Wikipedia contributors. Binary image — Wikipedia, the free encyclopedia, 2022. [Online; accessed 7-January-2023].

[9] Wikipedia contributors. Gaussian blur — Wikipedia, the free encyclopedia, 2023. [Online; accessed 7-January-2023].

[10] Wikipedie. Optické rozpoznávání znaků — wikipedie: Otevřená encyklopedie, 2019. [Online; navštíveno 7. 01. 2023].

[11] Wikipedie. Cannyho hranový detektor — wikipedie: Otevřená encyklopedie, 2021. [Online; navštíveno 7. 01. 2023].

[12] Wikipedie. Mem (informace) — wikipedie: Otevřená encyklopedie, 2022. [Online; navštíveno 7. 01. 2023].