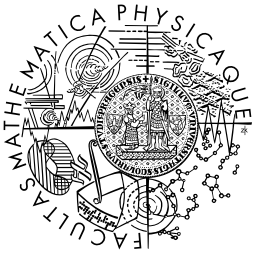


MATEMATICKO-FYZIKÁLNÍ FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Šárka Uramová

Vyhodnocování překladu textů v obrázcích

Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: Mgr. Michal Novák, Ph.D.
Studijní program: Informatika

Praha 2024

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ....

Podpis autora

Chtěla bych tímto poděkovat vedoucímu práce, Mgr. Michalu Novákovi, Ph.D., za cenné rady, pravidelné konzultace, ochotu a celkově příjemnou spolupráci.

Název práce: Vyhodnocování překladu textů v obrázcích

Autor: Šárka Uramová

Ústav: Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: Mgr. Michal Novák, Ph.D., Ústav formální a aplikované lingvistiky

Abstrakt: Tato práce se zaměřuje na vyhodnocení nástrojů pro překlad textu v obrázcích. Klíčovým úkolem bylo vytvoření datové sady obsahující obrázky s textem v různých jazycích, které jsou vzájemnými překlady. Data byla získána z veřejně dostupných internetových obrázků a systematicky strukturována. Součástí práce bylo také vytvoření evaluačního skriptu pro testování správnosti jednotlivých kroků překladového nástroje. Vyhodnocení bylo provedeno s využitím odpovídajících metod a metrik. Dále jsme vyvinuli základní nástroj pro překlad textu v obrázcích. Tento nástroj přijímá jako vstup obrázek s textem a požadovaný cílový jazyk, do kterého má být text přeložen. Nástroj provede detekci textu, jeho překlad a vykreslení přeloženého textu zpět do obrázku.

Klíčová slova: evaluace, parsování SVG, vykreslování SVG, strojový překlad, OCR

Title: Evaluation of text translation in images

Author: Šárka Uramová

Institute: Institute of Formal and Applied Linguistics

Supervisor: Mgr. Michal Novák, Ph.D., Institute of Formal and Applied Linguistics

Abstract: This work focuses on evaluating tools for translating text in images. The key task was to create a dataset containing images with texts in various languages, which are the translations of each other. The data was obtained from publicly available internet images and systematically structured. The work also involved creating an evaluation script to test the accuracy of individual steps of the translation tool. The evaluation was carried out using appropriate methods and metrics. Furthermore, we developed a basic tool for translating text in images. This tool accepts an image with texts and the desired target language as input. The tool performs text detection, translation, and renders the translated text back into the image.

Keywords: evaluation, SVG parsing, SVG rendering, machine translation, OCR

Obsah

1 Úvod **3**
- 1.1 Motivace . . . 3
- 1.2 Členění práce . . . 5
  - 1.2.1 Dataset . . . 5
  - 1.2.2 Evaluační metody . . . 5
  - 1.2.3 Překladový nástroj . . . 5

2 Použité nástroje a metriky **7**
- 2.1 SVG . . . 7
- 2.2 Tesseract . . . 7
- 2.3 Inkscape . . . 9
- 2.4 Míra chybovosti znaků . . . 9
- 2.5 M2M100 . . . 9
- 2.6 BLEU . . . 10
- 2.7 chrF . . . 10
- 2.8 Comet . . . 11
- 2.9 SSIM . . . 12

3 Dataset **13**
- 3.1 Sběr dat . . . 13
- 3.2 Formát datasetu . . . 15
- 3.3 Extrakce informací z SVG souborů . . . 17
- 3.4 Statistika jazyků v datasetu . . . 18
- 3.5 Dělení na vývojovou a testovací sadu . . . 18

4 Evaluační metody **21**
- 4.1 Evaluace textové detekce . . . 22
- 4.2 Evaluace překladu textů . . . 24
- 4.3 Evaluace výstupního obrázku . . . 25
- 4.4 Evaluace detekce textu výstupního obrázku . . . 26
- 4.5 Shrnutí . . . 27

5 Nástroj pro překlad textů v obrázku **29**
- 5.1 Vstup nástroje . . . 29
- 5.2 Textová detekce . . . 29
- 5.3 Překlad textů . . . 31
- 5.4 Vykreslení textu . . . 31
- 5.5 Výstup nástroje . . . 32
- 5.6 Vyhodnocení nástroje . . . 33

6 Uživatelská dokumentace **34**
- 6.1 Struktura balíčku . . . 34
- 6.2 Požadavky . . . 34
- 6.3 Instalace . . . 35
- 6.4 Spuštění nástroje na překlad obrázků . . . 35
- 6.5 Spuštění evaluačního nástroje . . . 36

6.6 Práce s daty 37
6.6.1 Struktura datasetu 37

7 Vývojová dokumentace **39**
7.1 Skript na vytvoření datasetu 39
7.2 Překladové a vyhodnocovací nástroje 40

Závěr **42**

Seznam použité literatury **43**

Seznam obrázků **44**

Seznam tabulek **45**

A Přílohy **46**
A.1 Elektronicky přiložené dokumenty 46

1. Úvod

V dnešní době je strojový překlad textu na jiné úrovni, než tomu bylo před několika lety, a existuje mnoho nástrojů pro jeho realizaci. Mezi takové nástroje řadíme např. Google Translate[1], nebo DeepL[2]. Stejně tak existují různé metriky a techniky pro vyhodnocování kvality těchto překladů, můžeme zmínit například BLEU, viz Sekce 2.6.

Překlad textu v obrázcích ovšem představuje mnohem komplexnější úkol, který vyžaduje kombinaci strojového překladu s technikami optického rozpoznávání znaků (OCR) a dalšími metodami. Evaluace tohoto typu překladu rovněž není triviální a vyžaduje složitější přístup.

První možností překladu textu v obrázcích je použití tzv. pipeline přístupu. V pipeline (white-box) systémech je možné vyhodnocovat každý krok procesu, což umožňuje identifikovat a opravovat chyby v jednotlivých fázích.

Další možností je tzv. end-to-end přístup, kde máme obrázek na vstupu a na výstupu, přičemž celý proces překladu probíhá uvnitř tzv. black-boxu, kde nemáme přístup k jednotlivým krokům. Tento způsob provedení může být efektivní, ale je méně transparentní, protože nevíme, co se děje uvnitř systému. Příklad takového nástroje je Google Lens,[3] což je aplikace umožňující uživatelům přeložit text přímo z obrázků v reálném čase.

Naše práce se však soustředí pouze na přístup pipeline, to znamená, že budeme provádět evaluaci jednotlivých kroků odděleně, a postupně zhodnotíme jejich přínos k celkové kvalitě překladu.

K tomu všemu potřebujeme tři základní komponenty: data, evaluační nástroj a nástroj pro překlad textu v obrázcích.

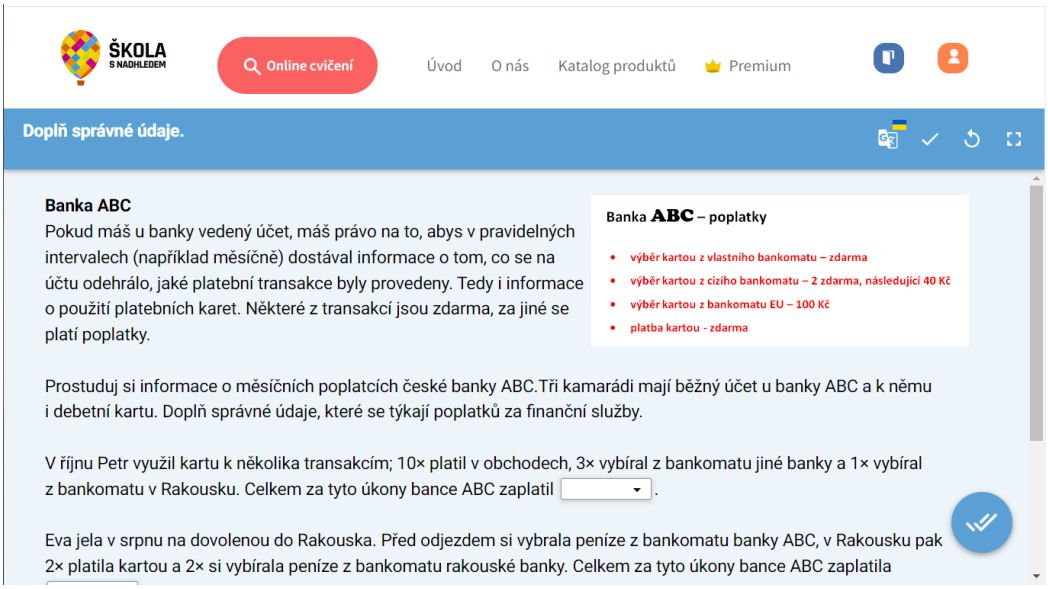
Naším cílem je tedy vytvoření základní pipeline, která na obrázku provádí postupně detekci textu, jeho překlad a vykreslení zpět do obrázku. Stěžejní částí naší práce je vytvoření evaluačního skriptu, jehož prostřednictvím vyhodnotíme správnost jednotlivých kroků pipeline, a dále sesbírání vhodných obrázků pro dataset potřebný k této úloze.

1.1 Motivace

Naše práce je spojena s projektem EdUKate,[4] což je projekt Ústavu formální a aplikované lingvistiky Univerzity Karlovy, jehož cílem je zmírnit jazykové bariéry mezi dětmi nemluvícími česky v České republice a vzděláváním v českém školním systému. Propojením oblastí digitálního vzdělávání, lingvistiky, studia překladu a strojového překladu se projekt zaměřuje na vývoj a šíření vícejazyčných digitálních vzdělávacích materiálů pro žáky základních a středních škol. Projekt vyvíjí vícejazyčný interaktivní obsah pro významný český výukový portál *Škola s Nadhledem*.

Škola s Nadhledem[5] je inovativní iniciativa Nakladatelství Fraus, která vznikla

[1] `https://translate.google.com/`
[2] `https://www.deepl.com/cs/translator`
[3] `https://lens.google/`
[4] `https://ufal.mff.cuni.cz/grants/edukate`
[5] `https://www.skolasnadhledem.cz/`



Obrázek 1.1: Příklad úlohy z webových stránek Školy s Nadhledem

s cílem podpořit rozvoj dětských znalostí. Tato platforma nabízí dětem možnost bezplatného procvičování vybraných témat. Škola s Nadhledem je součástí konceptu hybridních vzdělávacích materiálů, který spojuje tištěné učebnice a pracovní sešity s interaktivním online procvičováním a okamžitou zpětnou vazbou (příklad na Obrázku 1.1). Uživatelé na stránkách naleznou jak PDF soubory s informacemi k různým tématům, tak i spoustu zajímavých typů cvičení, jako jsou křížovky, přiřazování, kvízy, výběry z odpovědí ANO/NE atd.

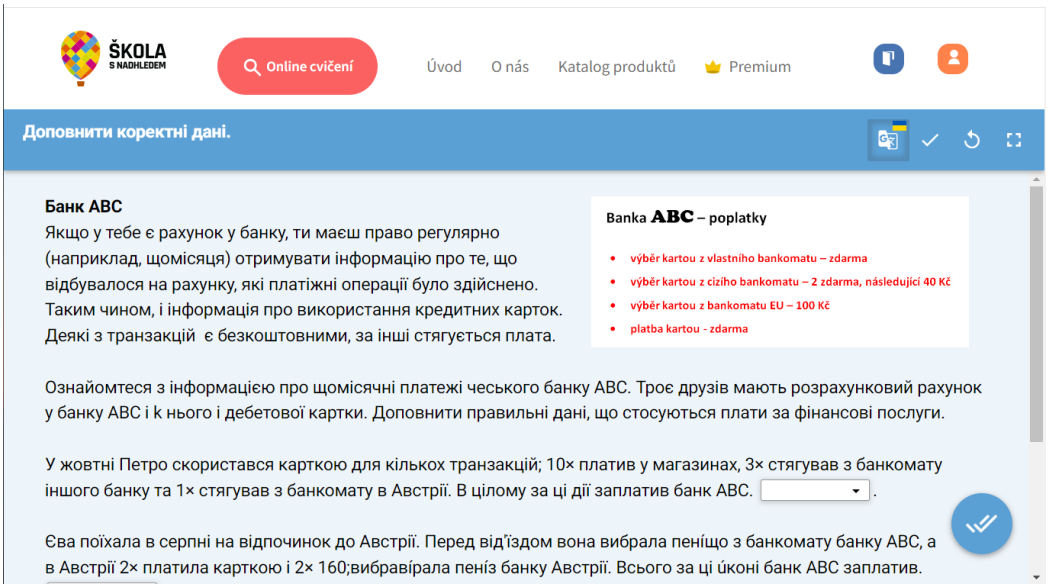
Tento portál a úlohy pro školáky jsou v současné době dostupné pouze v češtině (s výjimkou jazykových cvičení). Zejména po začátku ruské invaze na Ukrajinu a následné uprchlické vlně se ukázala potřeba překladu portálu do ukrajinštiny. Ačkoli ukrajinské děti po určité době strávené v českém prostředí nemají problém s porozuměním úloh v češtině, jejich rodiče, kteří jim chtějí případně s některými úlohami pomoci, mohou čelit nepřekonatelné jazykové bariéře. Navíc pro cizince z jazykově méně příbuzných zemí než Ukrajina je výukový portál dostupný pouze v češtině téměř nepoužitelný.

Proto se projekt EdUKate zaměřuje na rozšíření obsahu portálu do ukrajinštiny a postupně i do dalších jazyků, jako jsou například němčina a angličtina. Tímto způsobem bude portál přístupnější širšímu spektru uživatelů a usnadní vzdělávání dětem i jejich rodičům bez ohledu na jejich mateřský jazyk.

Provést strojový překlad textové části stránek je relativně snadné. Některé úlohy ovšem obsahují i obrázky, které mohou zahrnovat text podstatný pro vyřešení dané úlohy.

Ukážeme si problém na příkladu. Nakladatelství Fraus má k dispozici interně pilotní překlad do ukrajinštiny, přeložíme tedy úlohu z Obrázku 1.1 do tohoto jazyka, viz Obrázek 1.2. Na tomto příkladu můžeme vidět, že text v bílém poli přeložen nebyl, protože se jedná o obrázek. Tento text je však klíčový pro správné vyřešení úlohy. Bez jeho překladu nelze úlohu správně dokončit.

V portálu Škola s Nadhledem jsou obrázky obsahující text většinou rastrové konverze původně vektorových grafik. Kvalita vykreslení textu je tak poměrně vysoká. Tato práce se zaměřuje na zpracování takových obrázků a automatický



Obrázek 1.2: Úloha ze Školy s Nadhledem přeložená do ukrajinštiny

překlad textu v nich. Naopak se nezaměřuje na původně rastrové obrázky obsahující texty, jako jsou fotografie prostředí s nápisy nebo skeny dokumentů.

1.2 Členění práce

1.2.1 Dataset

Prvním cílem této práce je implementovat aplikaci pro automatický sběr dat pro úlohu překladu textů v obrázcích s důrazem na vektorové grafiky a diagramy ve formátu SVG. SVG je populární formát pro vektorovou grafiku na webu a v různých aplikacích umožňující změnu velikosti obrázků bez ztráty kvality. Zdrojem dat jsou volně dostupné obrázky z Wikipedie. Naším úkolem bude z nich vybrat dvojice obrázků obsahující grafiky i texty, lišící se pouze v textové části.

1.2.2 Evaluační metody

Automatické vyhodnocení systému provádějícího tuto úlohu pouze na základě výsledného rastrového obrázku může být poměrně obtížné. Vyhodnotit úspěšnost provedení jednotlivých kroků je s vhodně zvolenou testovací datovou množinou nicméně o dost snazší úkol. Na vyhodnocení budou použity standardní metriky pro hodnocení textové detekce, jako je míra chybovosti znaků, a metriky pro hodnocení úspěšnosti překladů, jako je BLEU (Bilingual Evaluation Understudy) a chrF (character n-gram F-score).

1.2.3 Překladový nástroj

Úlohu strojového překladu textů v obrázcích s rastrovou grafikou můžeme rozdělit do několika kroků. Prvním krokem je lokalizace textových polí v obrázku, následovaná převedením těchto polí z rastrové grafiky do textové reprezentace.

Poté následuje překlad textů, a nakonec vykreslení překladů zpět na odpovídající místa v obrázku.

V této práci se zaměřujeme na vývoj evaluačního schématu pro úlohu překladu textu v obrázcích. Cílem není navrhnout co nejlepší nástroj pro tuto úlohu. Účelem překladového nástroje implementovaného v této práci je spíše demonstrovat úlohu a verifikovat navržené evaluační schéma.

2. Použité nástroje a metriky

V této práci budeme pracovat s SVG soubory (2.1), které využijeme k získání referenčních dat potřebných k úloze překladu textu v obrázcích.

Pro tuto úlohu je potřeba daný text nejdříve z obrázku získat a lokalizovat. To řešíme pomocí nástroje Tesseract (2.2), který nám nejen poskytne extrahovaný text, ale také jeho souřadnice v obrázku. Abychom mohli vyhodnotit přesnost této textové detekce, použijeme míru chybovosti znaků (2.4). Tímto způsobem porovnáme detekovaný text s referenčním textem, který získáme ze souboru SVG pomocí nástroje Inkscape (2.3).

Dalším nezbytným krokem je samotný překlad textu. K tomu využíváme překladový model M2M100 (2.5). Výsledky překladu pak hodnotíme pomocí metrik BLEU (2.6), chrF (2.7) a Comet (2.8), abychom získali představu o kvalitě přeloženého textu.

Posledním krokem je vykreslení přeloženého textu zpět do obrázku, což nám poskytne finální výstupní obrázek. Tento obrázek následně porovnáme s referenčním výstupním obrázkem pomocí SSIM indexu (2.9), abychom vyhodnotili kvalitu vykreslení textu.

2.1 SVG

SVG, anglicky *Scalable Vector Graphics*, je XML formát pro vytváření dvourozměrné vektorové grafiky (Ferraiolo a kol., 2000). SVG se skládá z různých elementů, které společně tvoří komplexní grafiku. Nyní uvedeme několik příkladů, které budeme v naší práci potřebovat.

Element *<text>* se používá k vykreslení textu v SVG, přičemž Jeho atributy x a y umožňují umístění textu.

Atributy *<width>* a *<height>* definují šířku a výšku SVG a jsou použity v kořenovém *<svg>* elementu. *Transform* atribut v SVG souborech určuje transformaci, která se aplikuje na daný element. Tato transformace může zahrnovat posunutí (*translate*), změnu velikosti (*scale*), otočení (*rotate*) nebo zkosení (*skew*). Transformace tedy způsobují, že souřadnice vykreslených textů jsou jiné než ty, které jsou v samotném SVG souboru zapsány.

Tag *<switch>* se používá k poskytování různých verzí textu. Každý prvek *<text>* má atribut *systemLanguage*, který definuje jazyk, ve kterém bude text zobrazen. Každý text má také přiřazen jedinečný identifikátor *id*. Identifikátor představuje několik písmen či čísel, v případě jednoho textu v tagu *switch* se tyto identifikátory liší pouze posledními dvěma/třemi písmeny, která představují kód jazyka, ve kterém je daný textový obsah napsán (např. *en* pro angličtinu, *cs* pro češtinu).

2.2 Tesseract

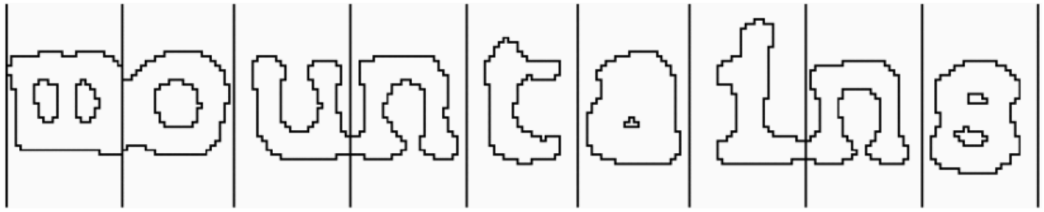
Textová detekce, označována jako OCR (Optical Character Recognition = optické rozpoznávání znaků) je metoda, která umožňuje rozpoznávat text z obrázků nebo skenů dokumentů a převádět ho do editovatelného formátu.

Pro textovou detekci použijeme nástroj Tesseract,[1] což je modul pro optické rozpoznávání znaků pro Windows, Linux a Mac OS X. Jedná se o volně dostupný software podporující různé formáty obrázků a umožňující extrakci textu přímo bez ukládání do souboru.

Tesseract, jak uvádí Smith (2007), přijímá binární obrázky[2] jako vstup a má schopnost identifikovat a zpracovat text v konkrétních polygonálních oblastech v rámci těchto obrázků. Tato flexibilita umožňuje cílenější zpracování OCR, což je užitečné v situacích, kde pouze určité části obrázku obsahují relevantní text.

Zpracování probíhá krok za krokem v rámci tzv. pipeline. Prvním krokem je analýza komponent, ve které jsou identifikovány a uloženy obrysy komponent. Tato analýza umožňuje snadnou detekci jak černého textu na bílém pozadí, tak i bílého textu na černém pozadí. Následně jsou tyto obrysy seskupeny a organizovány do textových řádků. Řádky textu jsou dále rozděleny na jednotlivá slova na základě mezery mezi znaky.

Tesseract testuje řádky textu, aby zjistil, zda je text pevně řazený, neboli zda se jedná o tzv. *fixed-pitch* text. Fixed-pitched text se vyznačuje pevně daným rozestupem mezi jednotlivými znaky. Může u něj docházet ke zkrácení nebo ztrátě některých znaků, například kvůli špatné kvalitě obrázku nebo jiným faktorům při zpracování textu. Tam, kde Tesseract najde slovo s pevnou šířkou písma, rozseká slova na znaky podle dané šířky a deaktivuje asociátor těchto slov pro krok rozpoznávání slov. Obrázek 2.1 ukazuje typický příklad pevně řazeného slova.



Obrázek 2.1: Text s pevnou šířkou písma (Smith, 2007)

Rozpoznávání probíhá dvouprůchodovým procesem. V prvním průchodu se Tesseract postupně pokusí rozpoznat každé slovo. Úspěšně rozpoznaná slova jsou předána adaptivnímu klasifikátoru (*adaptive classifier*) jako trénovací data, což umožňuje klasifikátoru lépe rozpoznat další texty. Druhý průchod je prováděn za účelem rozpoznání slov, která nebyla úspěšně identifikována v prvním průchodu.

Poslední fáze se zabývá neostrými mezerami a zkoumáním alternativních hypotéz týkajících se výšky písma, s cílem nalézt text s menšími písmeny.

V modulu rozpoznávání slov hraje lingvistická analýza menší roli ve srovnání se segmentací. Modul vybírá nejlepší slovní řetězec z různých kategorií, jako jsou nejčastěji používaná slova, slova ze slovníku, velká a malá písmena apod. Každá kategorie má u rozpoznávání svou váhu.

Tesseract má podporu pro více než 100 jazyků, nicméně může mít obtíže s rozpoznáváním některých písem, jako je například azbuka.

[1] `https://github.com/tesseract-ocr/tesseract`

[2] Binární obrázek (*binary image*) je obrázek, který se skládá jen ze dvou hodnot pixelů, obvykle černé a bílé (0 a 1).

2.3 Inkscape

Získat referenční text z SVG souborů pro evaluaci textové detekce je možné buď přímo zpracováním SVG, tzn. přečtením příslušných tagů *<text>*. Souřadnice zmíněných textů nalezneme pod atributy x a y. V tomto případě však musíme počítat se značnými problémy, jako jsou různé transformace souřadnic (viz 2.1). Proto jsme se vydali jinou cestou, a to zobrazením obrázku přímo v aplikaci Inkscape, a následné extrakce textu a jeho souřadnic.

Inkscape je bezplatný veřejně dostupný grafický editor vektorových grafik, který se hojně využívá k vytváření a úpravám souborů vektorové grafiky ve formátu SVG. Uživatelům poskytuje široký výběr nástrojů pro kreslení pro různě složité umělecké práce, včetně ilustrací či diagramů (Kirsanov, 2021).

Inkscape nabízí uživatelské rozhraní umožňující snadné použití pomocí grafického uživatelského rozhraní (GUI), ale lze ho také ovládat z příkazové řádky.

2.4 Míra chybovosti znaků

Při evaluaci detekovaného textu, pokud máme zlaté texty z SVG souborů, můžeme využít míry chybovosti znaků (neboli *Character Error Rate*, *CER*) k určení podobnosti mezi detekovaným a zlatým textem. Tam (2021) popisuje CER jako metriku, která měří, kolik znaků bylo nesprávně rozpoznáno nebo chybí v porovnání s referenčním textem. Vypočítává se jako počet chyb (vložení, vynechání a substituce znaků) dělený celkovým počtem znaků v referenčním textu. Méně chyb znamená nižší CER a vyšší přesnost.

Existuje také míra chybovosti slov (*Word Error Rate*, *WER*), tato metrika se používá při vyhodnocování detekce textu, kde je jednotka změny a normalizace slovo (a ne znak). V této práci ji však nebudeme vyhodnocovat, protože rozpoznané texty jsou ve většině případech velmi krátké – průměrná délka textu v datasetu je menší než 2 slova.

2.5 M2M100

M2M100 (Fan a kol., 2021) je vícejazyčný model zaměřený především na překladové úlohy. Jeho název *M2M* odkazuje na *many–to–many*, což naznačuje jeho schopnost překládat mezi různými kombinacemi jazyků. Jedná se o první mnohojazyčný model, který dokáže překládat mezi libovolnou dvojicí 100 jazyků bez potřeby anglických dat, což umožňuje lepší zachování významu vět.

Důraz na modely nezávislé na angličtině přináší zlepšení o více než 10 bodů na BLEU metrice (viz 2.6) při překladech mezi neanglickými směry.

Model využívá techniku nazývanou „mezijazykový překlad“ nebo „mostový překlad“, kde některé jazyky slouží jako prostředník při překladu mezi páry jiných jazyků. Tyto mostové jazyky jsou vybírány na základě lingvistické klasifikace nebo kulturních podobností. Místo tradičního způsobu, kdy se používá angličtina jako mezistupeň, využívá M2M100 přímo vícejazyčná trénovací data, která zahrnují překlady v rámci jazykových skupin a mezi nimi prostřednictvím tzv. „bridge“ jazyků.

M2M100 lze snadno použít. Stačí mu poskytnout vstupní text v jednom jazyce a on vygeneruje odpovídající překlad do cílového jazyka. Model potřebuje tři parametry - vstupní text, kód zdrojového jazyka a kód cílového jazyka pro překlad.

2.6 BLEU

Jak uvádí Papineni a kol. (2002), BLEU (Bilingual Evaluation Understudy) je algoritmus pro vyhodnocení kvality textu, který byl strojově přeložen z jednoho jazyka do druhého. Kvalita se posuzuje podle podobnosti mezi výstupem strojového překladu a textu přeloženého člověkem.

Skóre jsou počítaná pro jednotlivé přeložené části, většinou věty, porovnáváním s množinou kvalitních referenčních překladů. V našem případě však máme jen jednu referenci, a to texty získané z SVG.

BLEU používá modifikovanou přesnost slovních n-gramů jako hlavní kritérium hodnocení. To znamená, že se počítá přesnost n-gramů (sekvencí slov) přeloženého textu vůči referenčnímu textu, přičemž se zohledňuje penalizace za příliš krátké překlady (*brevity penalty*). Modifikovaná přesnost se vypočítává tak, že se sčítají n-gramy, které se vyskytují jak v překladu, tak v referenci, a tento počet se dělí celkovým počtem n-gramů v překladu.

Hodnota skóre BLEU se pohybuje od 0 do 1. Čím blíže je hodnota k 1, tím lepší je překlad. Prakticky není možné dosáhnout hodnoty 1 a obvykle je za dobré skóre považována hodnota vyšší než 0,3.

Při použití BLEU dostaneme tyto informace:

- *bleu (float)*: BLEU skóre
- *precisions (list of floats)*: geometrický průměr přesnosti n-gramů
- *brevity_penalty (float)*: penalta pro krátké hypotézy
- *length_ratio (float)*: poměr délek
- *translation_length (int)*: délka přeloženého textu
- *reference_length (int)*: délka referenčního textu

K použití metriky BLEU využijeme knihovny `sacrebleu`.[3] Tato knihovna je vyvinutá nejen pro výpočet BLEU skóre, ale i skóre chrF.

2.7 chrF

Metrika chrF (*CHaRacter-level F-score*) slouží k hodnocení strojového překladu a vypočítává podobnost mezi výstupem strojového překladu a referenčním překladem pomocí znakových n-gramů, nikoli n-gramů slov (Popović, 2015). Metriky založené na slovních n-gramech jsou obzvlášť problematické pro jazyky s vysokou morfologií.

[3] `https://pypi.org/project/sacrebleu/`

Obecný vzorec pro skóre chrF je:

$$\text{chrF}_\beta = (1 + \beta^2)\frac{\text{chrP} \cdot \text{chrR}}{\beta^2 \cdot \text{chrP} + \text{chrR}}$$

kde chrP a chrR označují přesnost (anglicky *precision*) a úplnost (anglicky *recall*) znakových n-gramů aritmeticky průměrované přes všechny n-gramy:

- chrP: procento n-gramů v hypotéze, mají odpovídající verzi v referenci
- chrR: procento znakových n-gramů v referenci, které jsou také obsaženy v hypotéze

a β je parametr, který přiřazuje βkrát větší důležitost úplnosti než přesnosti – pokud je $\beta = 1$, pak jsou hodnoty stejně důležité.

2.8 Comet

COMET (Crosslingual Optimized Metric for Evaluation of Translation; Rei a kol., 2020) je metrika pro automatickou evaluaci strojového překladu, která počítá podobnost mezi výstupem strojového překladu a referenčním překladem pomocí tokenového nebo větného embeddingu.[4] Míra podobnosti embeddingů je natrénována na manuálních hodnoceních strojového překladu. Tím se COMET liší od metrik BLEU a chrF, které jsou založené na výpočtu přímé podobnosti textových řetězců a nejsou trénovány na žádných datech.

Jednou z klíčových inovací COMET je zahrnutí informací ze zdrojového textu do procesu hodnocení. Tradiční metriky hodnocení strojového překladu se obvykle spoléhají pouze na referenční překlad, ale COMET tento přístup rozšiřuje, aby zahrnul informace jak ze zdrojového textu, tak z referenčního překladu. Tato nová metoda umožňuje COMETu lépe zachytit jemné rozdíly v kvalitě strojového překladu a zlepšit přesnost svých hodnocení.

Framework podporuje dvě různé architektury: Estimator model, který je trénován tak, aby přímo odhadl nebo predikoval skóre kvality překladu, a Translation Ranking model, který je trénován ke snížení vzdálenosti mezi hypotézami a odpovídajícími referencemi a zdroji. Oba modely obsahují multilinguální enkodér (cross-lingual encoder) a sdružovací vrstvu (pooling layer), což jim umožňuje efektivně zpracovávat vícejazyčná vstupní data.

Na výstupu má metrika tyto dva seznamy:

- *scores*: COMET skóre pro každou větu na vstupu, s rozsahem mezi 0 a 1
- *mean_score*: střední hodnota COMET skóre *scores* ve všech vstupních větách v rozmezí 0-1

Původní studie COMETu uvádí průměrné hodnoty COMET skóre v rozmezí od 0,4 do 0,6, přičemž hodnoty závisí na jazykových párech použitých k hodnocení překladových modelů. Taktéž dokazují, že COMET dobře koreluje s lidskými hodnoceními ve srovnání s jinými metrikami, jako je BLEU a CHRF.

[4]Embedding je vektorová reprezentace slov nebo vět, která zachycuje jejich významové vlastnosti a vztahy.

2.9 SSIM

Strukturální podobnostní index, *Structural Similarity Index, SSIM* (Nilsson a Akenine-Möller, 2020) je metrika používající se pro měření podobnosti dvou obrázků. Tento index bere v úvahu nejen rozdíly v pixelech, ale i strukturální informace obrazu. SSIM nabývá hodnot od -1 do 1, přičemž hodnota 1 označuje identické obrázky.

3. Dataset

3.1 Sběr dat

Úkolem projektu je sesbírat takové dvojice obrázků, které se liší pouze v textech. Zároveň musí platit, že texty na stejných pozicích v dvojici obrázků si jsou navzájem překlady.

Zdrojem dat pro projekt je internetová stránka Wikimedia Commons[1], která slouží jako organizované úložiště volně dostupných fotografií, videí, zvuků a dalšího multimediálního obsahu z Wikipedie. Commons neobsahuje přímo články o jednotlivých tématech, ale funkčně jsou článkům obdobné stránky s galeriemi obrázků, na nichž lze obrázky či jiné soubory různými způsoby vybírat, seskupovat a popisovat či komentovat. Základní informace o tématu nebo objektu lze uvést též na stránce kategorie, avšak primárním způsobem popisu témat kategorie je propojení meziprojektovými odkazy na články Wikipedie o příslušném tématu, případně položku Wikidat.

Každý mediální soubor dostupný na Wikimedia Commons má přiřazený svůj vlastní popis, jehož obsah by měl poskytnou stručné a úplné informace o příslušném médiu, obrázku či videu. Tyto informace zprostředkovává šablona *Information*, která poskytuje strojově čitelná data. Tato šablona se používá k formátování základních informací o souborech (popis, zdroj, autor, atd.) a je automaticky vložena při nahrávání.

Formát šablony Information můžeme vidět na Obrázku 3.1.

```
{{Information
 |description    =
 |date           =
 |source         =
 |author         =
 |permission     =
 |other versions =
}}
```

Obrázek 3.1: Šablona Information

Nejdůležitějšími parametry šablony jsou tedy *description*, což je stručný popis obrázku nebo jiného mediálního souboru, *date* reprezentující datum vzniku originálního souboru, *source* informující o původu souboru včetně informací o souborech, na kterých je daný mediální soubor založen, a *author*, kde se dozvíme jméno autora/autorů.

Pro naši úlohu je však klíčovou částí parametr *other versions*. Tato sekce zahrnuje odkazy na soubory s velmi podobným obsahem nebo na různé odvozené soubory (anglicky *derivative files*) od původního mediálního souboru. Other versions tedy mohou odkazovat na soubory s jiným rozlišením, formátem nebo s rozlišnými úpravami, což napomáhá udržovat pořádek a usnadňovat navigaci.

[1] `https://commons.wikimedia.org`

V určitých případech mezi tyto alternativní varianty souboru Wikimedia Commons řadí i soubory, které jsou překladem původního média, tzn. texty původního obrázku jsou přeloženy do jiného jazyka. Obrázek obsahuje přeložené verze v sekci *other versions* pouze tehdy, je-li daný obrázek na Wikimedia Commons k dispozici v různých jazykových překladech. Úkolem našeho projektu bylo nalézt dostatek obrázků splňujících tuto podmínku.

Pro tento krok jsme se inspirovali sběrem obrázků pomocí webového API poskytovaného Wikimedia Commons, konkrétně *MediaWiki Action API*.[2] Toto rozhraní nabízí různorodé možnosti pro získávání obrázků, včetně vyhledávání na základě dotazu. Například pomocí metod jako *action=query* a *prop=images* se dají získat obrázky z konkrétních kategorií či článků.

Příklad dotazu z MediaWiki Action API a odpovědi na daný dotaz můžeme vidět na Obrázku 3.2.

```
api.php ? action=query  & prop=globalusage & format=json &
titles=File%3A100%20Years%20War%20France%201435.svg &
formatversion=2

{"query": {
   "pages": [
    {
      "ns": 6,
      "title": "File:100 Years War France 1435.svg",
      "globalusage": [
       {
         "title": "Doppelmonarchie_von_England_und_
         Frankreich",
         "wiki": "de.wikipedia.org",
         "url": "https://de.wikipedia.org/wiki/
         Doppelmonarchie_von_England_und_Frankreich"
       },
       {
         "title": "Dual_monarchy_of_England_and_France",
         "wiki": "en.wikipedia.org",
         "url": "https://en.wikipedia.org/wiki/Dual_
         monarchy_of_England_and_France"
        }
      ]
    }
  ]
 }
}
```

Obrázek 3.2: Příklad dotazu s odpovědí z MediaWiki Action API

Při procházení stránek Wikimedia Commons jsme narazili na kategorii s názvem *Translation_possible_-_SVG*, která se ukázala být užitečná pro náš projekt, a proto jsme s ní začali pracovat. Tato kategorie označuje SVG (Scalable

[2] `https://www.mediawiki.org/wiki/API:Main_page`

Vector Graphics) soubory, které jsou vhodné pro překlad a je snadné nahrát jejich novou jazykovou verzi. Překlady mohou být rovněž zahrnuty v tom stejném souboru použitím elementu *<switch>*.

Kategorie *Translation_possible_-_SVG* obsahuje rozlišné druhy SVG přes mapy až po diagramy, dohromady pod tuto kategorii spadá 23 726 souborů. Z toho ne všechny soubory obsahují text nebo nejsou vhodné pro naši úlohu. Proto jsme rozšířili množinu hledaných kategorií i o další kategorie, do kterých patří soubory z *Translation_possible_-_SVG*. Pro získání informací o kategoriích, do nichž obrázek na stránce Wikimedia Commons patří, a o jeho dalších verzích ze sekce *other versions*, jsme využili metodu web scraping, tj. automatizovanou extrakci dat z webových stránek. Automaticky procházíme webové stránky, extrahujeme požadovaná data a ukládáme je ve strukturovaném formátu. Díky této metodě jsme nyní schopni získat následující údaje:

- URL SVG obrázku
- URL PNG obrázku
- URL stránky daného obrázku
- popisek ze sekce *other versions*, v našem případě se zaměřujeme pouze na jazykové popisky, tedy na jazykové kódy, názvy jazyků atp.

Do datasetu jsme vybrali skupiny obrázků, které obsahují více než dva prvky – tedy obrázky s více než dvěma jazykovými popisky v sekci *other versions*. Tyto jazykové popisky zahrnují jak jazykové kódy (dvou nebo třípísmenné jazykové zkratky), tak i celé názvy jazyků, a to jak v angličtině, tak i v příslušném jazyce.

V této fázi máme k dispozici URL adresy jak pro SVG, tak PNG soubory, a rovněž odkazy na stránky, na kterých se původní obrázky nacházejí spolu s jejich titulky (názvy jazyků). Nejprve tedy stáhneme všechny PNG a SVG soubory patřící do jedné skupiny vzájemných překladů a následně pro každou dvojici jazyků vytvoříme konkrétní příklad zlatých dat v podobě JSON struktury.

3.2 Formát datasetu

Data do datasetu ukládáme v JSON (JavaScript Object Notation) formátu, který nabízí systematické uspořádání všech souvisejících informací. Struktura datasetu je znázorněna na Obrázku 3.3.

Pro lepší porozumění struktury a obsahu dat nyní podrobně popíšeme jednotlivé části datasetu:

- *source_language*: kód jazyka původního textu, který má být přeložen
- *source_PNG*: sekce s informacemi o původním obrázku s textem, který má být přeložen
  - *size*: velikost PNG (šířka a výška)
  - *path_to_image*: cesta k původnímu obrázku
  - *wikimedia_url*: adresa webové stránky původního obrázku na Wikimedia Commons

```
{
    "source_language": language code,
    "source_PNG": {
        "size": {
            "width": px,
            "height": px
        },
        "path_to_image": path,
        "wikimedia_url": url
    },
    "text_bounding_box": [
        {
            "x": float,
            "y": float,
            "w": float,
            "h": float
        }
    ],
    "texts": [
        string
    ],
    "target_language": language code,
    "translated_texts": [
        string
    ],
    "target_PNG": {
        "size": {
            "width": px,
            "height": px
        },
        "path_to_image": path,
        "wikimedia_url": url
    }
}
```

Obrázek 3.3: JSON struktura datasetu

- *text_bounding_box*: sekce obsahující seznam ohraničujících obdélníků textových bloků
  - x: x-ová souřadnice levého horního rohu obdélníku
  - y : y-ová souřadnice levého horního rohu obdélníku
  - w : šířka obdélníku
  - h : výška obdélníku
- *texts*: seznam textů z původního obrázku
- *target_language*: kód jazyka, do kterého má být text přeložen
- *translated_texts*: seznam přeložených textů
- *target_PNG*:
  - *size*: velikost PNG (šířka a výška)

- *path_to_image*: cesta k obrázku s přeloženými texty
- *wikimedia_url*: webová stránka obrázku s přeloženými texty na Wikimedia Commons

Seznamy *text_bounding_box*, *texts* a *translated_texts* jsou stejně dlouhé a položky na stejné pozici v seznamech si navzájem odpovídají.

3.3 Extrakce informací z SVG souborů

Pro vytvoření korektních zlatých dat ve formátu specifikovaném v Sekci 3.2 je nezbytné extrahovat určité informace z SVG souborů, především samotné texty spolu s jejich souřadnicemi. Při prvním pohledu by se mohlo zdát, že se jedná o jednoduchou úlohu – SVG formát popisuje vektorovou grafiku pomocí XML a texty jsou uloženy v tagu *<text>*, přičemž jejich souřadnice jsou obsaženy pod klíči x a y.

Klíče x a y se ukázaly být ne úplně spolehlivým zdrojem informací o poloze textu v rámci obrázku. Jedním z důvodů je různé zarovnání textů. Jejich souřadnice tak mohou odpovídat buď levému dolnímu rohu textového bloku, nebo středu spodního bodu textového bloku. Dalším důvodem může být přítomnost atributu *transform*, jenž může kompletně transformovat souřadnicový prostor. Obecně je obtížné zjistit finální pozici daného prvku v SVG souboru bez toho, aby se nejprve celý soubor interpretoval, tedy vykreslil.

Proto jsme se rozhodli použít jiný přístup a využít aplikaci Inkscape (viz Sekce 2.3), která nám umožnila nejprve vykreslit obrázek a až poté získat správné souřadnice textů. Jediným problémem bylo normalizovat tyto souřadnice s ohledem na PNG soubor. Stačilo provést jednoduchou operaci:

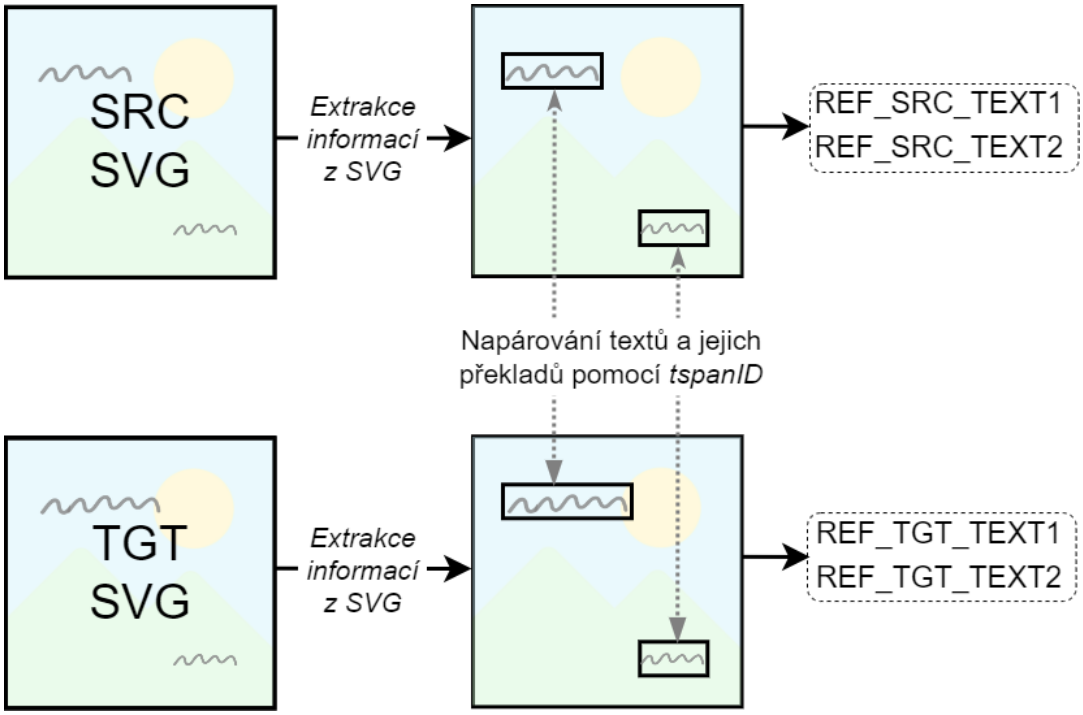
$$coordinate_{PNG} = coordinate_{SVG} \cdot \frac{height_{PNG}}{height_{SVG}} \text{ kde } coordinate \in \{x, y, w, h\}$$

Výšku SVG získáváme z atributu *height* v tagu *<svg>*. Avšak tento údaj může být vyjádřen v různých jednotkách (jako jsou milimetry, pixely nebo body), a proto je nutné tyto hodnoty vhodně převést na námi požadované jednotky.

V datasetu uvádíme jak původní texty, tak jejich překlady. Abychom nemuseli zahrnout souřadnice obou těchto skupin textů, zachováme jejich pořadí. To znamená, že první přeložený text odpovídá prvnímu původnímu textu, druhý přeložený text odpovídá druhému původnímu textu, a tak dále. Tato shoda textů je snadná díky identifikátorům *tspanID*, které získáme využitím Inkscape. Tyto identifikátory se shodují mezi jazyky – v některých případech jsou naprosto totožné, jindy mají na konci přidanou příslušnou jazykovou zkratku.

Proces napárování textů z různých jazyků můžeme vidět na Obrázku 3.4. Texty REF_SRC_TEXT{1,2,...} označující referenční texty ve zdrojovém jazyce a texty REF_TGT_TEXT{1,2,...}, které představují referenční texty v cílovém jazyce (tedy překlady ze zdrojového jazyka) označíme za zlatá data a budeme je využívat při evaluaci, více v Kapitole 4.

Někdy jsme narazili na situaci, kdy nebyly k dispozici kompletní textové páry – například může být různý počet textů v odlišně strukturovaných jazycích, nebo texty obsahovaly jiná formátování. V těchto případech jsme nemohli zařadit tyto SVG soubory do našeho datasetu, protože nebylo možné vytvořit páry textů a jejich překladů.



Obrázek 3.4: Znázornění napárování textů a překladu textů z SVG souborů

3.4 Statistika jazyků v datasetu

Celkově jsme byli schopni sesbírat data z 26 jazyků. Nejvíce jazykových párů se týká angličtiny, a to konkrétně 1530 párů, jak je ukazuje Tabulka 3.1.

Data jsou rozdělena do skupin obrázků lišících se jen v jazyce textů. Tyto skupiny mohou být různě velké, nejvýše se nám povedlo nalézt 12 obrázků, které jsou si navzájem překlady.

Celková statistika je uvedena na Obrázku 3.5. V datasetu je nejvíce skupin se dvěma obrázky lišícími se v jazycích popisků, konkrétně 330.

Tabulka 3.2 ukazuje jazykové páry s nejvyšší četností v datasetu. Nejčastějším párem je angličtina-francouzština, s celkovým počtem 256 párů. (Je třeba poznamenat, že se jedná o kombinace angličtina-francouzština i francouzština-angličtina.) Český jazyk se vyskytuje nejvíce v páru angličtina-čeština, s konkrétně 64 výskyty.

3.5 Dělení na vývojovou a testovací sadu

Při vývoji a testování evaluačního nástroje je klíčové rozdělit data na vývojovou a testovací sadu.

Vývojová (*development*) sada slouží k ladění nástroje, používáme ji během vývoje evaluačního programu pro průběžné testování a optimalizaci. Výsledky evaluačních metod v Kapitole 4 jsou výsledky provedení evaluací právě nad vývojovou částí datasetu.

Testovací (*test*) sada je určena k finálnímu vyhodnocení, tedy k otestování evaluačního nástroje na datech, která nástroj dosud neviděl.

Při dělení datasetu na tyto dvě části jsme postupovali tak, že jsme rozdě-

lili data podle velikosti skupin obrázků lišících se jen v jazycích textů a poté jsme z každé množiny skupin stejných velikostí vybrali náhodně polovinu skupin do vývojové sady, zbytek jsme přidali do testovací sady.

Jazyk	**Kód**	**Počet skupin**[1]	**Počet párování**[2]
Angličtina	*en*	435	1530
Francouzština	*fr*	204	796
Němčina	*de*	160	668
Italština	*it*	120	604
Maďarština	*hu*	79	368
Polština	*pl*	67	398
Ruština	*ru*	67	296
Španělština	*es*	62	266
Čeština	*cs*	59	314
Turečtina	*tr*	50	260
Švédština	*sv*	49	284
Holandština	*nl*	37	200
Finština	*fi*	31	184
Norština	*no*	28	146
Rumunština	*ro*	28	146
Latina	*la*	9	70
Kannadština	*kn*	8	56
Indonéština	*id*	7	26
Čínština	*zh*	5	20
Korejština	*ko*	4	22
Gruzínština	*ka*	4	22
Japonština	*ja*	3	20
Vietnamština	*vi*	3	22
Albánština	*alb*	1	2
Běloruština	*be*	1	2
Estonština	*et*	1	2

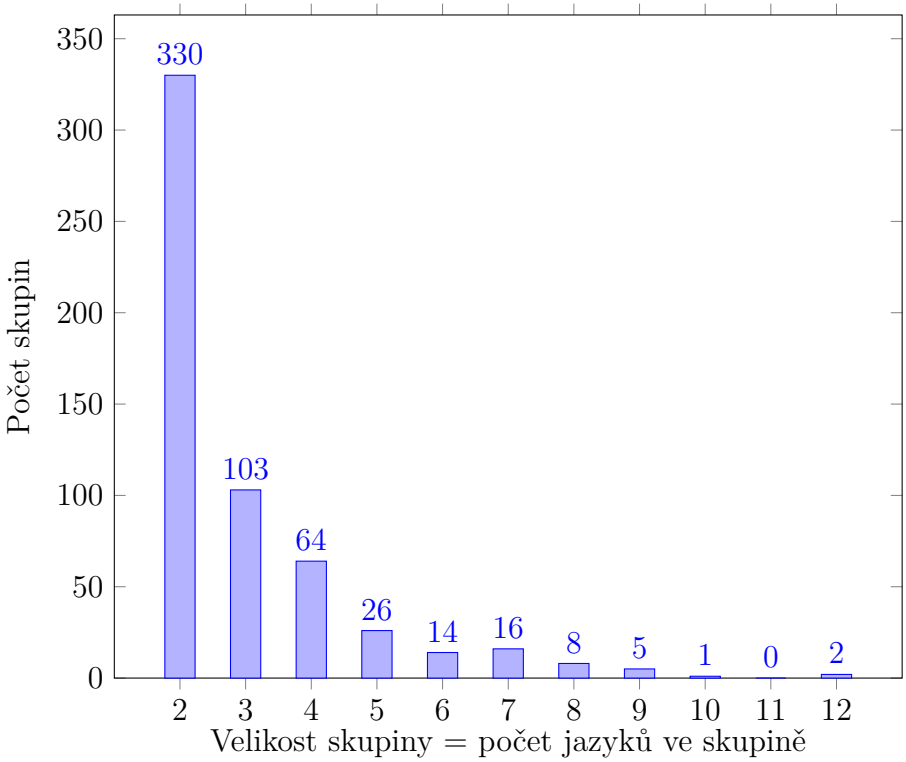
Pozn: [1] Počet skupin obrázků lišících se jen v jazyce textů, obsahující daný jazyk.
[2] Počet jazykových párů v datasetu, kde je daný jazyk buď zdrojový, nebo cílový.

Tabulka 3.1: Statistika jazyků v datasetu

Jazykový pár	**Počet**
Angličtina – Francouzština	256[1]
Angličtina – Němčina	200
Angličtina – Italština	138
Angličtina – Maďarština	100
Angličtina – Ruština	94
Angličtina – Polština	92
Francouzština – Italština	86
...	
Angličtina – Čeština	64
...	
Celkový počet párů	3804

Pozn: [1] Počet párů en-fr i fr-en, platí i pro ostatní jazykové páry

Tabulka 3.2: Statistika jazykových párů v datasetu (nejpočetnější)



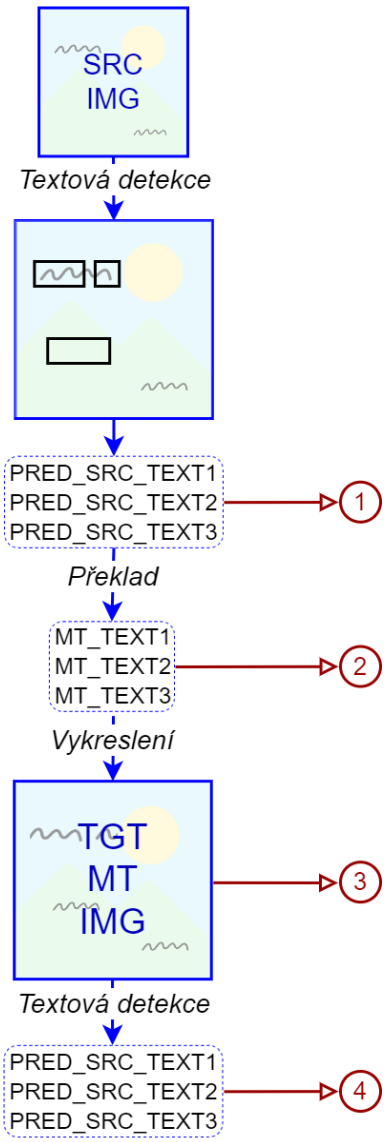
Obrázek 3.5: Statistika velikostí skupin obrázků lišících se jen v jazyce textů

4. Evaluační metody

V této kapitole popíšeme program, který jsme navrhli na evaluaci metod překladu textů v obrázcích. Je určený zejména pro vyhodnocení metod, jenž jsou implementovány jako posloupnost na sebe navazujících kroků textové detekce, překladu a vykreslení tak, jak je znázorněno na Obrázku 4.1. Textová detekce přitom sestává z lokalizace textových bloků a rozpoznání textu.

Evaluační program umožňuje vyhodnotit i mezivýsledky těchto kroků. Červené šipky v diagramu vycházejí z příslušných úseků programu, které potřebujeme pro daný krok evaluace, přičemž červená čísla odkazují na sekce této kapitoly, kde se daným evaluačním metodám věnujeme.

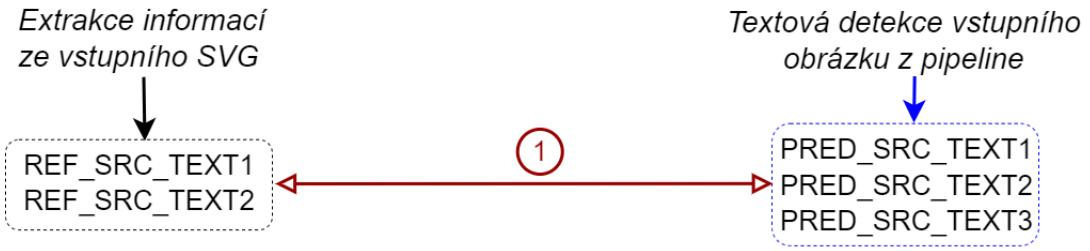
Ukázkovou implementaci překladového nástroje, který odpovídá tomuto schématu, pak popisujeme v Kapitole 5. Níže uvedené evaluační metody lze použít pro každý překladový nástroj, který dodržuje tuto architekturu.



Obrázek 4.1: Znázornění překladového nástroje

4.1 Evaluace textové detekce

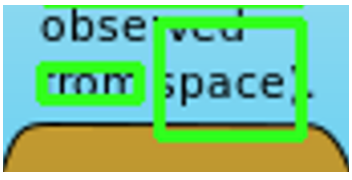
Jak popisuje Diagram 4.2, při evaluaci textové detekce předpokládáme, že máme k dispozici již detekované texty PRED_SRC_TEXT{1,2,...} společně s jejich souřadnicemi, tedy obdélníky ohraničujícími dané texty (ukázka na Obrázku 4.4), konkrétně x-ovou a y-ovou souřadnici levého horního rohu textového boxu s šířkou a výškou tohoto boxu. Taktéž disponujeme zlatými daty, což jsou referenční texty REF_SRC_TEXT{1,2,...} a jejich souřadnicemi vůči danému PNG obrázku (viz Obrázek 4.5).



Obrázek 4.2: Znázornění evaluace textové detekce

Nicméně není jasné, který detekovaný text (označujeme také jako predikovaný text) odpovídá kterému referenčnímu textu. Naším prvním úkolem je tedy provést párování detekovaných textů s referenčními. K tomu využíváme překrývání textových bloků. Konkrétně hledáme ke každému detekovanému textu ten referenční text, jehož textový blok má s detekovaným největší překryv (*max overlap area*).

Obrázek 4.3: Ukázka části obrázku před textovou detekcí

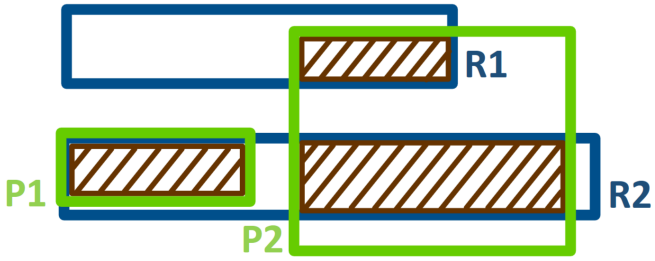


Obrázek 4.4: Ohraničení textu podle souřadnic z nástroje textové detekce

Obrázek 4.5: Ohraničení textu podle referenčních souřadnic

Obrázek 4.6 ukazuje hledání největšího překryvu bloků z Obrázků 4.4 a 4.5. Pro snadnější popis označme postupně referenční obdélníky R1, R2, a predikované obdélníky P1 a P2.

Jak můžeme vidět, predikovaný obdélník P2 se překrývá se dvěma referenčními obdélníky R1 a R2 reprezentující zlatá data. Obsah překrývající se plochy s R2 je však větší, proto se text v obdélníku P2 přiřadí k referenčnímu textu v obdélníku



Obrázek 4.6: Hledání největšího překryvu pro napárování textů z obrázku 4.3

R2. Detekovaný obdélník P1 se překrývá pouze s jedním modrým obdélníkem reprezentujícím zlatý text, s obdélníkem R2, proto se P1 přiřadí k R2.

Po provedení tohoto kroku máme tedy několik množin textů:

- páry referenční text – detekovaný text
- detekovaný nenapárovaný text
- referenční text, který nástroj na detekci textu nerozpoznal.

Zvažovali jsme také úplné rozdělení referenčního textu na jednotlivá slova, čímž by se zvýšila kvalita párování – rozpoznaná slova by se skutečně mohla napárovat s odpovídajícími zlatými slovy. Avšak tento přístup by vyvolal další problémy, jako je přesné rozdělení celého bloku referenčního textu na jednotlivá slova a také rozdělení predikovaných textových bloků s více slovy (pro porovnání a napárování potřebujeme jak x-ovou, tak y-ovou souřadnici). Tento fakt by pravděpodobně ještě více zkomplikoval úlohu a mohl by vést k nesprávnému rozdělení textů. Také by se mohly vyskytnout obtíže v dalším kroku – při překladu textu, a to kvůli odlišné délce slov nebo slovosledu v různých jazycích.

Abychom předešli rozdělení textu na jednotlivá slova a nesprávnému párování pouze části textu, přidali jsme možnost spojení textů (anglicky *text merging*). Pokud se dva či více textů napárovaly na jeden referenční text (metodou *max overlap area*), pak tyto texty s největší pravděpodobností patří k sobě a jen při detekci byly rozděleny. Takové texty jsou spojeny a napárovány na referenční text. Texty spojujeme mezerami v pořadí podle jejich x-ové a y-ové souřadnice.

Nyní se vraťme zpět k Obrázku 4.6. Zde jsme již zjistili, že oba predikované obdélníky P1 i P2 se napárovaly na referenční obdélník R2, ale texty z P1 a P2 vyhodnocujeme každý zvlášť. Pokud ovšem přidáme možnost spojení textů, oba texty z P1 a P2 se spojí, napárují se na R2 a vyhodnotí se dohromady.

Textovou detekci vyhodnotíme pro každou metodu zvlášť. Nejprve spočítáme F1 skóre překryvu detekovaných ploch s plochami danými referenčními texty (*Bounding Box Area*, *BBA*), a to Rovnicí 4.3, kde $\text{Přesnost}_{\text{BBA}}$ (anglicky *precision*) definujeme Rovnicí 4.1 a $\text{Úplnost}_{\text{BBA}}$ (anglicky *recall*) Rovnicí 4.2.

$$\text{Přesnost}_{\text{BBA}} = \frac{\sum \text{Obsahů průniků referenčních a predikovaných bloků}}{\sum \text{Obsahů predikovaných bloků}} \tag{4.1}$$

$$\text{Úplnost}_{\text{BBA}} = \frac{\sum \text{Obsahů průniků referenčních a predikovaných bloků}}{\sum \text{Obsahů referenčních bloků}} \tag{4.2}$$

$$\mathrm{F1}_{\mathrm{BBA}} = 2 \cdot \frac{\mathrm{Přesnost}_{\mathrm{BBA}} \cdot \mathrm{Úplnost}_{\mathrm{BBA}}}{\mathrm{Přesnost}_{\mathrm{BBA}} + \mathrm{Úplnost}_{\mathrm{BBA}}} \tag{4.3}$$

Dále zjistíme F1 skóre počtu správně detekovaných ploch (*Bounding Box Count*, *BBC*), k tomu definujeme následující pojmy: pojem TP (*True Positive*) označuje detekované textové bloky, které správně patří k nějakému referenčnímu bloku, FP (*False Positive*) značí detekované bloky neodpovídající žádnému referenčnímu bloku a FN (*False Negative*) symbolizují referenční textové bloky, ke kterým nebyl nalezen detekovaný blok.

$\mathrm{F1}_{\mathrm{BBC}}$ tedy spočítáme Rovnicí 4.6, kde $\mathrm{Přesnost}_{\mathrm{BBC}}$ definujeme Rovnicí 4.4 a $\mathrm{Úplnost}_{\mathrm{BBC}}$ Rovnicí 4.5.

$$\mathrm{Přesnost}_{\mathrm{BBC}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{4.4}$$

$$\mathrm{Úplnost}_{\mathrm{BBC}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{4.5}$$

$$\mathrm{F1}_{\mathrm{BBC}} = 2 \cdot \frac{\mathrm{Přesnost}_{\mathrm{BBC}} \cdot \mathrm{Úplnost}_{\mathrm{BBC}}}{\mathrm{Přesnost}_{\mathrm{BBC}} + \mathrm{Úplnost}_{\mathrm{BBC}}} \tag{4.6}$$

Technika spojení textů by měla ovlivnit i míru chybovosti znaků (CER) u spárovaných textů. Spočteme její průměr přes všechny texty ve vývojové množině dat pro metody se spojením textu i bez spojení textu. Spočítáme také $\mathrm{F1}_{\mathrm{BBA}}$ a $\mathrm{F1}_{\mathrm{BBC}}$ skóre.

Metrika	**Metoda bez mergování**	**Metoda s mergováním**
CER	0,297	0,25
$\mathrm{F1}_{\mathrm{BBA}}$[1]	0,495	0,493
$\mathrm{F1}_{\mathrm{BBC}}$[2]	0,543	0,529

Pozn: [1] F1 skóre překryvu ploch detekovaných při textové detekci.
[2] F1 skóre počtu detekovaných bloků při textové detekci.

Tabulka 4.1: Evaluace textové detekce různými metodami

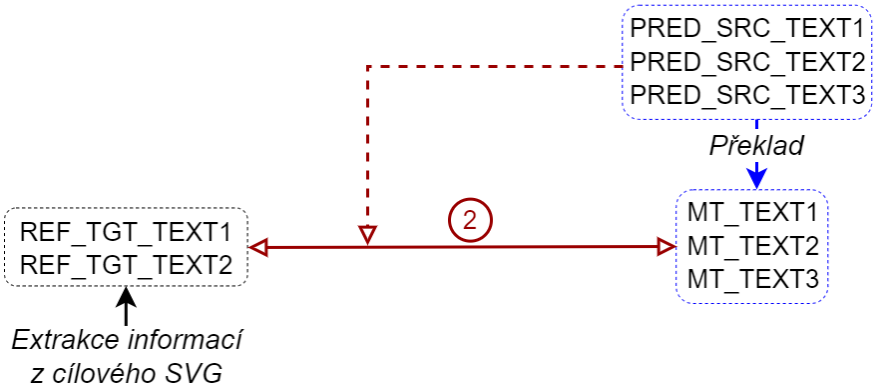
Jak uvádí Tabulka 4.1, míra chybovosti znaků pro metodu bez mergováním textu je zhruba 0,3 a pro metodu s mergování 0,25. $\mathrm{F1}_{\mathrm{BBA}}$ skóre ploch detekovaných bloků pro první metodu je 0,495, pro druhou 0,493, zatímco $\mathrm{F1}_{\mathrm{BBC}}$ skóre počtu detekovaných bloků je 0,54 pro metodu bez mergováním a 0,53 pro metodu s mergování. Můžeme si všimnout, že výsledky pro metodu se spojováním textu vychází lépe.

Toto jsou metriky, které jsou výstupem této vyhodnocovací fáze.

4.2 Evaluace překladu textů

Během vyhodnocování překladu textů postupujeme podobným způsobem, jako při evaluaci textové detekce (viz 4.1). Nejprve potřebujeme znát referenční texty REF_TGT_TEXT{1,2,...} v jazyce, do kterého překládáme původní obrázek, a

také detekované texty přeložené do cílového jazyka, texty MT_TEXT{1,2,...}, vše je znázorněno na Obrázku 4.7. Přerušovaná šipka z textů detekovaných z obrázku ve zdrojovém jazyce PRED_SRC_TEXT{1,2,...} směřující k šipce znázorňující druhý evaluační krok, tedy vyhodnocení překladu, symbolizuje potřebu textu ve zdrojovém jazyce při vyhodnocování překladu použitím nástroje COMET.



Obrázek 4.7: Znázornění evaluace překladu

K vyhodnocení je potřeba opět napárovat příslušné přeložené texty s referenčními texty z cílových SVG. Na to máme dvě možnosti, jak postupovat.

První způsob je zisk párování z předchozího kroku, viz Sekce 4.1. Z tohoto kroku máme páry detekovaný text ve zdrojovém jazyce – referenční text ve zdrojovém jazyce. Vztah detekovaného textu a jeho překladu je zřejmý, vztah zdrojového textu k překladu v referenčním SVG je jasně definovaný pomocí identifikátorů tspanID.

Druhý způsob spočívá v párování textů z vykreslených PNG obrázků, které jsou výstupem nástroje na překlad obrázků. Tuto metodu popisuje Podkapitola 4.4.

Nyní se podívejme na výsledky evaluace. V tabulce 4.2 můžeme porovnat evaluaci pomocí chrf, BLEU a COMET. Tabulka nám ukazuje, že metodu s mergováním textu byla opět úspěšnější, konkrétně s hodnotami 0,36 pro metriku BLEU, dále 28,02 pro chrF a -0,22 pro COMET.

Výstupem této evaluační fáze jsou právě tyto metriky.

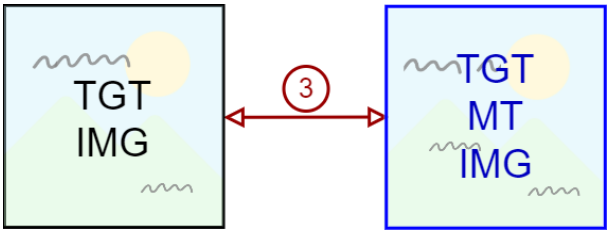
Metrika	**Metoda bez mergování**	**Metoda s mergováním**
BLEU	0,206	0,36
chrF	27,785	28,021
COMET	−0,273	−0,221

Tabulka 4.2: Evaluace překladu různými metrikami

4.3 Evaluace výstupního obrázku

Jednoduchým a přímým způsobem, jak porovnat referenční obrázek (označíme TGT IMG, viz Diagram 4.8) v cílovém jazyce s obrázkem s vloženým textem (TGT MT IMG), je vypočítat rozdíl mezi hodnotami pixelů v referenčním a

upraveném obrázku. Tento rozdíl lze kvantifikovat například pomocí střední kvadratické chyby (*Mean squared error - MSE*).



Obrázek 4.8: Znázornění evaluace výstupního obrázku

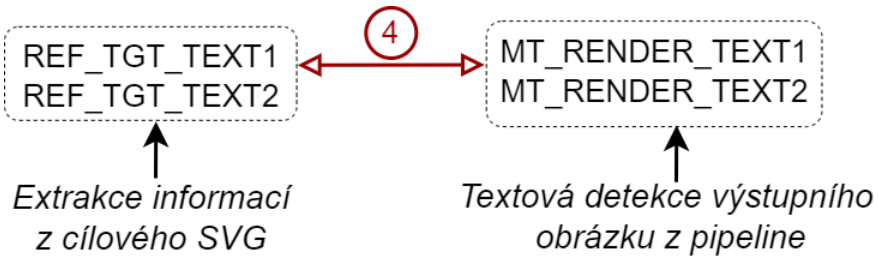
Hodnoty MSE nám však neposkytují dostatečně informativní porovnání, a proto jsme se rozhodli použít pokročilejší metodu – strukturální podobnostní index (2.9).

Po porovnání výstupních obrázků vývojové části dat s referenčními obrázky v cílových jazycích jsme dostali průměrný výsledný SSIM s hodnotou 0,88.

I přesto, že SSIM poskytuje pokročilou metodu porovnání, vizuální posouzení lidským zrakem je často nejlepším způsobem, jak hodnotit výsledky vykreslení textu na obrázku. Jedná se však pouze o subjektivní hodnocení a také není možné toto hodnocení automatizovat a získat v krátkém čase výsledky pro větší množství dat.

4.4 Evaluace detekce textu výstupního obrázku

Po vykreslení přeloženého textu zpět do obrázku můžeme provést čtvrtý krok evaluace, a to zkombinováním několika již zmíněných metod. Jak vidíme na Diagramu 4.9, nejprve detekujeme text na obrázku s přeloženým textem, čímž získáme přeložený text MT_RENDER_TEXT{1,2,...} a jeho souřadnice. Tyto obdržené informace použijeme k napárování detekovaného textu s referenčním textem REF_TGT_TEXT{1,2,...} z SVG souboru v cílovém jazyce. Postupujeme podobně jako v Podkapitole 4.1.



Obrázek 4.9: Znázornění evaluace detekce textu výstupního obrázku

Tímto způsobem vyhodnotíme nejen přesnost detekce textu a kvalitu překladu, ale i správnost vykreslení textu zpět do obrázku.

Jak můžeme vidět na Tabulce 4.3, hodnoty pro CER, $F1_{BBA}$ a $F1_{BBC}$ vychází pro metodu s mergováním textu postupně zhruba 0,51, 0,26 a 0,35.

Podobně jako v Sekci 4.1, míra chybovosti znaků, $F1_{BBA}$ a $F1_{BBC}$ skóre jsou metriky, kterými vyhodnocujeme tento krok evaluace.

Metrika	Metoda bez mergování	Metoda s mergováním
CER	0,506	0,507
$F1_{BBA}$[1]	0,255	0,257
$F1_{BBC}$[2]	0,348	0,348

Pozn: [1] F1 skóre překryvu ploch detekovaných při textové detekci.
[2] F1 skóre počtu detekovaných bloků při textové detekci.

Tabulka 4.3: Evaluace textové detekce provedené na výstupním obrázku

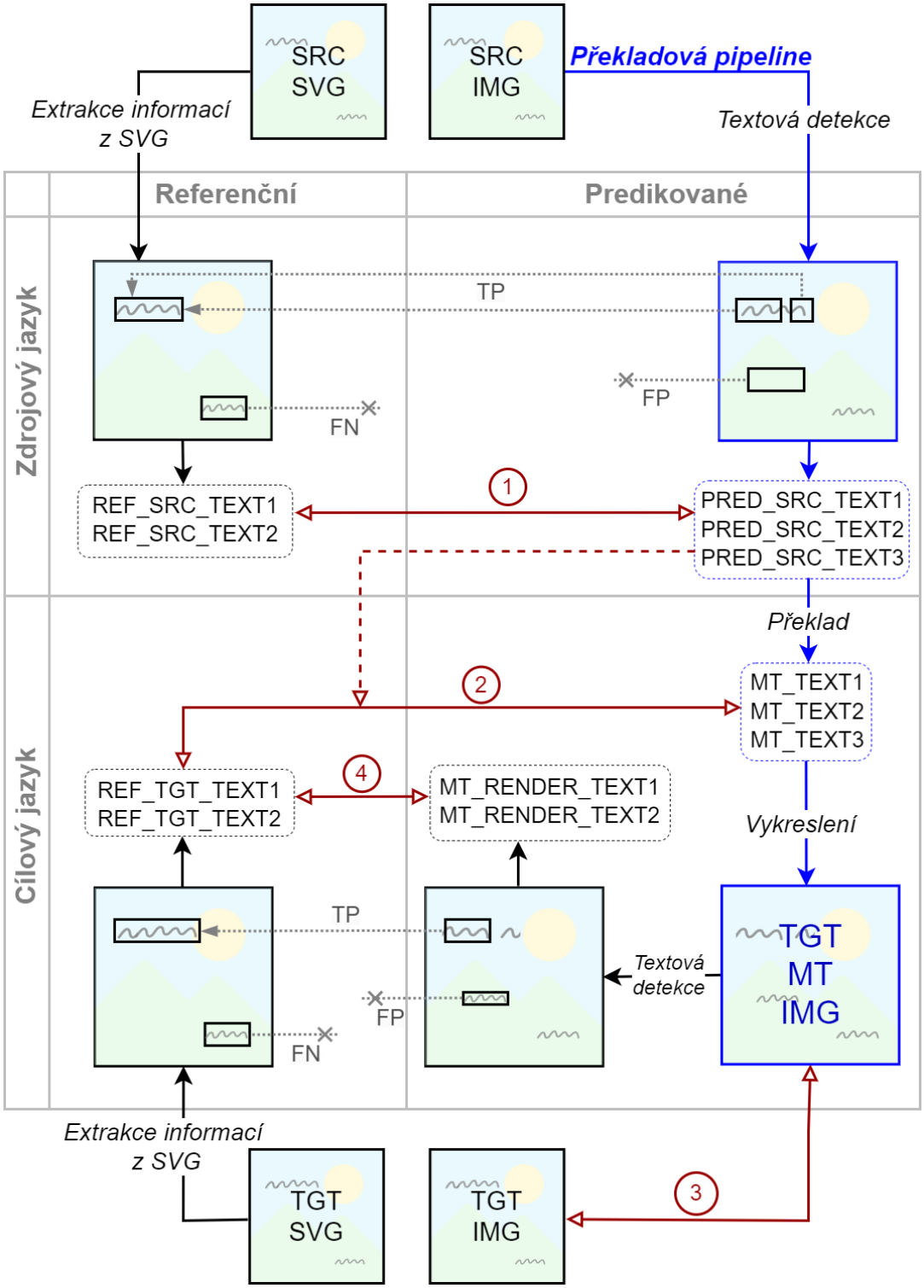
4.5 Shrnutí

Na výsledcích jednotlivých kroků evaluace se ukázalo, že metoda s mergováním textu dosahuje ve většině případech lepších výsledků. Proto je metoda s mergováním používaná v evaluačních skriptech (je nastavena jako výchozí). Verze bez mergování textu je případně dostupná, uživatel si ji může zvolit při spouštění evaluačního nástroje.

Evaluační diagram

Diagram 4.10 je celkové znázornění všech výše zmíněných kroků evaluace. Červené šipky s čísly 1 až 4 odpovídají postupně vyhodnocovacím metodám zmíněných v Sekcích 4.1 – 4.4 této kapitoly. Diagram spojuje Obrázek 4.1 zobrazující nástroj na překlad textu v obrázcích a všechna znázornění kroků evaluace, jedná se postupně o Obrázky 4.2, 4.7, 4.8 a4.9.

Symboly TP, FP a FN jsou vysvětleny v Sekci 4.1.



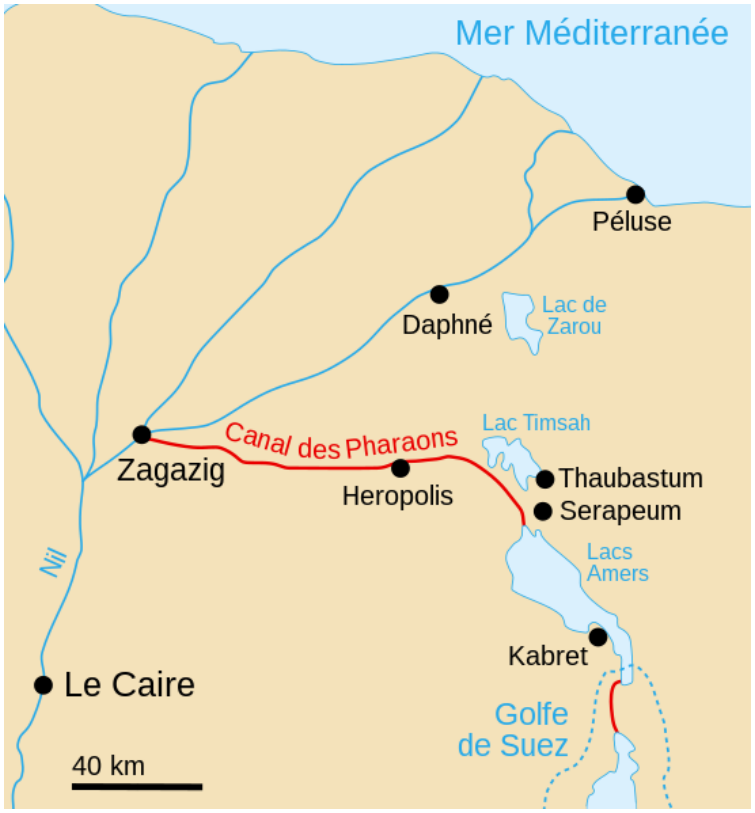
Obrázek 4.10: Znázornění evaluačních metod

5. Nástroj pro překlad textů v obrázku

V této kapitole představíme jednotlivé kroky nástroje na překlad textů v obrázku, přičemž na demonstraci použijeme Obrázek 5.1.

5.1 Vstup nástroje

Povinným vstupem nástroje je obrázek, se kterým budeme pracovat. Obrázek by měl obsahovat text, ale pokud text neobsahuje, nástroj neselže – správně by měl vrátit původní nezměněný obrázek. Nicméně může dojít k nesprávné detekci textu. Dalšími vstupy jsou kód jazyka, do kterého má být text přeložen, a kód jazyka původního obrázku.



Obrázek 5.1: Příklad vstupního obrázku ve francouzštině.

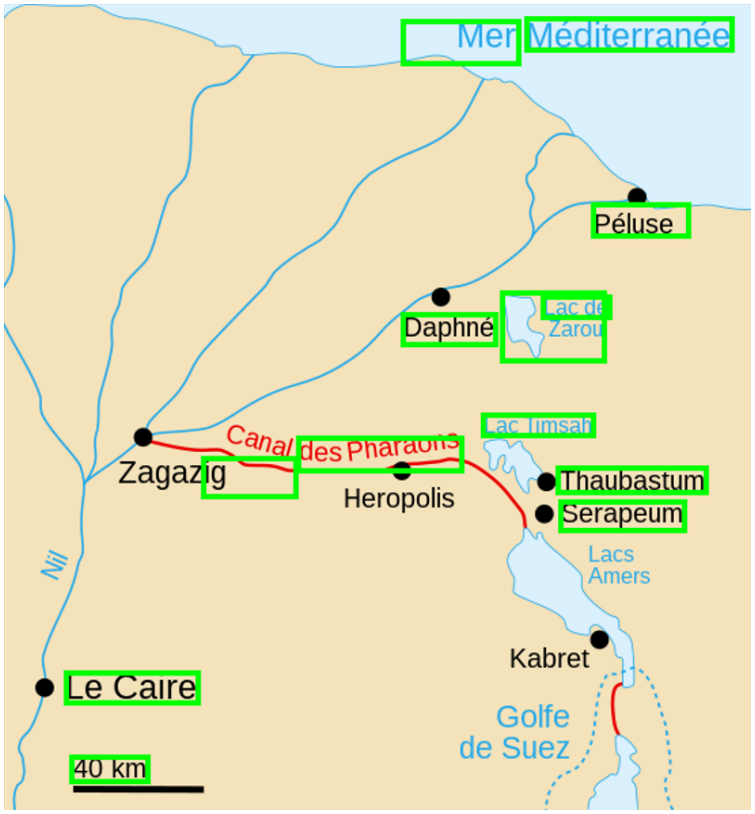
5.2 Textová detekce

Před vykonáním detekce textu je třeba obrázek předzpracovat, a to převedením obrázku z formátu BGR do RGB[1], což je formát, který vyžaduje nástroj Tesseract

[1]RGB a BGR jsou formáty, ve kterých je barva reprezentována třemi složkami v příslušném pořadí: červená (R), zelená (G) a modrá (B)

(viz 2.2) pro optické rozpoznávání znaků. Po aplikaci nástroje Tesseract jsou výsledky detekce uloženy do slovníku, který obsahuje několik klíčů:

- *text* - obsahuje samotný detekovaný text
- *left* - udává levou stranu obdélníku, který obklopuje detekovaný text, tedy x-ovou souřadnici tohoto textového boxu
- *top* - obsahuje horní stranu detekovaného obdélníku, tedy jeho y-ovou souřadnici (nebo obsahuje y-ovou souřadnici horní strany detekovaného obdélníku)
- *width* - značí šířku detekovaného obdélníku
- *height* - uvádí výšku detekovaného obdélníku
- *conf* - popisuje pravděpodobnost správné detekce textu (confidence)



Obrázek 5.2: Obrázek s detekovanými texty ohraničenými zelenou barvou

Detekované texty jsou dále zpracovány prostřednictvím analýzy jejich blízkosti. Pokud se nově detekovaný text nachází napravo v bezprostřední blízkosti předchozího textu, dochází k jejich spojení do jednoho textového bloku. Tím je efektivně řešena situace, kdy se jedná o jeden souvislý text, který byl při detekci rozdělen do více bloků. Dále dochází k jednoduché úpravě v podobě odstranění nežádoucích znaků, jako jsou mezery na kraji slova.

Po provedení detekce textu dostaneme tedy informace o detekovaných textech a o příslušných souřadnicích a tyto informace uložíme pro pozdější zpracování.

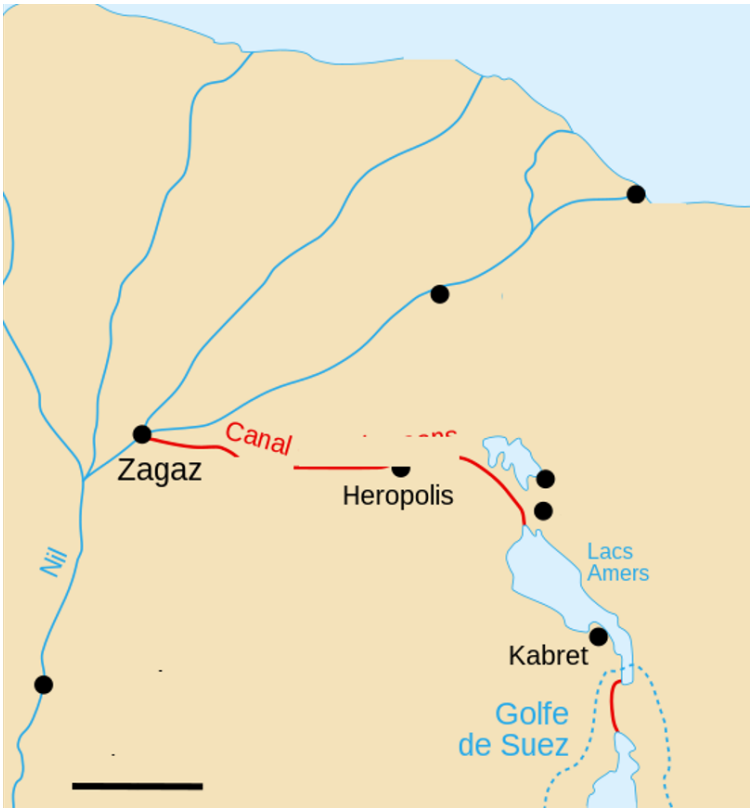
Na Obrázku 5.2 jsou zvýrazněny detekované textové bloky ohraničené zelenou barvou.

5.3 Překlad textů

Pro překlad každého detekovaného textu použijeme model M2M100 (viz 2.5). Tento model vyžaduje nejen samotný text, ale také kód jazyka, do kterého má být text přeložen, a rovněž kód jazyka, ze kterého se provádí překlad. Jednotlivé překlady jsou následně uloženy a zbývá je pouze vykreslit na obrázek.

Kvalita překladu samozřejmě závisí na mnoha faktorech, přičemž jedním z nejdůležitějších je úspěšnost textové detekce. I drobná chyba v tomto kroku, například špatně rozpoznaný znak nebo nesprávné rozdělení slova či sousloví, může mít významný dopad na výsledný překlad.

Na vzorovém obrázku můžeme například vidět, že sousloví „Mer Méditerranée“, což česky znamená Středozemní moře, bylo při detekci nesprávně rozděleno na dvě samostatná slova. Ve francouzském jazyce je slovosled odlišný, některá přídavná jména následují až po podstatném jménu, ke kterému se vztahují, na rozdíl od angličtiny. Při správné detekci by překladač přeložil „Mer Méditerranée“ jako „Mediterranean Sea“. Nicméně kvůli nesprávné detekci překladač přeloží každé slovo zvlášť, což vede k chybnému výsledku v anglickém jazyce - místo „Mediterranean Sea“ se objeví dvě samostatné fráze „The Sea“ a „The Mediterranean“, což můžeme vidět na Obrázku 5.4.



Obrázek 5.3: Obrázek po odstranění původního textu

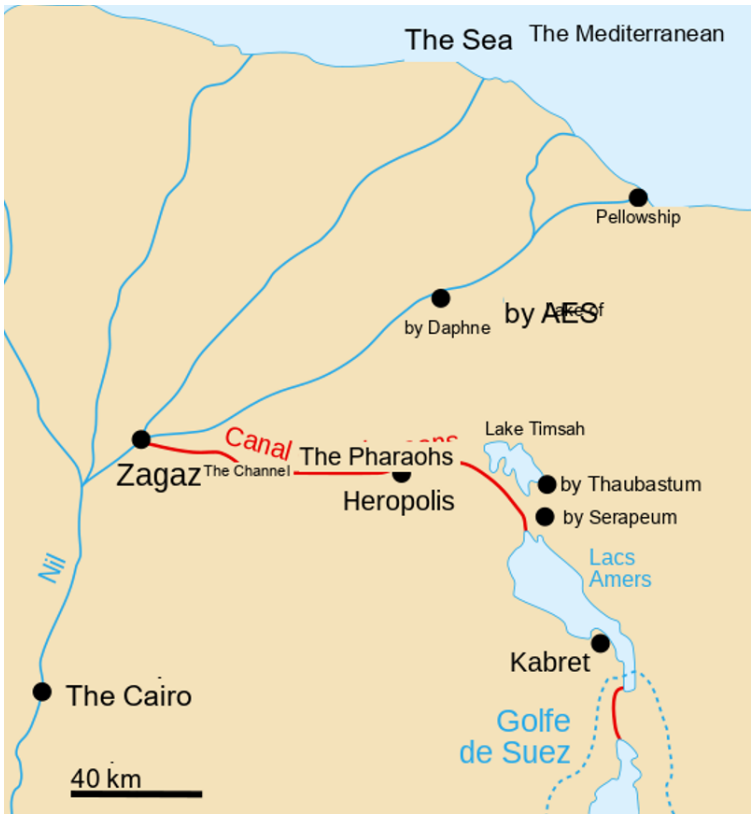
5.4 Vykreslení textu

Tato část zahrnuje dvě hlavní fáze: nejprve zakrytí původního textu na obrázcích a následné vykreslení nového textu na správná místa.

Zakrytí původního textu je náročný úkol, protože pozadí pod textem může být velmi různorodé – od jednobarevných ploch až po složité obrazce, fotografie nebo grafiky. V naší základní pipeline jsme však zvolili jednodušší přístup: původní text zakryjeme jednobarevnými obdélníky. Tento postup je relativně snadný, protože Tesseract poskytuje informace o souřadnicích a rozměrech textových bloků, tedy x-ovou a y-ovou souřadnici, výšku a šířku detekovaného textového bloku. Barvu obdélníka zakrývajícího původní text volíme podle nejčastěji se vyskytující barvy na krajích detekovaného obdélníka. Alternativně bychom mohli zvolit nejčastější barvu v celém obdélníku, ale často by se jednalo o černou barvu textu.

Samotné vykreslení nového textu je potom jednoduché, protože máme k dispozici informace o jeho umístění a rozměrech – výšce a šířce textového bloku. Velikost písma se nastaví tak, aby se text největší možné velikosti vešel do daného obdélníku. Naše aplikace by teoreticky mohla vykreslovat text v příslušných barvách a fontech, ale Tesseract nerozpoznává takové detaily, proto text vykreslujeme pouze černou barvou, jak je vidět na Obrázku 5.4.

Je důležité zmínit, že jednobarevné obdélníky, které zakrývají původní text, mohou skrýt důležité části obrázku, obzvlášť pokud byl text detekován špatně a nejedná se skutečně o text. Na Obrázku 5.3 zakrývají vykreslené obdélníky jezero vedle textu „Daphné“, což znamená, že na výsledném obrázku bude toto jezero chybět.



Obrázek 5.4: Obrázek s vykresleným přeloženým textem

5.5 Výstup nástroje

Výstupem nástroje na překlad textu v obrázcích je následující:

- zdrojový a cílový jazyk (tyto informace získáme ze vstupu)
- detekované texty
- souřadnice detekovaných textů
- detekované texty přeložené do cílového jazyka
- cesta k obrázku s vykreslenými přeloženými texty uloženému v PNG formátu

Tyto informace jsou uložené v JSON formátu a odpovídají struktuře datasetu, která je popsána v Podkapitole 3.2.

5.6 Vyhodnocení nástroje

Během vývoje jsme pracovali pouze s vývojovou částí datové sady, nyní se tedy podíváme na celkové výsledky evaluačních metrik i na testovací sadě. Tabulka 5.1 ukazuje porovnání výsledků vývojové (*Dev*) sady a testovací (*Test*) sady. Vidíme, že se výsledky mezi sadami příliš neliší, což naznačuje konzistenci výkonu našeho nástroje.

Celkově náš nástroj umožňuje vyhodnocení výkonu překladových systémů pro obrázky, které splňují specifikovanou architekturu. Uživatelé tak získají důkladný přehled o kvalitě výsledků, které nástroj poskytuje.

Evaluační metrika	**Dev**	**Test**
Evaluace textové detekce		
CER	0,25	0,213
$F1_{BBA}$	0,496	0,387
$F1_{BBC}$	0,529	0,411
Evaluace překladu		
BLEU	0,36	0,205
chrF	28,021	29,043
COMET	−0,221	−0,179
Evaluace výstupního obrázku		
SSIM	0,88	0,866
Evaluace detekce textu výstupního obrázku		
CER	0,507	0,441
$F1_{BBA}$	0,257	0,233
$F1_{BBC}$	0,348	0,296

Tabulka 5.1: Vyhodnocení překladového nástroje na testovací sadě dat

6. Uživatelská dokumentace

V této kapitole postupně popíšeme strukturu balíčku *imgmt*, instalaci projektu, práci s nástrojem pro překlad textu v obrázcích a práci s evaluačním skriptem sloužícím pro vyhodnocení výsledků z překladového nástroje.

6.1 Struktura balíčku

Na Obrázku 6.1 můžeme vidět znázornění struktury balíčku *imgmt*. Obrázek ukazuje složkovou strukturu, všechna data související s datasetem jsou ve složce `creating_dataset`. Výsledky všech kroků evaluace můžeme nalézt v CSV souborech ve složce `results`. Složka `schemas` obsahuje schémata referenčních dat (`dataset_schema.json`) a predikovaných dat (`pipeline_output_schema.json`). Dále balíček *imgmt* obsahuje skripty pro překlad textů v obrázcích, konkrétně `imgmt.py` a `imgmt_dataset.py` a skripty pro vyhodnocení výsledků z překladového nástroje `imgmt_eval.py` a `imgmt_eval_dataset.py`.

```
imgmt/
|-- creating_dataset/
|   |-- images_urls.txt
|   |-- imgmt_dataset.py
|   |-- lang_codes.txt
|   |-- split_to_dev_test.py
|-- results/
|   |-- eval_outputimage_dev.csv
|   |-- eval_outputimage_test.csv
|   |-- eval_textdetection_dev.csv
|   |-- eval_textdetection_test.csv
|   |-- eval_textdetectionrendered_dev.csv
|   |-- eval_textdetectionrendered_test.csv
|   |-- eval_translation_dev.csv
|   |-- eval_translation_test.csv
|-- schemas/
|   |-- dataset_schema.json
|   |-- pipeline_output_schema.json
|-- requirements.txt
|-- imgmt.py
|-- imgmt_all.py
|-- imgmt_eval.py
|-- imgmt_eval_all.py
```

Obrázek 6.1: Struktura balíčku *imgmt*

6.2 Požadavky

Pro správné fungování softwaru je nutné mít nainstalovaný Python. Během vývoje byla použita verze 3.10.11 – není zaručeno, že skripty budou fungovat při nižší

verzi. Dále je nezbytné mít nainstalovaný software Tesseract a Inkscape. Instalace těchto softwarových programů je popsána v Sekci 6.3.

6.3 Instalace

Chcete-li aplikaci nastavit a spustit, postupujte podle následujících kroků.

Nejprve stáhněte a uložte ZIP složku s projektem do svého zařízení a extrahujte ji.

Pokud ještě nemáte nainstalovaný software Tesseract, stáhněte a nainstalujte správnou verzi pro vaši platformu:

- Pro Windows: Stáhněte instalační balíček z oficiálních stránek Tesseract OCR[1] a postupujte podle pokynů pro instalaci.
- Pro Linux: Nainstalujte Tesseract pomocí správce balíčků vaší distribuce, například příkazem `sudo apt-get install tesseract-ocr`.

Ujistěte se, že je Tesseract přístupný spuštěním příkazu `tesseract --version` v příkazovém řádku.

Dále nainstalujte Inkscape:

- Pro Windows: Stáhněte instalační balíček z oficiálních stránek Inkscape[2] a postupujte podle pokynů pro instalaci.
- Pro Linux: Nainstalujte Inkscape pomocí správce balíčků vaší distribuce, například příkazem `sudo apt-get install inkscape`.

Zkontrolujte, že máte nainstalovaný Python verze 3.10.11 nebo vyšší.

Jakmile jsou všechny závislosti nainstalovány, přesuňte se do kořenového adresáře a v příkazovém řádku spusťte příkaz:

```
pip install -r requirements.txt
```

Nyní se přesuňte do složky `creating_dataset` a spusťte `imgmt_dataset.py`. Následně spusťte skript `split_to_dev_test.py`, který již stažený dataset rozdělí na vývojovou a testovací sadu.

Ostatní Python skripty, tedy `imgmt.py`, `imgmt_all.py`, `imgmt_eval.py` a `imgmt_eval_all.py` spouštějte z kořenového adresáře.

6.4 Spuštění nástroje na překlad obrázků

Pro přeložení textu z jednoho obrázku použijte skript `imgmt.py`. Tento skript má následující parametry:

- `-i`, `--image`: Cesta ke vstupnímu obrázku
- `-s`, `--source-lang`: Zdrojový jazyk obrázku – jazykový kód

[1] `https://sourceforge.net/projects/tesseract-ocr.mirror/`

[2] `https://inkscape.org/release/inkscape-1.3.2/windows/64-bit/msi/?redirected=1`

- `-t`, `--target-lang`: Cílový jazyk obrázku – jazykový kód
- `-o`, `--output`: Cesta k JSON souboru s výstupními daty (výchozí = *pipeline_output.json*)
- `-oi`, `--output-image`: Cesta k výstupnímu obrázku s vykresleným textem (výchozí = *output_img.png*)
- `--tesseract-cmd`: Cesta ke spustitelnému souboru Tesseract (výchozí = `r'C:\Program Files\Tesseract-OCR\tesseract.exe'` – pro Windows)

Parametry `--image`, `--source-lang` a `--target-lang` jsou povinné, zatímco parametry `--output` a `--output-image` jsou nepovinné.

Vstupem je obrázek s textem. Obrázek by měl být ve formátu PNG a neměl by obsahovat složité grafiky. Nástroj je zaměřen na rastrové obrázky nikoliv na fotografie nebo naskenované dokumenty. Skript rovněž očekává, že text na obrázku bude v jazyce, který uživatel uvede jako zdrojový jazyk.

Výstupem skriptu `imgmt.py` je obrázek ve formátu PNG uložený buď jako *output_img.png*, nebo podle uživatelem zadané cesty pomocí `--output-image`. Dále je výstupem JSON soubor s informacemi, jako je detekovaný a přeložený text, souřadnice textu a cesta k vykreslenému obrázku.

Pokud chceme zpracovat celý dataset a provést pipeline na všech obrázcích vývojové, případně testové sady datasetu, použijeme skript `imgmt_dataset.py`. Při práci budeme využívat následujících parametrů:

- `-d`, `--dataset-part`: Určuje, na kterou část datasetu se má skript použít, možnosti jsou `dev` a `test`
- `-tdr`, `--text-detect-render`: Skript provede detekci textu na výstupním obrázku
- `--tesseract-cmd`: Cesta ke spustitelnému souboru Tesseract (výchozí = `r'C:\Program Files\Tesseract-OCR\tesseract.exe'`)

6.5 Spuštění evaluačního nástroje

Evaluační skript `imgmt_eval.py` umožňuje vyhodnotit přesnost a kvalitu různých fází zpracování a překladu textu v obrázcích. Pro spuštění evaluačního skriptu použijeme následující parametry:

Parametry

- `-r`, `--reference`: Cesta k souboru s referenčními daty
- `-p`, `--predicted`: Cesta k souboru s predikovanými daty k vyhodnocení
- `-td`, `--text-detect`: Provede se vyhodnocení detekce textu
- `-tdr`, `--text-detect-render`: Provede se vyhodnocení detekce textu z výstupního obrázku
- `-tb`, `--translate-bleu`: Provede se vyhodnocení překladu textu pomocí BLEU

- `-tch`, `--translate-chrf`: Provede se vyhodnocení překladu textu pomocí chrF
- `-tc`, `--translate-comet`: Provede se vyhodnocení překladu textu pomocí COMET
- `-ri`, `--render-image`: Provede se vyhodnocení vykresleného obrázku
- `--no-merge`: Provede se vyhodnocení bez metody spojení textu

Výstupem skriptu `imgmt_eval.py` jsou hodnoty zjištěné během evaluace. Ty jsou zobrazeny přímo v příkazové řádce.

Pokud chceme, podobně jako v Sekci 6.4, zpracovat celý dataset a provést evaluační metody na všech obrázcích vývojové, případně testové sady datasetu, použijeme skript `imgmt_eval_all.py`. Tento skript podporuje následující parametry:

- `-d`, `--dataset-part`: Určí, na kterou část datasetu se má skript použít, možnosti jsou `dev` a `test`
- `-td`, `--text-detect`: Provede se vyhodnocení detekce textu
- `-tdr`, `--text-detect-render`: Provede se vyhodnocení detekce textu z výstupního obrázku
- `-t`, `--translate`: Provede se vyhodnocení překladu textu pomocí BLEU, chrF a COMET
- `-ri`, `--render-image`: Provede se vyhodnocení vykresleného obrázku
- `--no-merge`: Provede se vyhodnocení bez metody spojení textu

6.6 Práce s daty

6.6.1 Struktura datasetu

Obrázek 6.2 nastiňuje strukturu datasetu používaného v této práci. Dataset se vytvoří po spuštění skriptu `imgmt_dataset.py` (bez použití parametrů). Je rozdělen na dvě části: vývojovou (`dev`) a testovací (`test`) sadu. Každá z těchto sad obsahuje složky označené čísly, které shromažďují data a obrázky vzájemných překladů.

Každá číselná složka obsahuje složku (`png`) s danými obrázky a také JSON soubory (např. `en-de.json` nebo `fr-cs.json`), které obsahují referenční data úlohy překladu obrázků. Po spuštění skriptu `imgmt_all.py` je každá číselná složka doplněna o složkou `pipeline_output`, která obsahuje složku `render_image` s přeloženými a vykreslenými texty a dále JSON soubory s výstupními daty z překladového nástroje.

```
data/
|-- dev/
|   |-- {cisloJazykoveSkupiny}/
|   |   |-- png/
|   |   |   |-- {jazykovyKod}.png
|   |   |   |-- ...
|   |   |-- pipeline_output/
|   |   |   |-- render_png/
|   |   |   |   |-- {jazykovyKodZdrojovy}-{jazykovyKodCilovy}.png
|   |   |   |   |-- ...
|   |   |   |-- {jazykovyKodZdrojovy}-{jazykovyKodCilovy}.json
|   |   |   |-- ...
|   |   |-- {jazykovyKodZdrojovy}-{jazykovyKodCilovy}.json
|   |   |-- ...
|-- test/
|   |-- Stejná struktura jako dev složka
```

Obrázek 6.2: Struktura datasetu

7. Vývojová dokumentace

Tato dokumentace podrobně popisuje proces překladu textu z obrázků a hodnocení výkonu překladové pipeline na správně zvoleném datasetu.

Pro vývoj jsme zvolili jazyk Python, protože, ačkoliv není nejrychlejší, je široce využíván v úlohách strojového překladu a nabízí podporu pro nástroje, které jsme potřebovali.

Pro práci využíváme následující knihovny:

- pytesseract[1]: Pro optické rozpoznávání znaků na obrázcích.
- OpenCV[2]: Pro manipulaci s obrázky. Nástroj Tesseract vyžaduje načtení obrázku právě pomocí OpenCV. Tato knihovna umožňuje i vykreslení textu do obrázku, tato funkce je však omezená pouze na znaky patřící do *ASCII tabulky*[3], což je při práci s mnoha jazyky velice omezující.
- PIL[4]: Pro práci s obrázky. Narozdíl od knihovny OpenCV umožňuje větší množství znaků.
- dl-translate[5]: Pro strojový překlad textu, využíváme model M2M100.
- Comet[6]: Slouží pro vyhodnocování strojového překladu.
- sacrebleu[7]: Pro hodnocení kvality překladu pomocí BLEU skóre a skóre chrF.
- scikit-image[8]: Využíváme pro analýzu podobnosti obrázků.
- BeautifulSoup4[9]: Pro zpracování XML souborů, konkrétně SVG souborů.

7.1 Skript na vytvoření datasetu

Skript `imgmt_dataset.py` je navržen pro vytvoření datasetu z PNG souborů, které obsahují textové bloky, které jsou si navzájem překlady. Cílem je extrahovat textové bloky a jejich souřadnice z SVG souborů, stáhnout odpovídající PNG soubory a vytvořit JSON soubory, které obsahují tyto informace pro každý jazykový pár.

Třída `SVGHandler` pracuje s SVG soubory a extrahuje z nich informace. Třída `DatasetMaker` spravuje celý proces stahování, extrakce dat a vytváření JSON souborů.

[1] `https://pypi.org/project/pytesseract/`
[2] `https://pypi.org/project/opencv-python/`
[3] ASCII je zkratka pro „American Standard Code for Information Interchange“. Tento systém kóduje pouze 128 znaků.
[4] `https://pypi.org/project/pillow/`
[5] `https://pypi.org/project/dl-translate/`
[6] `https://pypi.org/project/comet-ml/`
[7] `https://pypi.org/project/sacrebleu/`
[8] `https://pypi.org/project/scikit-image/`
[9] `https://pypi.org/project/beautifulsoup4/`

První se inicializuje objekt `DatasetMaker` a volá se funkce `get_dataset()`. Tato funkce načte jazykové kódy ze souboru *lang_codes.txt* a URL obrázků z *images_urls.txt*, stáhne SVG a PNG soubory a volá `create_data_pair()` pro každý jazykový pár. Funkce `create_data_pair()` používá SVGHandler pro získání souřadnic textových bloků a textů z SVG souborů a následně volá funkci `create_json()` pro uložení dat do JSON souboru. `SVGHandler` používá funkci `inkscape_command()` pro získání informací o tspan elementech SVG souborů a `get_bounding_boxes_and_texts()` pro extrakci textů a jejich souřadnic.

Skript `split_dev_test.py` následně rozdělí dataset na vývojovou a testovací část podle velikostí skupin.

7.2 Překladové a vyhodnocovací nástroje

Modul `imgmt.py` představuje nástroj pro překlad textu v obrázcích. Jeho hlavní třída `ImagePipeline` obsahuje klíčové metody pro zpracování obrázku, detekci textu, překlad a vykreslení přeloženého textu. Klíčové atributy třídy zahrnují:

- `path`: Cesta k vstupnímu obrázku
- `img`: Objekt obrázku načtený pomocí OpenCV
- `texts`: Detekovaný text na obrázku
- `blocks_coordinates`: Souřadnice textových bloků na obrázku.
- `translated_texts`: Přeložený text
- `image_to_save`: Obrázek s vykresleným přeloženým textem

Skript `imgmt_all.py` iteruje přes složky v datasetu. Importuje modul `imgmt`, který je využíván pro překlad každého obrázku v datasetu.

Funkce `process_directory(dir_name, mt)` provádí následující:

- Prochází všechny složky ve zvoleném adresáři `dir_name` a zpracovává obrázky.
- Pro každý jazykový kód v `dir_name` provádí detekci textu, přeloží text do ostatních jazyků přítomných ve složce a nakonec přeložený text vykreslí na obrázek.
- Ukládá výstupy do JSON souborů obsahujících detekovaný a přeložený text a vykreslené obrázky.

Modul `imgmt_eval.py` slouží k hodnocení výkonu překladové pipeline. Testujeme schopnost tohoto skriptu na výstupech, které jsme získali díky skriptu `imgmt_all.py`. Klíčové komponenty zahrnují:

- Hodnocení detekce textu: Detekci textu provádíme na vstupním i výstupním obrázku. Funkce `evaluate_text_detection(merging, rendered)` porovnává detekované textové bloky s referenčními bloky a vyhodnocuje F1 skóre obsahů a počtu detekovaných textových ploch. Parametr `merging` určuje, zda budeme užívat metody spojování textu, a parametr `rendered` určuje, jestli se bude vyhodnocovat textová detekce vstupního, nebo výstupního obrázku.

- Hodnocení překladu: `evaluate_translation(metric, merging, model)` vyhodnocuje kvalitu přeloženého textu pomocí metrik BLEU, chrF a COMET.
- Hodnocení vykreslení: Funkce `evaluate_output_image()` používá SSIM index pro porovnání vykreslených obrázků s referenčními obrázky.

Python skript `imgmt_eval_all.py` je navržen pro evaluaci překladové pipeline v rámci datasetu. Skript iteruje přes složky v datasetu, aplikuje evaluační nástroje a ukládá výsledky do CSV souborů. Následně vypočítává průměrné hodnoty metrik pro každou z těchto oblastí.

Třída `EvaluationDatasetProcessor` je zodpovědná za zpracování datasetu a provádění hodnocení. Funkce `process_directory(self, dir_name, method, results, merging, rendered, model)` prochází obrázky v jednotlivých složkách, načítá odpovídající JSON soubory obsahující predikovaná a referenční data, a vyhodnocuje metriky na základě zadané metody.

Funkce pro výpočet průměrných hodnot z vyhodnocení (`calculate_avg_TD`, `calculate_avg_Tr`, `calculate_avg_SSIM`) čtou CSV soubory s výsledky a vypočítávají průměrné hodnoty metrik.

Závěr

V práci jsme se zaměřili na vyhodnocení nástrojů pro překlad textu v obrázcích. Navrhli jsme evaluační schéma, což zahrnovalo sběr dat a implementaci evaluačního skriptu. Rovněž jsme implementovali základní překladový nástroj, na kterém jsme demonstrovali použití navrženého evaluačního schématu.

Klíčovým úkolem práce bylo vytvoření datasetu obsahujícího obrázky s textem v různých jazycích, které jsou si navzájem překlady. Data jsme získali z veřejně dostupných obrázků na Wikipedii a systematicky je strukturovali pro účely naší analýzy. Prvním krokem byl sběr URL obrázků ve formátech PNG a SVG pro náš dataset, následovaný navržením vhodné struktury a extrakcí textových informací z SVG obrázků. Výzvou bylo správné určení souřadnic textu v obrázku, což je nezbytné pro správnou interpretaci textu při jeho detekci a překladu. Nakonec jsme data rozdělili na vývojovou a testovací sadu.

Stěžejní částí naší práce bylo vytvoření evaluačního skriptu, kterého cílem je testovat úspěšnost jednotlivých kroků systémů pro překlad textů v obrázcích. Tento skript zahrnuje evaluaci detekce textu, překladu textu a vykreslení přeloženého textu zpět do obrázku, stejně jako další detekci textu výsledného obrázku. Během vývoje jsme přistupovali ke dvěma metodám vyhodnocování – buď jsme detekovaný text vyhodnocovali rovnou bez žádných úprav, nebo jsme zkoušeli detekovaný text odpovídající jednomu referenčnímu textu spojovat, čímž jsme dosáhli lepších výsledků a zvolili tuto metodu jako výchozí při vyhodnocování. Metody jsme vyvíjeli na vývojové části datasetu a poté finálně vyzkoušeli na testovacích datech.

Dále jsme implementovali samotný nástroj pro překlad textů v obrázcích, který využívá existující metody jako *Tesseract* pro detekci textu nebo model *M2M100* pro strojový překlad. Nástroj přijímá obrázek s textem ve zdrojovém jazyce a kód cílového jazyka, provádí detekci textu, přeloží ho, zakryje původní text a vykreslí přeložený text zpět do obrázku, který je poté výstupem překladové pipeline.

Budoucí práce

V budoucnu se naše práce může značně rozšířit. Jedním z kroků by bylo přidání dalších veřejně dostupných obrázků do datasetu. Dále bychom chtěli zkoumat jiné metody a metriky, například při překladu by se model *M2M100* mohl nahradit jiným větším modelem, který podporuje více jazyků.

Pokročilé zpracování by mělo zajistit i vylepšení detekce textu, které by mohlo zahrnovat rozpoznání různých fontů, stylů a barev textu. Tato funkcionalita by byla klíčová pro přesnější a flexibilnější vykreslování přeloženého textu zpět do obrázků.

Naše práce by tak mohla nejen poskytnout lepší nástroje pro překlad textů v obrázcích, ale také přispět k dalšímu rozvoji metod pro jejich efektivní analýzu a interpretaci.

Seznam použité literatury

Fan, A., Bhosale, S., Schwenk, H., Ma, Z., El-Kishky, A., Goyal, S., Baines, M., Celebi, O., Wenzek, G., Chaudhary, V., Goyal, N., Birch, T., Liptchinsky, V., Edunov, S., Grave, E., Auli, M. a Joulin, A. (2021). Beyond english-centric multilingual machine translation. *J. Mach. Learn. Res.*, **22**(1). ISSN 1532-4435.

Ferraiolo, J., Jun, F. a Jackson, D. (2000). *Scalable vector graphics (SVG) 1.0 specification*. iuniverse Bloomington.

Kirsanov, D. (2021). *The Book of Inkscape, 2nd Edition: The Definitive Guide to the Graphics Editor*. No Starch Press. ISBN 9781718501751. URL `https://books.google.cz/books?id=HFxEEAAAQBAJ`.

Nilsson, J. a Akenine-Möller, T. (2020). Understanding ssim. URL `https://arxiv.org/abs/2006.13846`.

Papineni, K., Roukos, S., Ward, T. a Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In Isabelle, P., Charniak, E. a Lin, D., editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL `https://aclanthology.org/P02-1040`.

Popović, M. (2015). chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395.

Rei, R., Stewart, C., Farinha, A. C. a Lavie, A. (2020). COMET: A neural framework for MT evaluation. *arXiv preprint arXiv:2009.09025*.

Smith, R. (2007). An Overview of the Tesseract OCR Engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 629–633. doi: 10.1109/ICDAR.2007.4376991.

Tam, T. (2021). Deciphering accuracy evaluation metrics in nlp and ocr: A comparison of character error rate (cer). `https://medium.com/@tam.tamanna18/deciphering-accuracy-evaluation-metrics-in-nlp-and-ocr-a-comparison-of-chara` Accessed: 2024-07-16.

Seznam obrázků

1.1 Příklad úlohy z webových stránek Školy s Nadhledem 4
1.2 Úloha ze Školy s Nadhledem přeložená do ukrajinštiny 5

2.1 Text s pevnou šířkou písma (Smith, 2007) 8

3.1 Šablona Information 13
3.2 Příklad dotazu s odpovědí z MediaWiki Action API 14
3.3 JSON struktura datasetu 16
3.4 Znázornění napárování textů a překladu textů z SVG souborů . . 18
3.5 Statistika velikostí skupin obrázků lišících se jen v jazyce textů . . 20

4.1 Znázornění překladového nástroje 21
4.2 Znázornění evaluace textové detekce 22
4.3 Ukázka části obrázku před textovou detekcí 22
4.4 Ohraničení textu podle souřadnic z nástroje textové detekce . . . 22
4.5 Ohraničení textu podle referenčních souřadnic 22
4.6 Hledání největšího překryvu pro napárování textů z obrázku 4.3 . 23
4.7 Znázornění evaluace překladu 25
4.8 Znázornění evaluace výstupního obrázku 26
4.9 Znázornění evaluace detekce textu výstupního obrázku 26
4.10 Znázornění evaluačních metod 28

5.1 Příklad vstupního obrázku ve francouzštině. 29
5.2 Obrázek s detekovanými texty ohraničenými zelenou barvou . . . 30
5.3 Obrázek po odstranění původního textu 31
5.4 Obrázek s vykresleným přeloženým textem 32

6.1 Struktura balíčku *imgmt* 34
6.2 Struktura datasetu 38

Seznam tabulek

3.1 Statistika jazyků v datasetu 19
3.2 Statistika jazykových párů v datasetu (nejpočetnější) 20

4.1 Evaluace textové detekce různými metodami 24
4.2 Evaluace překladu různými metrikami 25
4.3 Evaluace textové detekce provedené na výstupním obrázku 27

5.1 Vyhodnocení překladového nástroje na testovací sadě dat 33

A. Přílohy

A.1 Elektronicky přiložené dokumenty

- Balíček *imgmt* se zdrojovými kódy v jazyce Python
- Výsledky evaluací textové detekce, překladu textů, výstupního obrázku a detekce textu z výstupního obrázku