KNIHOVNA AKADEMIE VĚD ČR | MORAVSKÁ ZEMSKÁ KNIHOVNA | FAKULTA INFORMAČNÍCH TECHNOLOGIÍ | Národní knihovna České republiky

Technická dokumentace k software AnnoPage – systém pro automatickou anotaci objektů na stránce

Martin Kišš, Michal Hradiš, Martina Dvořáková, Petr Žabička, Filip Jebavý, Benjamin Lapoš, Boris Lehečka, Jana Hrzinová, Václav Jiroušek, Filip Pavčík, Filip Kersch, Markéta Herudková, Anna Najmanová, Martin Lhoták

2025

Tento dokument byl vytvořen s finanční podporou MK ČR v rámci programu **NAKI III program na podporu aplikovaného výzkumu v oblasti národní a kulturní identity na léta 2023 až 2030**

v projektu Orbis Pictus – oživení knihy pro kulturní a kreativní odvětví.

Číslo a název projektu:

DH23P03OVV033	Orbis Pictus – oživení knihy pro kulturní a kreativní odvětví

Název a popis dílčího výstupu:

AnnoPage – systém pro automatickou anotaci objektů na stránce
Tento dokument popisuje funkčnost a použití software AnnoPage, který umožňuje detekovat netextové elementy a určit jejich kategorii (mapa, fotografie, graf, atd.). Systém dále detekuje titulky těchto elementů na stránce a přiřazuje je ke konkrétním netextovým elementům. Detekové netextové elementy jsou také zpracovány pomocí modelů dostupných přes komerční API, které generují textový popis elementu, titulek, klíčová slova a informaci o barevnosti. Systém také umožňuje vytvářet sémantickou reprezentaci elementů (embeddingy).

Jazyk dokumentu

Čeština

Organizace a řešitel

Knihovna AV ČR, v. v. i.	Ing. Martin Lhoták
Moravská zemská knihovna v Brně	Mgr. Filip Jebavý
Vysoké učení technické v Brně	Ing. Michal Hradiš Ph.D.
Národní knihovna ČR	Bc. Václav Jiroušek

Obsah

Obecné informace

AnnoPage je nástroj pro zpracování dokumentů, který umožňuje detekci netextových prvků na stránce, identifikaci jejich popisu v textu, vytvoření krátkého popisu, názvu a seznamu klíčových slov pomocí LLM a vytvoření jejich sémantické vektorové reprezentace (embedding).

Implementace AnnoPage poskytuje jak samotný zpracovávací nástroj a konfigurace s natrénovanými modely, tak také implementaci REST API serveru a obslužnými nástroji (client, worker).

Dostupnost a instalace

Veškerá data k nástroji AnnoPage jsou dostupná v GitHub repozitáři https://github.com/LIBCAS/AnnoPage. Tento repozitář obsahuje veškerou implementaci i natrénované modely. Implementaci je možné nainstalovat přímo z GitHub repozitáře pomocí příkazu:

```
pip install "anno-page @ git+https://github.com/LIBCAS/AnnoPage.git"
```

Vzhledem k tomu, že repozitář obsahuje i natrénované modely, tak, byť jsou uloženy v rámci Git LFS, dojde k jejich stažení společně s veškerými zdrojovými kódy, což může být zdlouhavé. K této situaci nedojde pokud nemáte nainstalovaný nástroj `git lfs` nebo při nastavení proměnné prostředí, která způsobí, že se soubory z Git LFS nebudou stahovat:

```
UNIX/Linux:
export GIT_LFS_SKIP_SMUDGE=1

Windows (PowerShell):
$env:GIT_LFS_SKIP_SMUDGE = "1"
```

Tím, že je v rámci repozitáře implementováno více různých spustitelných částí, které vyžadují různé závislosti, nejsou v základu nainstalovány žádné závislosti. Je však možné nastavit, pro kterou část se mají závislosti nainstalovat. Tyto možnosti jsou:

- `client`
- `worker`
- `api`
- `tool`
- `full`

Při použití možnosti `client`, `worker` a `api` dojde k nainstalování závislostí, které jsou potřeba pro spuštění klienta, workera a REST API. Možnost `tool` nainstaluje závislosti pro spuštění zpracovávacího nástroje a při použití možnosti `full` dojde k nainstalování všech závislostí pro všechny části. Pro nainstalování všech závislostí by pak příkaz vypadal takto:

```
pip install "anno-page[full] @ git+https://github.com/LIBCAS/AnnoPage.git"
```

Po nainstalování AnnoPage dojde k vytvoření spustitelných programů pro příkazový řádek:

- `annopage(.exe)`
- `annopage_api(.exe)`
- `annopage_client(.exe)`
- `annopage_worker(.exe)`

Informace ke spouštění všech těchto nástrojů je popsáno dále v příslušných částech.

Licence

Software: GPL-3.0 license
Modely: GPL-3.0 license

Požadavky

Pro spuštění AnnoPage je vyžadován Python ve verzi 3.10 nebo vyšší spolu s řadou standardních balíčků. Seznam závislostí je uveden v souboru `pyproject.toml` v rámci GitHub repozitáře. Zpracovávací nástroj AnnoPage je možné spouštět přímo na CPU, avšak z důvodu rychlosti zpracování je doporučeno spouštět jej na GPU s alespoň 4GB GPU RAM, kde probíhá inference modelů mnohem rychleji. Používání OpenAI modelů vyžaduje platný OpenAI API klíč.

Nástroj AnnoPage byl vyvíjen a testován na OS Linux (Ubuntu 22.04) a Python verze 3.12. Momentálně není nijak zajištěno, že bude tento software spustitelný také na OS Windows, macOS, nebo jiných systémech, s výjimkou AnnoPage klienta, který byl testován také na OS Windows 11.

AnnoPage

AnnoPage je nástroj příkazového řádku určený ke zpracování netextových prvků na stránkách dokumentů. Nástroj pracuje s dávkami obrázků, což znamená, že vstupní cesty musí být adresáře, nikoli jednotlivé soubory (s výjimkou konfiguračního a metadatového souboru). Zpracování vyžaduje konfigurační soubor, který definuje parametry zpracování a v případě potřeby specifikuje cesty k modelům, které mají být použity. Jakmile je prostředí a konfigurace připravena, lze AnnoPage spustit příkazem `annopage`:

```
annopage \
  --config=config.ini \
  --input-image-path=input/images/dir \
  --output-alto-path=output/alto_xmls/dir \
  --output-render-path=output/renders/dir
```

Parametry příkazové řádky

```
-h, --help
```

Program vypíše nápovědu a skončí.

```
--config CONFIG
```

Cesta ke konfiguraci zpracování.

```
--input-image-path INPUT_IMAGE_PATH
--input-xml-path INPUT_XML_PATH
--input-alto-path INPUT_ALTO_PATH
```

Parametry pro specifikaci cest k adresářům se vstupními obrázky, PAGE XML, ALTO XML. Parametry pro PAGE XML a ALTO XML není možné kombinovat (může být zadán maximálně jeden z nich).

```
--input-metadata-path INPUT_METADATA_PATH
```

Parametr pro specifikaci cesty k souboru s metadaty (JSON).

```
--output-xml-path OUTPUT_XML_PATH
--output-alto-path OUTPUT_ALTO_PATH
--output-render-path OUTPUT_RENDER_PATH
--output-crops-path OUTPUT_CROPS_PATH
--output-image-captioning-prompts-path OUTPUT_IMAGE_CAPTIONING_PROMPTS_PATH
--output-embeddings-path OUTPUT_EMBEDDINGS_PATH
--embeddings-jsonlines
```

Parametry pro specifikaci výstupních dat - PAGE XML, ALTO XML, rendery s detekovanými netextovými elementy, výřezy, prompty použitými k popisování netextových elementů a JSONy se sémantickými embeddingy. Při nastavení příznaku --embeddings-jsonline je výstup s embeddingy uložen jako JSON Lines (jsonl) soubor.

```
-s, --skip-processed
```

Příznak pro přeskočení již zpracovaných dokumentů při opětovném spuštění.

```
--device {gpu,cpu}
--gpu-id GPU_ID
```

Nastavení zařízení, na kterém budou probíhat výpočty PyTorch modelů.

```
--process-count PROCESS_COUNT
```

Počet procesů pro paralelní zpracování. Pokud je pouze nastaven jeden proces (výchozí hodnota), pak probíhá zpracování sekvenčně.

```
--logging-level LOGGING_LEVEL
```

Parametr pro nastavení úrovně logování.

Vstupní data

Konfigurace

Soubor s konfigurací slouží k definování zpracování. Tento soubor je formátován jako INI soubor, kde každá sekce odpovídá jednomu kroku zpracování (moduly). Ukázková konfigurace, která se skládá z modulů pro detekci netextových elementů, detekci a přiřazení popisků netextových elementů, vygenerování popisu, titulku, klíčových slov a analýzu barevnosti pomocí ChatGPT a vytvoření sémantických embeddingů pomocí CLIP modelu, vypadá následovně:

```
[OPERATION_1]
METHOD = YOLO_DETECTION
MODEL_PATH = ./anno_page.pt
DETECTION_THRESHOLD = 0.283
IMAGE_SIZE = 1024

[OPERATION_2]
METHOD = CAPTION_YOLO_ORGANIZER
YOLO_PATH = ./annopage_captions_detector.pt
YOLO_IMAGE_SIZE = 1024
YOLO_DETECTION_THRESHOLD = 0.499
ORGANIZER_PATH = ./annopage_captions_organizer.pt
ORGANIZER_CATEGORIES = ["Padding", "Image", "Photograph", "Caricature and
comics", "Stamp", "Barcode and QR code", "Symbol, logo, coat of arms",
"Vignette", "Frieze", "Signet", "Initial", "Other book decor", "Decorative
inscription", "Musical notation", "Table", "Map", "Graph", "Geometric
drawing", "Other technical drawing", "Diagram", "Floor plan", "Mathematical
expression and equation", "Chemical formula and equation", "Exlibris",
"Advertisement", "Handwritten note", "Image caption"]

[OPERATION_3]
METHOD = GPT_IMAGE_CAPTIONING
API_KEY = openai_api_key.txt
PROMPT_SETTINGS = image_captioning_prompt.json
CATEGORIES = ["Photograph", "Geometric drawing", "Graph", "Caricature and
comics", "Map", "Image", "Other technical drawing", "Floor plan", "Diagram"]
NUM_PROCESSES = 4
MAX_ATTEMPTS = 3

[OPERATION_4]
METHOD = CLIP_IMAGE_EMBEDDING
MODEL = clip-ViT-L-14
DECIMAL_PLACES = 8
PRECISION = bfloat16
CATEGORIES = ["Photograph", "Geometric drawing", "Graph", "Caricature and
comics", "Map", "Image", "Other technical drawing", "Floor plan", "Diagram"]
```

Tato konfigurace obsahuje několik cest k natrénovaným modelům (jejich popis je v další části) a souboru s nastavením popisování obrázků (viz níže). U sekce s popisováním obrázků pomocí ChatGPT je definován klíč `API_KEY`, do kterého je možné buď přímo vložit OpenAI API klíč, nebo cestu k textovému souboru, kde první řádek obsahuje API klíč.

Nastavení popisování elementů

Pro modul, který zajišťuje generování popisů, titulků, klíčových slov a analýzu barevnosti jednotlivých elementů pomocí ChatGPT, je možné upravovat nastavení modelu, text dotazu a maximální délku výstupu pomocí jednoduchého JSONu (v ukázkové konfiguraci jako `image_captioning_prompt.json`):

```
{
  "model": "gpt-4o-mini",
  "text": {
    "image": "Describe the following image ...",
    "photograph": "For the provided photograph, generate ...",
    "default": "Caption the attached picture using ..."
  }
  "max_tokens": 500
}
```

V nastavení popisování může být text promptu definován buď jako řetězec – v tu chvíli je pro všechny popisované kategorie text promptu stejný – nebo jako slovník, kdy klíčem je název kategorie a hodnotou je text promptu pro danou kategorii. Při definování slovníku je možné použít kategorii `default`, která se použije pro nespecifikované kategorie.

Text promptu i model mohou být jakékoliv, které nabízí OpenAI. Podmínkou funkčnosti však je, aby model vždy vygeneroval odpověď ve formátu JSON, který má tuto strukturu:

```
{
  "description_en": "string or null",
  "description_cz": "string or null",
  "caption_en": "string or null",
  "caption_cz": "string or null",
  "topics_en": "string, list of strings or null",
  "topics_cz": "string list of strings or null",
  "color_en": "string or null",
  "color_cz": "string or null"
}
```

Všechny atributy jsou ve výstupu jak v české, tak i v anglické variantě, a všechny hodnoty ve výstupu mohou být řetězec, případně `null` (prázdné), a v případě klíčových slov to může být také seznam řetězců.

Metadata

V rámci zpracování je také možné využít metadata (např. název dokumentu, rok vydání, jméno autora, atd.). Tyto informace pak mohou být využity v rámci generování výstupů pomocí ChatGPT. Tyto informace pak mohou být použity v rámci textu promptu, který funguje jako Jinja šablona. V promptu je možné definovat proměnnou (např. `{{document_name}}`), která se před zasláním na ChatGPT naplní příslušnou hodnotou. Využít je možné také pokročilejší funkcionalitu, jako jsou např. podmínky. Soubor s metadaty je pak JSON, kdy první úroveň klíčů odpovídá názvům zpracovávaných stránek a hodnotou je slovník obsahující libovolná metadata:

```
{
  "page_image_001.jpg": {
    "document_name": "Orbis Pictus",
    "document_author": "John Amos Comenius",
    "document_year": 1658,
    // ...
  },
  // ...
}
```

Výstupní data

AnnoPage může generovat množství různých výstupů (viz parametry příkazové řádky). Mezi hlavní výstupy patří výstup do ALTO XML, které obsahuje informace o umístění a obsahu jednotlivých netextových elementů, a JSON se sémantickými embeddingy.

ALTO XML

Ve výstupním ALTO XML formátu jsou uloženy informace o detekovaných netextových elementech (kategorie, umístění a rozměry), vygenerované popisy, titulky, klíčová slova barevnost elementů a případně jejich návaznost na okolní text.

V rámci elementu `PrintSpace` se netextový element zapisuje jako `ComposedBlock`, ve kterém je ještě zanořený `GraphicalElement`. Pomocí ID v atributu `TAGREFS` je pak `ComposedBlock` spojen s elementem `LayoutTag`, který se náchází v elementu `Tags`. Element `LayoutTag` pak obsahuje MODS elementy, ve kterých jsou uloženy vygenerované informace (popis, titulek, klíčová slova a barevnost), informace o samotném MODS záznamu a případně návaznost na okolní text (detekovaný titulek).

Konkrétní ukázka zápisu obrázku do ALTO XML se nachází na konci tohoto dokumentu.

JSON se sémantickými embeddingy

Hlavním účelem JSONu se sémantickými embeddingy je předání sémantického embeddingu (vektoru), který popisuje daný netextový element. K tomuto elementu jsou v JSONu uloženy metadata a samotný embedding. Metadata obsahují informace, jako jsou ID stránky, ID LayoutTag elementu v rámci ALTO XML, ID v rámci MODS, kategorie netextového elementu a informace o zpracování. Embedding je pak zde uložen jako seznam desetinných čísel. V

rámci zpracování je možné pomocí přepínače volit mezi variantou, kdy je výsledek standardní JSON, a variantou, která vytvoří JSONLines, v rámci kterého pak každý řádek představuje jeden JSON objekt.

Konkrétní ukázka JSONu se nachází na konci tohoto dokumentu.

Moduly a dostupné modely

Modul pro detekci objektů

Modul pro detekci je implementován třídou `YoloDetectionEngine`. Tento modul využívá implementaci YOLO detektoru z balíčku Ultralytics a pro jednotlivé detekované objekty vytvoří výstup ve formě regionu, který je vložen do objektu reprezentující rozložení zpracovávané stránky.

Pro detekci netextových elementů je v repozitáři zveřejněný natrénovaný model i s příslušnou konfigurací. Tento model byl trénován na AnnoPage Datasetu, který je zveřejněn na platformě Zenodo, a dosahuje zde hodnot mAP@50: 0.658 a mAP@50-95: 0.598. Kategorie, které umí detektor rozlišovat, vycházejí z Metodiky zpracování obrazových dokumentů.

Parametry konfigurace

- `MODEL_PATH`: Cesta k YOLO modelu.
- `DETECTION_THRESHOLD`: Práh jistoty modelu.
- `IMAGE_SIZE`: Velikost v pixelech, na kterou se delší strana vstupního obrázku bude normalizovat. Tato velikost by měla odpovídat tomu, v jakém rozlišení se model trénoval.
- `CATEGORIES`: Seznam kategorií, které se mají detekovat. Tento seznam funguje jako filtr nad kategoriemi, které umí model detekovat.

Moduly pro analýzu titulků netextových elementů

Analýza titulků netextových elementů se skládá ze dvou kroků: detekce titulku a přiřazení k netextovému elementu. První krok je řešen pomocí modulu pro detekci objektů z předchozí části a pro přiřazení titulku jsou implementovány tři různé moduly: přiřazení na základě vzdálenosti (třída `CaptionYoloNearestEngine`), pomocí klíčových bodů detektoru (třída `CaptionYoloKeypointsEngine`) a pomocí modelu typu Transformer (třída `CaptionYoloOrganizerEngine`).

V prvním případě je detekovaný popisek přiřazen k nejbližšímu netextovému elementu, přičemž pro výpočet vzdálenosti jsou použity středy obou objektů. Při přiřazení na základě klíčových bodů je výstupem detektoru titulků také sada bodů, které určují, ve kterých místech se nachází příslušný netextový element. Tento bod je následně opět přiřazen k nejbližšímu netextovému elementu na základě vzdálenosti od středu. Poslední možnost využívá natrénovaný model typu Transformer, který má na svém vstupu všechny netextové elementy i detekované titulky a jeho výstupem je matice, která reprezentuje relace mezi

všemi těmito objekty. Na základě hodnot v této matici jsou přiřazeny titulky k příslušným netextovým elementům.

V repozitáři byl zveřejněn model pro detekci titulků a Transformer model pro přiřazování. Pro tuto dvojici je také připravena konfigurace. Oba modely byly trénovány na prozatím nezveřejněné datové sadě, která vznikla v rámci tohoto výzkumného projektu.

Parametry konfigurace

Pro všechny varianty:

- `YOLO_PATH`: Cesta k YOLO modelu.
- `YOLO_DETECTION_THRESHOLD`: Práh jistoty YOLO modelu.
- `YOLO_IMAGE_SIZE`: Velikost v pixelech, na kterou se delší strana vstupního obrázku bude normalizovat. Tato velikost by měla odpovídat tomu, v jakém rozlišení se model trénoval.

Extra parametry pro přiřazení s Transformer modelem (`CaptionYoloOrganizerEngine`):

- `ORGANIZER_PATH`: Cesta k natrénovanému Transformer modelu.
- `ORGANIZER_CATEGORIES`: Seznam kategorií, které Transformer rozeznává. Tento seznam musí odpovídat tomu, jak byl použit při trénování modelu.

Modul pro generování textového popisu

Modul pro generování textového popisu netextových elementů je implementován ve třídě `ChatGPTImageCaptioningEngine`. Tento modul vyřízne ze stránky jednotlivé detekované netextové elementy, vytvoří text dotazu na základě šablony a metadat (viz předchozí část) a odešle jej na OpenAI API. Výsledek (JSON) je následně zpracován a uložen v rámci struktury přiřazené k danému netextovému elementu.

Parametry konfigurace

- `API_KEY`: OpenAI API klíč, nebo cesta k souboru, kde na prvním řádku je API klíč.
- `MAX_IMAGE_SIZE`: Maximální rozlišení delší strany obrázku, na kterou jsou obrázky normalizovány.
- `CATEGORIES`: Seznam kategorií, které se mají popisovat. Pokud není zadán, jsou popisovány všechny kategorie.
- `PROMPT_SETTINGS`: Cesta k JSONu, který obsahuje nastavení popisování netextových elementů.
- `NUM_PROCESSES`: Počet procesů, ve kterých popisování běží.
- `MAX_ATTEMPTS`: Maximální počet pokusů k vytvoření popisů. Může se například stát, že ChatGPT nevygeneruje odpověď ve správném výstupním formátu, takže se dotaz v takovém případě bude opakovat.

Modul pro vytváření sémantických embeddingů

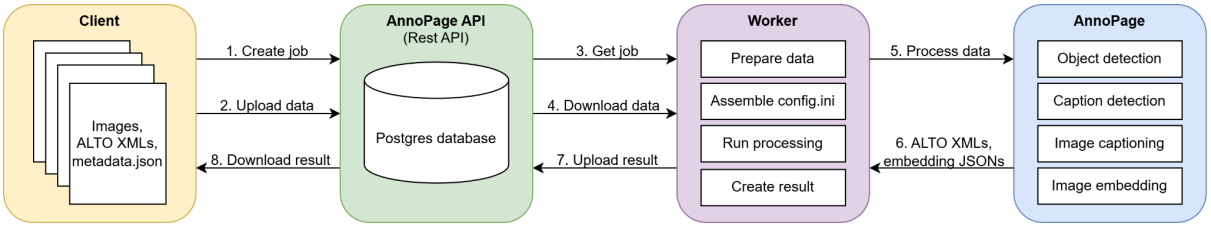
Pro vytváření sémantických embeddingů z netextových elementů je implementován modul `ClipImageEmbeddingEngine`, který využívá existující model CLIP.

Parametry konfigurace

- `MODEL`: Název CLIP modelu, který se má použít pro vytváření embeddingu. Je možné použít tyto modely: `clip-ViT-L-14`, `clip-ViT-B-16`, `clip-ViT-B-32`.
- `DECIMAL_PLACES`: Počet desetinných míst embeddingu, která jsou zapsána do výsledného JSONu.
- `PRECISION`: Parametr pro určení datového typu, s jakou přesností se embedding generuje. Tyto datové typy mohou být: `float32`, `float16`, `bfloat16`. Výchozí hodnota je `float16`.
- `CATEGORIES`: Seznam kategorií, pro které se má embedding generovat. Pokud není zadán, jsou embeddingy vytvořeny pro všechny kategorie.

AnnoPage API, klient, worker

V rámci vytváření nástroje AnnoPage vzniklo také REST API, ke kterému byl implementován i tzv. worker a jednoduchý klient. Tato část implementace umožňuje mít distribuované zpracování a rozhraní, přes které je možné vytvářet úlohy ke zpracování a následně získávat výsledky. Schéma celého systému a komunikace mezi jednotlivými částmi je na obrázku níže.



Typický pracovní postup pro zpracování stránek pomocí AnnoPage API se skládá z následujících kroků:

1. Klient vytvoří novou úlohu zpracování pomocí AnnoPage API.
2. Klient nahraje všechna potřebná data (obrázky, soubory ALTO XML, metadata a konfigurační soubor) do AnnoPage API.
3. Worker se periodicky dotazuje AnnoPage API na nové úlohy ke zpracování.
4. Jakmile obdrží úlohu, stáhne všechna požadovaná data z AnnoPage API.
5. Worker připraví konfiguraci zpracování a spustí AnnoPage nad poskytnutými daty.
6. Po dokončení zpracování shromáždí worker výsledky z AnnoPage a zabalí je do ZIP archivu.
7. Worker nahraje finální výsledek do AnnoPage API a označí úlohu jako dokončenou.
8. Klient si po dokončení úlohy stáhne výsledky z AnnoPage API.

Implementace API, worker a klienta vychází z DocAPI.

AnnoPage API

AnnoPage API je REST API, které umožňuje spravovat jednotlivé úlohy ke zpracování, spravovat uživatelské API klíče a spravovat konfigurace zpracování. Po nainstalování balíčku AnnoPage je možné spustit API příkazem `annopage_api`. Po spuštění je API veřejně

dostupné na portu `8000` (adresa `0.0.0.0:8000`). AnnoPage API se ve výchozím natavení spouští s touto konfigurací:

- název serveru: `AnnoPageAPI`
- verze aplikace: `1.0.0`
- produkční verze: `ne`
- prefix API klíčů: `annopage`
- adresář pro ukládání dat: `annopage_api_data`

Všechny parametry konfigurace se zadávají pomocí proměnných prostředí a jejich výčet i s krátkým popisem je dostupný v config.py v rámci DocAPI.

Aktuální verze API je dostupná na adrese https://annopage.orbis.lib.cas.cz.

Přidání konfigurace zpracování

Do API je možné přidávat konfigurace zpracování pomocí webového rozhraní. Nejprve je potřeba se přihlásit jako administrátor (tlačítko `Authorize` nahoře na stránce a do některé z možností zadat administrátorský API klíč). Dalším krokem je vytvoření nové konfigurace (metoda `POST /v1/admin/engines`) a jako poslední krok je potřeba nahrát ZIP s potřebnými soubory (konfigurace zpracování, natrénované modely, atd.; metoda `PUT /v1/admin/engines/{name}/{version}/files`). Podmínkou pro tento ZIP je, aby v kořenové úrovni obsahoval konfigurační soubor pojmenovaný `config.ini`.

Worker

Worker získává úlohy ke zpracování z AnnoPage API a provádí zpracování stránek pomocí AnnoPage. Worker se pravidelně dotazuje na novou úlohu a jakmile je nová úloha k dispozici, stáhne si všechna potřebná data (obrázky, soubory ALTO XML, metadata a konfiguraci), připraví konfiguraci zpracování a spustí nástroj AnnoPage. Po zpracování worker vytvoří ZIP archiv s výsledky, nahraje jej do AnnoPage API a označí úlohu jako dokončenou.

Workera je možné spustit pomocí příkazu `annopage_worker`:

```
annopage_worker \
  --api-url=API_URL \
  --api-key=API_KEY
```

Minimálními parametry spuštění worker jsou URL, na kterém je dostupné API, a API klíč, který je přiřazený workerovi. Další parametry spuštění umožňují nastavit adresáře, kam se stahují natrénované modely a data ke zpracování, nastavení intervalu dotazování na API pro novou úlohu, přepínače pro odstraňování nepotřebných dat, či nastavení logování. Veškeré informace o těchto parametrech je možné získat příkazem `annopage_worker --help`, který vypíše nápovědu programu.

Klient

Implementace klienta slouží jako ukázka toho, jak je možné komunikovat s AnnoPage API pro zpracování dokumentů, tak i pro uživatele, kteří si chtějí nechat zpracovat vlastní data. Klient se spouští příkazem annopage_client a v současné době nabízí dvě funkcionality: získat informace o konfiguracích zpracování AnnoPage API a vytvářet úlohy ke zpracování.

Získání informací o konfiguracích zpracování

Pro získání informací o všech poskytovaných konfiguracích zpracování je možné použít následující příkaz:

```
annopage_client \
  --list-engines \
  --api-url=API_URL
  --api-key=API_KEY
```

Tento příkaz získá z AnnoPage API všechny dostupné konfigurace a vypíše je:

```
Available engines:

Name: 'Engine A'
Description:
The primary AnnoPage engine which consists of the following parts:
1. Detection of non-textual elements
2. Detection and assignment of captions on the page
3. Description of images using ChatGPT
4. Creation of embeddings using the CLIP model
```

Název konfigurace zpracování (v ukázce `Engine A`) je pak možné použít při vytváření úlohy ke zpracování.

Vytváření úlohy ke zpracování

V případě vytváření úlohy ke zpracování se klientovi předávají cesty ke vstupním datům, přepínače specifikující, jaké výstupy jsou požadovány, a nastavení URL a API klíče:

```
annopage_client \
  --images=input/images \
  --alto-xmls=input/alto \
  --output=output \
  --output-alto \
  --api-url=API_URL
  --api-key=API_KEY
```

Textový výstup klienta pak může vypadat nějak takto:

```
Job fd020269-e08e-43ac-b0f5-42cfa605214c created
```

```
Created: 2025-12-06 18:06:06.000000+00:00
Engine: name=Engine A, version=v1.0.0
Images: 1
Requirements: ALTO=True, PAGE=False, Meta=False

Waiting for job to complete...
Job status: ProcessingState.QUEUED, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job status: ProcessingState.PROCESSING, progress: 0.0
Job fd020269-e08e-43ac-b0f5-42cfa605214c completed successfully
```

Pokud by byl běh klienta přerušen kdykoliv před dokončením zpracování a stažením výsledků, bude úloha označena za zrušenou.

Pomocí parametrů spuštění klienta je možné definovat vstupní data (obrázky, ALTO XML nebo PAGE XML, metadata), konfiguraci popisování netextových elementů, zvolenou konfiguraci zpracování, požadované výstupy zpracování, či nastavení logování. Seznam všech parametrů i s popisem je možné získat pomocí příkazu `annopage_client --help`.

Ukázkové výstupy AnnoPage

ALTO XML

```
<?xml version='1.0' encoding='utf-8'?>
<alto xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:mods="http://www.loc.gov/mods/v3"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.loc.gov/standards/alto/ns-v4#">
  ...
  <Tags>
    <Tags>
    <LayoutTag ID="photograph_001" TYPE="Structural" LABEL="Photograph">
      <XmlData>
        <mods:mods ID="MODS_PICT_0001">
```

```
          <mods:titleInfo altRepGroup="title-1">
            <mods:title>Obr. 9. Barokní okno v jižní straně hlavní
lodi.</mods:title>
          </mods:titleInfo>
          <mods:typeOfResource>still image</mods:typeOfResource>
          <mods:genre altRepGroup="genre-1"
lang="eng">photograph</mods:genre>
          <mods:genre altRepGroup="genre-1"
lang="cze">fotografie</mods:genre>
          <mods:physicalDescription>
            <mods:extent unit="pixels">714x855</mods:extent>
          </mods:physicalDescription>
          <mods:physicalDescription altRepGroup="color-1">
            <mods:form type="color" lang="eng">grayscale</mods:form>
          </mods:physicalDescription>
          <mods:physicalDescription altRepGroup="color-1">
            <mods:form type="color" lang="cze">šedotónový</mods:form>
          </mods:physicalDescription>
          <mods:abstract altRepGroup="caption-1" type="caption"
lang="eng">Baroque window on the southern side.</mods:abstract>
          <mods:abstract altRepGroup="caption-1" type="caption"
lang="cze">Barokní okno na jižní straně.</mods:abstract>
          <mods:abstract altRepGroup="description-1" type="description"
lang="eng">The photograph displays a baroque window located on the southern
side of the main nave. The window features intricate stonework surrounding a
grid pattern of glass panes.</mods:abstract>
          <mods:abstract altRepGroup="description-1" type="description"
lang="cze">Fotografie zobrazuje barokní okno nacházející se na jižní straně
hlavní lodi. Okno má složité kamenné obložení kolem mřížkovaného vzoru
skleněných tabulí.</mods:abstract>
          <mods:subject altRepGroup="subject-1">
            <mods:topic lang="eng">baroque</mods:topic>
          </mods:subject>
          <mods:subject altRepGroup="subject-2">
            <mods:topic lang="eng">architecture</mods:topic>
          </mods:subject>
          <mods:subject altRepGroup="subject-3">
            <mods:topic lang="eng">window</mods:topic>
          </mods:subject>
          <mods:subject altRepGroup="subject-4">
            <mods:topic lang="eng">church</mods:topic>
          </mods:subject>
          <mods:subject altRepGroup="subject-1">
            <mods:topic lang="cze">baroko</mods:topic>
          </mods:subject>
          <mods:subject altRepGroup="subject-2">
            <mods:topic lang="cze">architektura</mods:topic>
          </mods:subject>
          <mods:subject altRepGroup="subject-3">
            <mods:topic lang="cze">okno</mods:topic>
          </mods:subject>
          <mods:subject altRepGroup="subject-4">
            <mods:topic lang="cze">kostel</mods:topic>
```

```
          </mods:subject>
          <mods:identifier
type="uuid">uuid:9a952729-3891-4962-8445-9a928debfb96</mods:identifier>
          <mods:relatedItem type="constituent"
IDREF="MODS_PICT_0001_CAPTION_0001"/>
          <mods:recordInfo>

<mods:recordCreationDate>2025-12-08T19:35:57</mods:recordCreationDate>
            <mods:recordContentSource>AnnoPage
v0.0.1</mods:recordContentSource>
            <mods:descriptionStandard>StandardNDK</mods:descriptionStandard>
            <mods:recordOrigin>machine-generated</mods:recordOrigin>
            <mods:recordInfoNote
type="confidence">0.909</mods:recordInfoNote>
            <mods:recordIdentifier
source="AnnoPage">uuid:f02c4701-df70-44ef-91fa-e62c1b486351</mods:recordIdent
ifier>
            <mods:languageOfCataloging>
              <mods:languageTerm
authority="iso639-2b">eng</mods:languageTerm>
              <mods:languageTerm
authority="iso639-2b">cze</mods:languageTerm>
            </mods:languageOfCataloging>
          </mods:recordInfo>
        </mods:mods>
      </XmlData>
    </LayoutTag>
    <StructureTag ID="fc.photograph_001" TYPE="Functional"
LABEL="FigureCaption" DESCRIPTION="Obr. 9. Barokní okno v jižní straně hlavní
lodi.">
      <XmlData>
        <mods:mods ID="MODS_PICT_0001_CAPTION_0001">
          <mods:titleInfo altRepGroup="title-1">
            <mods:title>Obr. 9. Barokní okno v jižní straně hlavní
lodi.</mods:title>
          </mods:titleInfo>
          <mods:typeOfResource>text</mods:typeOfResource>
          <mods:genre altRepGroup="genre-1" type="part">caption</mods:genre>
          <mods:genre altRepGroup="genre-1" lang="eng"
type="part">caption</mods:genre>
          <mods:genre altRepGroup="genre-1" lang="cze"
type="part">popis</mods:genre>
          <mods:identifier
type="uuid">uuid:edc32fb4-6d59-4c90-a8b1-f4e8b15bbf17</mods:identifier>
          <mods:relatedItem type="host" IDREF="MODS_PICT_0001"/>
          <mods:recordInfo>

<mods:recordCreationDate>2025-12-08T19:35:57</mods:recordCreationDate>
            <mods:recordContentSource>AnnoPage
v0.0.1</mods:recordContentSource>
            <mods:descriptionStandard>StandardNDK</mods:descriptionStandard>
            <mods:recordOrigin>machine-generated</mods:recordOrigin>
```

```
            <mods:recordInfoNote
type="confidence">0.909</mods:recordInfoNote>
            <mods:recordIdentifier
source="AnnoPage">uuid:d83f104f-e88e-42f6-8a1d-2d8cf2a3b46f</mods:recordIdent
ifier>
            <mods:languageOfCataloging>
              <mods:languageTerm
authority="iso639-2b">eng</mods:languageTerm>
              <mods:languageTerm
authority="iso639-2b">cze</mods:languageTerm>
            </mods:languageOfCataloging>
          </mods:recordInfo>
        </mods:mods>
      </XmlData>
    </StructureTag>
    ...
  </Tags>
  <Layout>
    <Page ID="id_9fd66fdf-7fb3-4982-97a7-f0a72ce3aa8c" PHYSICAL_IMG_NR="1"
HEIGHT="2869" WIDTH="1825">
      <TopMargin HEIGHT="285" WIDTH="1825" VPOS="0" HPOS="0"/>
      <LeftMargin HEIGHT="2869" WIDTH="265" VPOS="0" HPOS="0"/>
      <RightMargin HEIGHT="2869" WIDTH="0" VPOS="0" HPOS="1825"/>
      <BottomMargin HEIGHT="0" WIDTH="1825" VPOS="2869" HPOS="0"/>
      <PrintSpace HEIGHT="2583" WIDTH="1559" VPOS="285" HPOS="265">
        ...
        <TextBlock ID="block_r001" HEIGHT="33" WIDTH="638" VPOS="1902"
HPOS="859">
          <TextLine ID="line_r001-l036" BASELINE="859,1926 1143,1926
1498,1926" VPOS="1902" HPOS="859" HEIGHT="33" WIDTH="638"
TAGREFS="fc.photograph_001">
            <String CONTENT="Obr." HEIGHT="33" WIDTH="48" VPOS="1902"
HPOS="867" WC="1"/>
            <SP WIDTH="4" VPOS="1902" HPOS="915"/>
            <String CONTENT="9." HEIGHT="33" WIDTH="16" VPOS="1902"
HPOS="932" WC="1"/>
            ...
        </TextBlock>
        ...
        <ComposedBlock ID="id_9fd66fdf-7fb3-4982-97a7-f0a72ce3aa8c_CB0001"
TYPE="Photograph" HEIGHT="855" WIDTH="714" VPOS="992" HPOS="853"
TAGREFS="photograph_001">
          <GraphicalElement
ID="id_9fd66fdf-7fb3-4982-97a7-f0a72ce3aa8c_GE0001" HEIGHT="855" WIDTH="714"
VPOS="992" HPOS="853"/>
        </ComposedBlock>
        ...
      </PrintSpace>
    </Page>
  </Layout>
</alto>
```

JSON s embeddingy

```
[
  {
    "id": "uuid:9a952729-3891-4962-8445-9a928debfb96",
    "tag_id": "photograph_001",
    "page_uuid": "9fd66fdf-7fb3-4982-97a7-f0a72ce3aa8c",
    "category": "photograph",
    "source": null,
    "processing_info": {
      "system": "AnnoPage",
      "version": "0.0.1",
      "datetime": "2025-12-08T19:35:57.287710",
      "model": "clip-ViT-L-14",
      "decimal_places": 8,
      "precision": "bfloat16"
    },
    "embedding": [
      0.36328125,
      0.99609375,
      -0.24804688,
      1.640625,
      -0.01843262,
      -0.12255859,
      -0.1875,
      -0.04882812,
      0.16992188,
      ...
    ]
  },
  ...
]
```

V případě použití příslušného přepínače je možné výstup s embeddingy uložit i jako JSONLines, kde každý řádek souboru je jeden JSON objekt:

```
{"id": "uuid:9a952729-3891-4962-8445-9a928debfb96", "tag_id": ... }
...
```