# **Project Plan: Full Text Screening - AI model**

Timeline in the method project:

Training sets 2 & 3 ready in December

Full abstract screening done around January?

Timeline of AI full text screening

What is the desired schedule, just so I know what to prioritize as I have another project as well.

The currently relevant topics are highlighted with orange.

## **Scope**

The goal is to assist screeners with inclusion/exclusion decisions based on full data.   
The project can be divided to two sub tasks:

1. Tagging the data
2. Use tagged data to exclude/include, automatically or with human in the loop.

Tagging can mean either word counts from pre-defined dictionary, or question answering/data answering with NLP. I will first focus on the possibilities with NLP.

**Input:**  Full pdf

**Input format for the model**: As an image or as a text?

How big proportion of the papers are image like pdfs where some off-shelf OCR should be applied? This can cause additional errors in the predictions. And if we take text like input, this means that Figures and Tables are not included. This might also effect on how computationally expensive the model is. Is this a restriction for us? How much GPU time do we have?

**Desired output:**   
Tagging information with NLP  
PICO:

* Subject: [human, animal, in-vivo]
  + If animal, what species
  + If in-vivo, what cells
* Intervention / Exposure:
  + Radiation: [gamma, x, alpha,…]
  + Other interventions?
* Comparison
* Outcome

Decision Tree:

* Is literature review: y/n
* Includes ionizing radiation: y/n
* Only UV: y/n
* …

One possible approach is also to firs use some separate classifier to detect human/animal and in-vivo studies, and based on that apply the corresponding model for tagging.

**Output format:** JSON? .txt? .csv? and the structure of these outputs?

## **Data collection**

First collect test set by hand.

150 screened papers (abstract screening done manually)  
- 50 test  
- 50 validation  
- 50 train  
But only includes the URL, no the pdf. Is this URL the same as in the big dataset?

R Code provided by Edinburgh, run where and when?

## **DaTA Preprocessing**

Use OCR if image-mased PDF,  
or extract raw text with tool like Lang chain.  
Then remove references.

* Cropit tool? This was mentioned in the meeting but I couldn’t find it, maybe I misheard the name?

Possible workflow:

1. Pdfminer: find text, table and figure boxes (The order of the boxes is not perfect.)
2. PyPDF2 (or OCR if image-based): extract data from the text boxes
3. Use PyPDF2 font information to only include the main text (**assuming** main text has the most common font type)

## **Data Screening?**

We need at least some data for the test set. How will this be addressed?

Is the exclusion inclusion criteria the same as in the abstract screening?

## **Possible methods**

Pre-Pico   
Abstract text 🡪 PICO labels  
 [Species, Intervention, Comparator, Strain, Induction, Outcome]  
([hemmlin/pre-pico-test (github.com)](https://github.com/hemmlin/pre-pico-test))  
From my initial exploration on the code it seems like data extraction from abstract  
Waiting for further Documentation  
BIRD based solutions might not be the best approach for long context: [2203.11258.pdf (arxiv.org)](https://arxiv.org/pdf/2203.11258.pdf)

Donut 🍩 Image 🡪 question answering, or data extraction to json[arxiv.org/pdf/2111.15664.pdf](https://arxiv.org/pdf/2111.15664.pdf)Nice open source pre trained model with easy api through HuggingFaceAlready working somehowCould be limited in the size of context?Similar approach to Microsoft DUBLIN [2305.14218.pdf (arxiv.org)](https://arxiv.org/pdf/2305.14218.pdf)

General purpose VDUse.g. LayoutLM, LayoutLMv2, BERT  
OCR 🡪 Transformer 🡪 Encoder

Spade  
[Spatial Dependency Parsing for Semi-Structured Document Information Extraction (aclanthology.org)](https://aclanthology.org/2021.findings-acl.28.pdf)similar to WYVERN  
graph based method that utilizes the OCR bounding boxes to obtain better performance  
Maybe not so useful with articles where the hierarchy of the text is quite simple.

PromptingText context 🡪 question answering.  
[lamini-ai/llm-classifier: Classify data instantly using an LLM (github.com)](https://github.com/lamini-ai/llm-classifier)  
- both zero- and few-shot methods available  
- requires docker 🡪 not possible to test in my work laptop.[2303.05352.pdf (arxiv.org)](https://arxiv.org/pdf/2303.05352.pdf)[2305.19835.pdf (arxiv.org)](https://arxiv.org/pdf/2305.19835.pdf)Would require paid version of ChatGPT API, but probably easy to implement and can be used for highlighting. Fully zero-shot method.Or using Llama or other open-source option.ChatGPT API pricing: [Pricing (openai.com)](https://openai.com/pricing)  
0.003$/1K tokens  
🡪 approx. 0.03$/paper + answer tokens.

Classifiers for long context  
[applsci-12-04554-v2.pdf](file:///C:\Users\linda.hemmann\Downloads\applsci-12-04554-v2.pdf)  
[2203.11258.pdf (arxiv.org)](https://arxiv.org/pdf/2203.11258.pdf)  
- bench mark  
- truncated BERT is quite good 🡪 is it wise to do this with AI?  
- should we just reuse the abstract classifier with random or ranked sentences from the paper?  
[Building a classification service with Llama2 in Python (mikulskibartosz.name)](https://mikulskibartosz.name/building-classification-service-with-llama2-in-python)  
[sh0416/llama-classification: Text classification with Foundation Language Model LLaMA (github.com)](https://github.com/sh0416/llama-classification/tree/master)  
-Llama2 based classifiers seem to be quite popular  
- unfortunately, the maximum context size is **2048 tokens**

### Other papers:

**LLM for interpreting tables:**[2305.14336.pdf (arxiv.org)](https://arxiv.org/pdf/2305.14336.pdf)  
[2001.01469.pdf (arxiv.org)](https://arxiv.org/pdf/2001.01469.pdf)

**OCR libraries:**Pytesseract: [pytesseract · PyPI](https://pypi.org/project/pytesseract/)

easyOCR: [JaidedAI/EasyOCR: Ready-to-use OCR with 80+ supported languages and all popular writing scripts including Latin, Chinese, Arabic, Devanagari, Cyrillic and etc. (github.com)](https://github.com/JaidedAI/EasyOCR)

PyPDF: [pypdf · PyPI](https://pypi.org/project/pypdf/)

LangChain: [Introduction | 🦜️🔗 Langchain](https://python.langchain.com/docs/get_started/introduction.html)  
- works both with image and text pdfs

trOCR: [TrOCR (huggingface.co)](https://huggingface.co/docs/transformers/model_doc/trocr)

docTR: [docTR documentation (mindee.github.io)](https://mindee.github.io/doctr/)  
- Gives the spatial information and the text, can be used to recreate the document for highlighting etc.

PaddleOCR: [paddleocr · PyPI](https://pypi.org/project/paddleocr/)  
- gives bounding box, for highlighting  
- English and Chinese

Clova OCR   
- commercial

## **Results**

Clearly define the desired evaluation statistics and benchmarking metrics based on the project's objectives. What will be considered a successful outcome for this project?

## **Documentation and Resources**

Github Project: [ESA-RadLab/FullTextTagging (github.com)](https://github.com/ESA-RadLab/FullTextTagging)  
Front end?

## **Ethical Considerations**:

Is it okay to use ChatGPT?

For whom, can we share the files that were accessed under university license?

## **Glossary**

VDU – Visual Document Understanding  
OCR – Optical Character Recognition  
IE – Information Extraction  
DocVQA – Document Visual Question Answering

## **Other notes**

The performance of general-purpose VDUs relies on the performance of OCR (Donut)

A graph of different colored bars

Description automatically generated with medium confidence