There are four main components that I've developed. They cover the same basic concept of classifying and/or scraping FUS in ML related articles and classifying them based on fus/non-fus related, machine learning type, medical indication, treatment cycle, and keywords.

- app.py: This is a visualizer that can be opened either locally by running the code. It is also constantly running at https://huggingface.co/spaces/Gators123/fusf_pdf_2023. A single PDF file can be inputted and the program displays the classifications on the website. A chatbot is also available to answer any other questions about the paper.

- directory_parser.py: Intakes a directory and locates all the .pdf files. Then runs the classifications through each one and returns the results in an Excel sheet.

- scrape_code.py: Scrapes PubMed starting from a certain date specified by the user and returns the resulting papers in an Excel sheet, including basic information such as Title, Author, Publication Date, and Abstract.

- scrape+ml+final.py: Basically identical to scrape_code.py except additionally returns every one of the FUS/ML classifications.

*Note: all programs except scrape_code.py will require the user to enter an API key, accessible via the OpenAI website, https://openai.com/

Next Steps

- Currently, due to the restraints of the API, only the first 4000 characters can be passed into the classification models. This pretty consistently captures the abstract, but in the future, it would be beneficial to split up papers into different sections and have different weights based on each section.
- The existing scraping methods are confined to PubMed articles, neglecting potentially relevant articles from other journals. Consequently, developing analogous programs to extract data from other centralized databases would accelerate the rate of finding new articles.