Zackary Morelli

1/14/2022

[zackmorelli@gmail.com](mailto:zackmorelli@gmail.com)

Tesseract Execute is a very small console program that uses the open-source Tesseract library to read text from images. It is an optical character recognition (OCR) program that uses machine-learning datasets to interpret characters from various languages. The trained data file for English is in the tessdata folder. The Tesseract library used here is a specific C# wrapper of the original C++ library that I got from Github.

The purpose of this program is simply to be called by the PatientScansOCRBackgroundWorker windows service that I made. You can look in that program’s folder for more information, but it is a program that actually runs as a windows service in the background. It monitors the Aria documents folder for any changes that are made for when a new scanned patient document is added, as a PDF. When this happens, the PatientScansOCRBackgroundWorker windows service starts the Tesseract Execute program as a separate process.

Tesseract Execute then reads in the file and saves the output from Tesseract in a specific location on the Aria file server. The Plancheck program then checks for these files when it is run as part of its document check module.

The reason why Tesseract is run separately from the Plancheck Eclipse script itself, is that, because it is a C# wrapper for a C++ program, it requires .NET C++ runtime files for it to run. I was not able to install these on the Aria file server so it could run as part of Plancheck, so I came up with this as a workaround. The PatientScansOCRBackgroundWorker windows service would be running on my computer, and by running it as a windows service under my user account, instead of the OS, it was able to monitor the Aria file server network location that I have access to with my credentials. When it detected a new PDF written to the folder (which is always a scan of a patient document, like a HIPPA Consent), it would run Tesseract Execute on my computer as a separate process. It was easier to do this than include Tesseract in the windows service, because a windows service is very different than a normal executable program. Tesseract Execute would run fine on my computer because I had installed the required .NET C++ runtime files (which the author of the C# wrapper library, Charles Weld, explains how to do) in the Visual Studio environment on computer.

Believe it or not, I had this crazy setup working on my computer for around a month. These programs are not very useful anymore because we use Radformation’s scripts for this kind of thing, but they are here for those who are interested.