Article
Yuri Marx · Nov 19, 2020 6m read

Open Exchange

OCR and NLP together into InterSystems IRIS

According to IDC, more than 80% of information it is NoSQL, especially text into documents. When the digital services or applications not process all this information, the business lose. To face this challenge, it is possible use OCR technology. OCR uses machine learning and/or trained image patterns to transform image pixels into text. This is important, because many documents are scanned into images inside PDF, or many documents contains images with text inside. So OCR are an important step to get all possible data from a document.

To do OCR, the main open source solution used is Google Tesseract, the most popular solution into the Python and Java community. Tesseract has support to more than 100 idioms and can be trained with new models to recognize car plates, captchas and so on. Tesseract was created in C++, so Java uses it consuming an intermediate, called Tess4J. My following code shows it to you:

```
private String extractTextFromImage(File tempFile) throws TesseractException {
        ITesseract tesseract = new Tesseract();
       tesseract.setDatapath("/usr/share/tessdata/"); //directory to trained models
       tesseract.setLanguage("eng+por"); // choose your language/trained model
        return tesseract.doOCR(tempFile); //call tesseract function doOCR()
                                //passing the file to be processed with OCR technique
    }
```

To allows IRIS to use this Java Class and get the results from Java, we need to use PEX and Java Gateway solutions.

First it is necessary config Java Proxy into the production and second, config a PEX business operation or service to communicate IRIS and Java into a production.

```
Class dc.ocr.OcrProduction Extends Ens.Production
{
 XData ProductionDefinition
{
<Production Name="dc.ocr.OcrProduction" LogGeneralTraceEvents="false">
 <Description></Description>
 <ActorPoolSize>2</ActorPoolSize>
 <Item Name="OcrService" Category="" ClassName=
"dc.ocr.OcrService" PoolSize="1" Enabled="true"
Foreground="false" Comment="" LogTraceEvents="false" Schedule="">
 </Item>
 <Item Name="JavaGateway" Category="" ClassName=
"EnsLib.JavaGateway.Service" PoolSize="1"
Enabled="true" Foreground="false" Comment="" LogTraceEvents="false"
Schedule="">
```

```
    <Setting Target="Host" Name="ClassPath">
.:/usr/irissys/dev/java/lib/JDK18/*:/opt/irisapp/*
:/usr/irissys/dev/java/lib/gson/*
:/usr/irissys/dev/java/lib/jackson/*:/jgw/ocr-pex-1.0.0.jar
</Setting>
    <Setting Target="Host" Name="JavaHome">
/usr/lib/jvm/java-8-openjdk-amd64/</Setting>
  </Item>
  <Item Name="OcrOperation" Category="" ClassName=
"EnsLib.PEX.BusinessOperation" PoolSize="1"
Enabled="true" Foreground="false" Comment="" LogTraceEvents="false"
Schedule="">
    <Setting Target="Host" Name="%gatewayPort">55555</Setting>
    <Setting Target="Host" Name="%remoteClassname">
community.intersystems.pex.ocr.OcrOperation</Setting>
    <Setting Target="Host" Name=
"%gatewayExtraClasspaths">.:/usr/irissys/dev/java/lib/JDK18/*
:/opt/irisapp/*:/usr/irissys/dev/java/lib/gson/*
:/usr/irissys/dev/java/lib/jackson/*
:/jgw/ocr-pex-1.0.0.jar
</Setting>
  </Item>
</Production>
}
  }
```

Now any IRIS production can communicate with Java and Tesseract! See:

```
 //call ocr method to get text from image, if you want to use pex
        Set pRequest = ##class(dc.ocr.OcrRequest).%New()
        Set pRequest.FileName = file.Filename


// call java pex operation to do ocr, passing file into pRequest and receive ocr text
with pResponse
        Set tSC = ..SendRequestSync("OcrOperation", pRequest, .
pResponse, 1200)

        //save the results into database to use text analytics - nlp
        Set ocrTable = ##class(dc.ocr.OcrTable).%New()
        Set ocrTable.FileName = file.Filename
```

```
        Set ocrTable.OcrText = pResponse.StringValue
    Set tSC = ocrTable.%Save()
```

All code details, with comments can be found into my OCR Service repository (https://openexchange.intersystems.com/package/OCR-Service).

Now, with the text extracted, we need to use IRIS NLP engine to analyze textual data and get insights to support decisions. For this, when a text is extracted, it is saved into a table, and this table is used by NLP engine as text source. See the table %Save() above and see the following code with NLP referencing OCRTable (place with texts extracted). See:

```
Class dc.ocr.OcrNLP Extends %iKnow.DomainDefinition [ ProcedureBlock ]
{
  XData Domain [ XMLNamespace = "http://www.intersystems.com/iknow" ]
{
<domain name="OcrNLP" disabled="false" allowCustomUpdates="true">
<parameter name="DefaultConfig" value="OcrNLP.Configuration" isList="false" />
<data dropBeforeBuild="true">
<table listname="OcrNLPTable" batchMode="true" disabled="false"
listerClass="%iKnow.Source.SQL.Lister" tableName=
"dcocr.OcrTable"   idField="ID"
groupField="ID" dataFields="OcrText" metadataColumns="FileName"
metadataFields="filename" />
</data>
<matching disabled="false" dropBeforeBuild="true" autoExecute="true"
ignoreDictionaryErrors="true" />
<metadata>
<field name="filename" operators="=" dataType="STRING" storage="0"
caseSensitive="false" disabled="false" />
</metadata>
<configuration name="OcrNLP.Configuration" detectLanguage="true"
languages="en,pt"
userDictionary="OcrNLP.Dictionary#1" summarize="true"
maxConceptLength="0" />
<userDictionary name="OcrNLP.Dictionary#1" />
</domain>
}
  }
```

See full details and configuration into my OCR Service github repository.

Now we can upload some files and go to the Explorer to see concepts and CRC generated.

See my animation with all steps discussed here:

Happy OCR/NLP hacking!

#Analytics #Interoperability #Java #InterSystems IRIS
Check the related application on InterSystems Open Exchange

Source URL:https://community.intersystems.com/post/ocr-and-nlp-together-intersystems-iris