Solution Approach

Word2vec with AWS SageMaker BlazingText

Raghuveer Varahagiri

September 11, 2018

The Approach

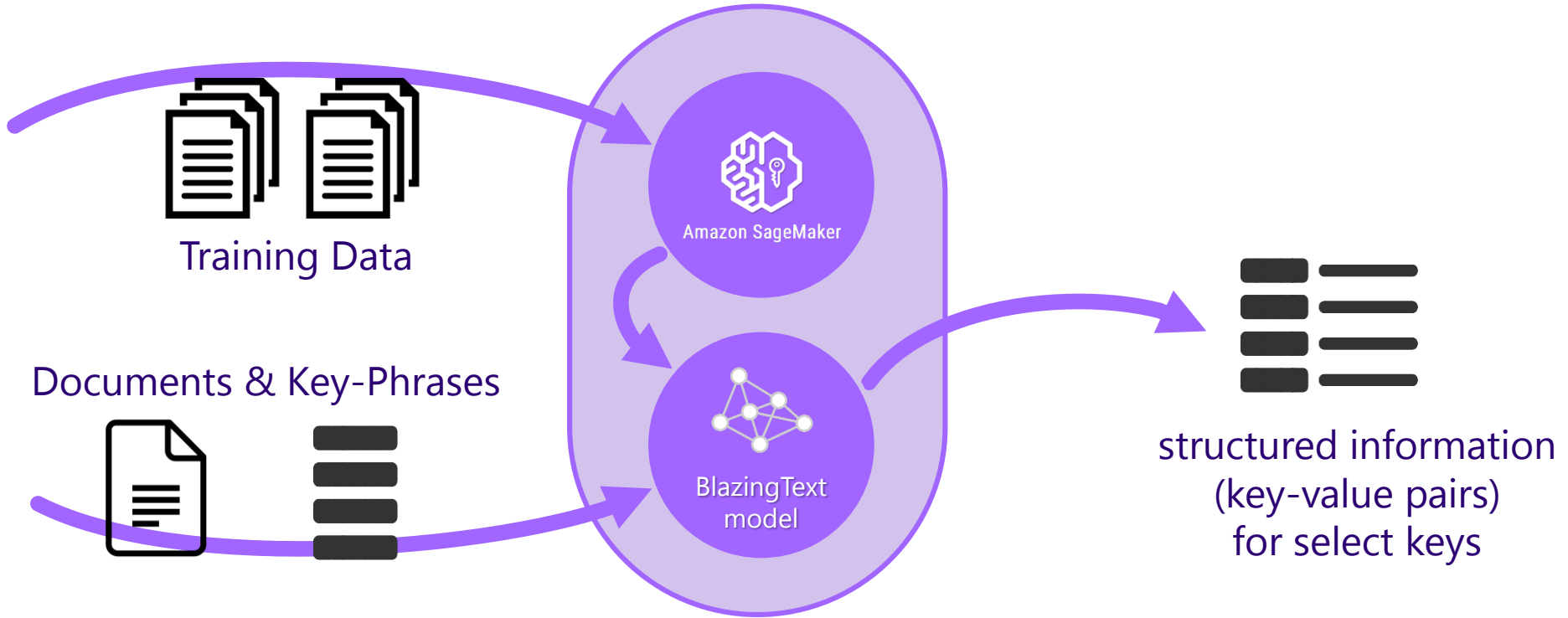
1. Defining the problem & understanding the input/output
2. Decomposing the problem into components
3. Selecting the right tools for each part of the problem
4. Implement the solution for each part and validate
5. Integrate all the pieces to make them work together
6. Refine the solution to optimize performance and cost

1. Problem Definition, Input & Output

THE CHALLENGE

Extraction of structured information from documents

- (1) Build, train and deploy a NLP (Natural Language Processing) model using AWS SageMaker machine learning algorithms
- (2) Use this model to process transactional documents from the given business domain and automatically extract the requested information structured as Key-Value pairs



1. Problem Definition, Input & Output

INPUT

- Documents are from a specific business domain and contain information in various sections and tables. They come in either DOCX or PDF format. This information is to be treated as raw Key-Value pairs.
- "**Keys**" in this case are descriptive phrases from documents – typically in the form of section headings or first-column entries (akin to form fields or labels) in a table.
- Even though all documents belong to the same business domain, they are not expected to have identical structures (hence the need for NLP). The keys in each document could be phrased slightly differently, and every document need not have the same set of keys either.
- The "**Values**" are the contents under those headings or form fields. Each document is expected to have varied values, and could be unique to that document (including proper nouns like company names that occur only once in the transactional document – and never in the training data).

1. Problem Definition, Input & Output

OUTPUT

- Given one or more transactional documents and one or more requested key-phrases as input, the output is expected as follows:

	Document #1	Document #2
requested key-phrase	***value***	***value***
Name	Henry O'Brien Plastering Contractors Ltd T/A VIB Group	J & J Ormington PLC and subsidiary companies
Employees paid below threshold	No	No
Trade of Client	Plastering Contractor	Importers, Retailers & Manufacturers of Kitchen, Bathroom & Bedroom furniture, White Good Suppliers, Joinery, Building Work

2. Problem Decomposition

- **(1) Build, train and deploy a NLP (Natural Language Processing) model using AWS SageMaker machine learning algorithms**
  - **Select an algorithm fit for the purpose (BlazingText)**
  - **Identify training data sets to use – general and domain specific**
  - **Launch an AWS SageMaker notebook instance**
  - **Prepare and load the training data (to S3)**
  - **Train the BlazingText model to generate word vectors (using corresponding sagemaker python libraries)**
  - **Deploy the model**
  - **Evaluate results and fine-tune the hyperparameters**
  - **Create a routine that accepts two phrases as input and returns a score of similarity between them using the above model**
- (2) Use this model to process transactional documents from the given business domain and automatically extract the requested information structured as Key-Value pairs

2. Problem Decomposition

- (1) Build, train and deploy a NLP (Natural Language Processing) model using AWS SageMaker machine learning algorithms
- **(2) Use this model to process transactional documents from the given business domain and automatically extract the requested information structured as Key-Value pairs**
  - **Pre-process the input documents: use a python SDK to read DOCX and PDF documents and extract their structure and contents as raw key-value pairs**
  - **For each requested "key-phrase" provided separately as input, use the similarity score to identify the most similar raw key present in the document. (A minimum threshold for similarity may be defined below which the requested key-phrase is deemed missing from the input document)**
  - **Return the corresponding raw "value" from the document for each requested "key-phrase"**

Next Steps & Considerations

1. Defining the problem & understanding the input/output
2. Decomposing the problem into components
   - Validate the understanding and approach with SMEs
3. Selecting the right tools for each part of the problem
   - Research how to extend the use of Word2Vec / BlazingText algorithms to apply to **phrases** (containing multiple words each)
   - Research standard ways to measure similarity of words and phrases (Cosine Similarity / Spearman's rank correlation)
   - Research and pick a python SDK that can read and extract Tables and Sections structure from DOCX and PDF files
4. Implement the solution for each part and validate
   - For evaluating the BlazingText model, manually create and use a JSON / Text input as a proxy for pre-processed raw key-value pairs from actual documents
5. Integrate all the pieces to make them work together
   - Consider automating the steps so that end users can get the desired results without having to use a Jupyter notebook / expose this as a service endpoint that other applications can access
6. Refine the solution to optimize performance and cost
   - Measure the performance of the solution and the cost incurred in training and using the model – look for opportunities to improve
   - The solution could be generalized and used in applications in multiple business domains