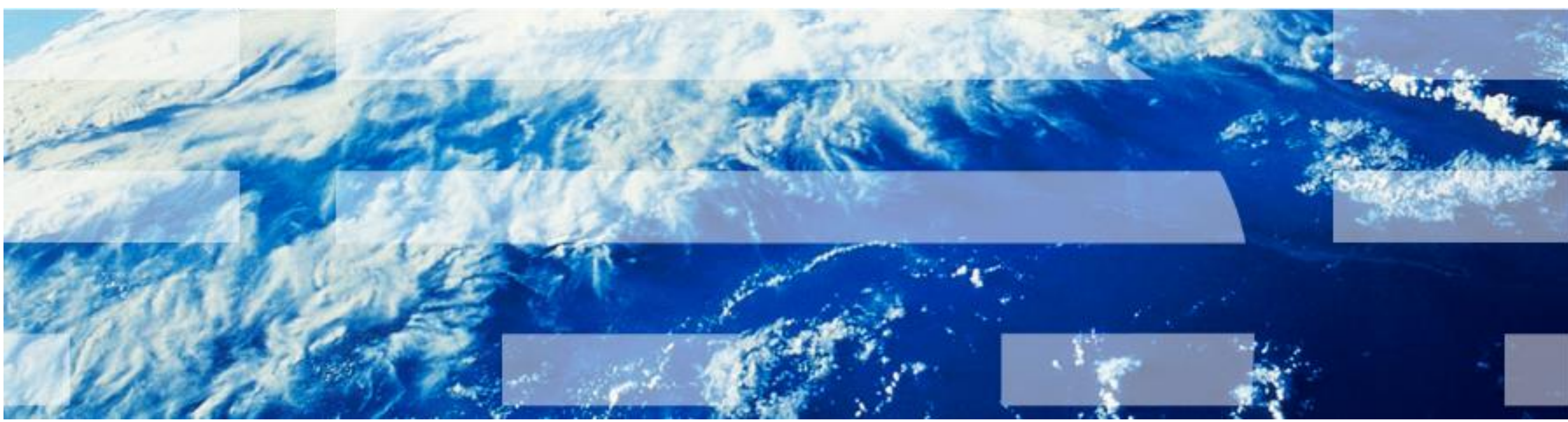
E6893 Big Data Analytics:

Speech Analytics Software Library

Team Members: Kyle White



December 9, 2014

Motivation

- A need exists in the speech processing and recognition community to adapt or create tools for coping with increasing amounts of available data.

- Example: A speech recognition project can require on the order of 50GB or more of training audio data for acoustic modeling to realize gains for popular approaches such as decoding speech via neural network classifiers. Transcribed domain data on the order of gigabytes is needed for high performance language models. This scenario calls for speech processing solutions that are able to store, access, and process this volume of data.

Software Library Goals

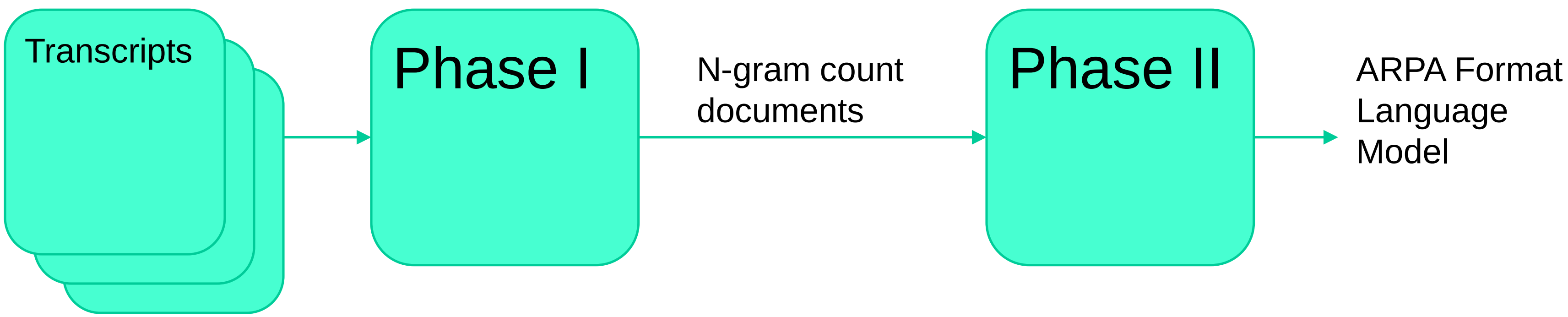
- The goal of the speech analytics library is to assemble, adapt, or develop capabilities to condition, decode, and understand speech content in an archive of speech related data.

- The initial goal of the library, for this course project, is to utilize an open-source language modeling toolkit in combination with the MapReduce paradigm to enable distributed training of n-gram language models.

Library Overview

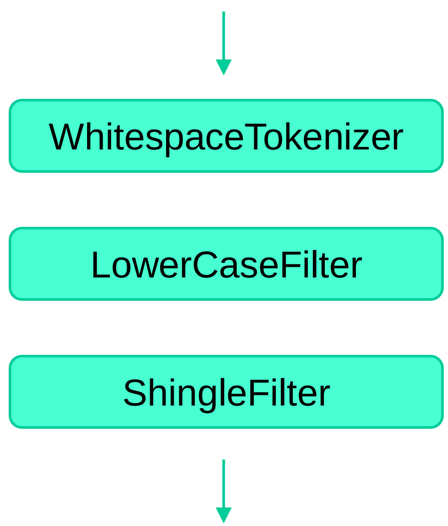
- The Speech Analytics Library is intended to be a collection of tools, and the single tool that currently inhabits the library allows the distributed training of n-gram language models for speech recognition. In the first phase, Hadoop is utilized to count n-grams across the input transcriptions. The intermediate output is an n-gram document in a simple, RSA internal format. Phase two compiles the intermediate n-gram count documents and utilizes berkeleylm for the creation of a n-gram language model in the commonly accepted ARPA format.

Language Model Generation Tool



Language Model Generation: Phase I

- Phase one processing is centered around the usage of Hadoop distributed computing and Lucene text analysis libraries.
  - The Hadoop Library is utilized to create a Java MapReduce application capable of taking textual transcriptions as input and producing output n-gram count documents. Custom Client, Mapper, Reducer, and Partitioner classes are utilized in this step.
  - The Lucene core library capabilities for analysis/tokenization are utilized by custom Mapper classes to perform analysis on input for n-gram creation from transcript strings. The sequence of Lucene TokenStream filters is shown below.



Language Model Generation: Phase II

- Phase two processing is centered around the usage of the berkeleylm library to generate a language model. At this time the second phase simply outputs this language model to an ARPA format language model file. Alternatively, berkeleylm excels at efficiently storing and querying large language models during runtime for a speech recognizer, and it is for possible future uses related to this shared focus on large data sets that berkeleylm was chosen.

Example Input

```
75122 zulu please </s>     1
75123 zulu ramp's one 1
75124 zulu roger taxi 1
75125 zulu runway one 1
75126 zulu runway three    2
```

Example Output

```
21304 -3.499307   you outside -0.155091
21305 -3.027635   uhh we  -0.155091
21306 -1.048397   papa turn   -1.030153
21307 -0.636688   national's airspace -0.155091
21308 -2.247552   it one  -0.155091
21309 -4.720337   <s> ac- -0.155091
21310 -2.517931   alpha left  -0.155091
21311 -2.371505   atis lima   -0.155091
21312 -2.335966   departure with  -0.632213
21313 -3.664310   two forty-  -0.155091
21314 -1.160186   block they  -0.155091
21315 -2.284937   care baw264 -0.155091
21316 -1.080543   westwind on -0.155091
21317 -2.438184   please i'll -0.155091
```

Library Dependencies

- The library utilizes the following existing tools to fulfill its goals.

Capability	Leveraged Tool
HDFS	Apache Hadoop
MapReduce	Apache Hadoop
Text Segmentation and Tokenization	Apache Lucene
Language Modeling	berkeleylm

- Hadoop – Chosen for solid file system and implementation of MapReduce framework. Selected for strong usage scenario with batch processing an audio archive of speech related data.
- Lucene – Top level Apache product with track record for text analysis and tokenization
- berkelylm – Library written for language modeling in the context of large data volumes. Meant for estimating, storing, and accessing large n-gram language models

Usage

- The library is currently a two phase affair, where each phase must be kicked off from the command line. Simplification to a single phase is planned. The library is built via Maven, but a pre-built JAR is included in the repository.
- Example scripts to run the two phases are included in the repository /scripts folder, and an example of one script is below.

```
#!/bin/bash

# This script is written to execute the speechtools library class LMTrainer
# The LMTrain is inteded to initiate a Hadoop MapReduce job to perform
# n-gram language model training on data in HDFS
# ${1} - input data path (currently only a file) of intermediate, ordered ngram
count file relative to the HDFS
# ${2} - output data path (file) relative to the HDFS for ARPA format ngram lang
uage model document



hadoop --config $HADOOP_CONF_DIR jar ~/workspace/speechtools/target/speechtools-
0.0.1-SNAPSHOT-jar-with-dependencies.jar ripley.speechtools.LMCompiler.KneserNey
LMCompiler ${1} ${2}
```

Conclusion

- Work completed so far allows distributed training of n-gram language models, and presents the opportunity for additional work to better utilize berkeleylm as a language model manager during runtime operations which potentially depend on large language models.