Rapid Enhancement of NLP systems by Acquisition of Data in Correlated Domains

Tejas Udayakumar , Kinnera Saranu, Mayuresh Sanjay Oak, Ajit Ashok Saunshikar, Sandip Shriram Bapat

Samsung Research and Development Institute, Bangalore, India

{t.udayakumar, kinnera.sar, mayuresh.oak, ajit.as, sandip.bapat}@samsung.com

Abstract

In a generation where industries are going through a paradigm shift because of the rampant growth of deep learning, structured data plays a crucial role in the automation of various tasks. Textual structured data is one such kind which is extensively used in systems like chat bots and automatic speech recognition. Unfortunately, a majority of these textual data available is unstructured in the form of user reviews and feedback, social media posts etc. Automating the task of categorizing or clustering these data into meaningful domains will reduce the time and effort needed in building sophisticated human-interactive systems. In this paper, we present a web tool that builds a domain specific data based on a search phrase from a database of highly unstructured user utterances. We also show the usage of Elasticsearch database with custom indexes for full correlated text-search. This tool uses the open sourced Glove model combined with cosine similarity and performs a graph based search to provide semantically and syntactically meaningful corpora. In the end, we discuss its applications with respect to natural language processing.

Index Terms: Glove Model, Elasticsearch, Automation, Speech Recognition

1. Introduction

Human-computer interaction has been growing rampantly for the last few years. Technologies like chat bots and personal digital assistants are now closer to clearing the Turing's test than ever. In such an era, the unavailability of data is a thing of the past. There is a huge amount of structured data sufficient to build these interactive systems. But the amount of unstructured data is ginormous compared to structured ones. These unstructured data can be gathered from articles, tutorials, user comments or feedback, social media posts, and whatnot. But, the availability of fewer tools and algorithms in the data science community to structure these data has been a deterrent for its usage.

One of the easiest ways to increase the usability of the unstructured data is to group them into similar categories or domains. The evolution of Deep Learning has provided us with comprehensive word embedding models which can be used to identify semantically closely related utterances. In this paper, we propose a user-friendly automation system to obtain domain specific text corpora from a vast database of utterances. In order to perform text similarity, word embedding such as bag-of-words, TF-IDF, word2vec are being used widely. For our work, we chose Glove model [1] and cosine similarity metric to measure sentence correlation. We use these collated corpora to build language models for different capsules of ASR and chat bot systems. We also use them to generate audios using text-to-speech that can be further used to train/test ASR on specific patterns.

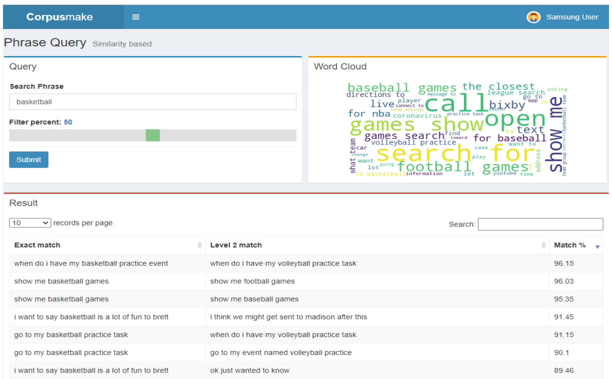

Figure 1: *Tool Snapshot*

2. System Architecture

Figure 1 shows a snapshot of our tool which contains a form to submit a search phrase and a similarity filter. When a user searches for a particular word or phrase, we fetch the related matches from the elastic DB based on the input similarity percent and provide text files or SRILM generated arpas for download. Along with the corpus, we provide word cloud data frequency visualization.

This tool can therefore help reduce the data pre-processing pipeline by automating the process of data garnering, data normalization and domain wise data extraction as shown in Figure 2.

2.1. Building Elastic database

It is highly time consuming to search for text patterns in traditional databases. Elasticsearch helps in such scenarios because of its scalability and full text-search features to make on the fly text corpora, test case suite etc. In our project we are building a database with user utterances with synonyms, homophones, and entities in key value pair format. Synonyms are extracted for each word in an utterance from python nltk synsets. We are also adding semantically similar words extracted from glove model and homophones from custom built lexicon db as meta-data for each utterance in our elasticsearch database. The Glove model used is an open-sourced 300-dimensional feature vector trained on Wikipedia 2014 and Gigaword 5. This entire meta-data is generated on the fly and indexed with text and audio files.

2.2. Data

The database is kept up to date by continuous augmentation with new data from various sources. This data is mostly in unstructured format. We have more than 5 million utterances indexed at any given point in time. The performance of the elasticsearch can be viewed in terms of retrieval of corpus based on the queried phrase which is, more than often, less than a minute.

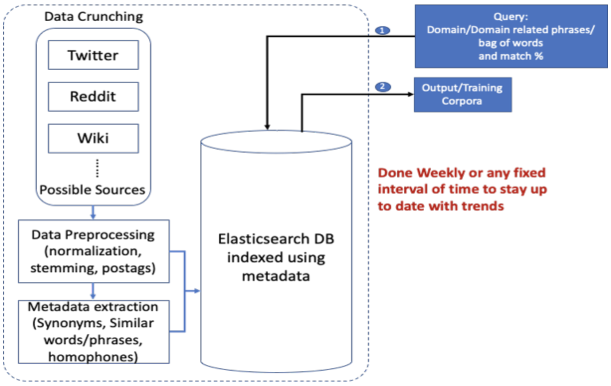

Figure 2: *System architecture*

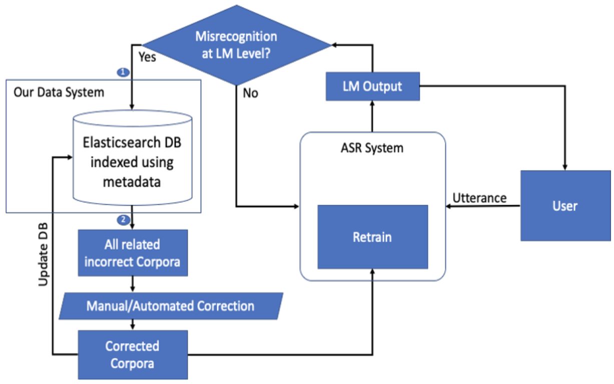

Figure 3: *Misrecognition Detection and Correction across all ASR domains*

2.3. Corpus Creation

To create a corpus of any particular domain, a search phrase is required which could be anything from a simple word to a complete utterance. After removing the stopwords and normalizing the search phrase, we perform elastic search on its meta-data to obtain the exact matches which we refer to as a level one match. Using the meta-data of each word in the level one matches, we go a level deep into the tree to get contextually linked utterances using cosine similarity on their equivalent vector format. Similarly, level i^{th} matches are obtained from the meta-data of level $(i-1)^{th}$ matches and their respective cosine similarities. On each level, the utterances can be filtered using a tunable cut-off percentage.

Exploring every word in the meta-data on an i^{th} level can be understood as a breadthwise search, whereas exploring all utterances for a single word can be understood as a depthwise search. By doing such an exhaustive search and match, we introduce variability in terms of entities and sub-domains, hence creating a comprehensive corpus. For example, consider the search phrase “basketball”. All breadthwise matches would include domains within the sport such as leagues, match scores, and MVPs. On the other hand, the depthwise matched would provide sports or events similar to basketball such as football or tennis. Therefore, by varying the similarity filter, breadth and depth levels we can create a domain-specific thorough corpus.

2.4. Applications

In this section we discuss some of the use cases of this system as a means to ease data augmentation and correction for language modelling and automatic speech recognition systems.

2.4.1. Continuous acquisition of training/testing data

We can use this system to create corpora focused on a specific domain, entity, or type of entity. e.g. If the Language model needs a context related to sports leagues, this system can be queried to whip-up a corpus encompassing all/most utterances related to sport leagues. This reduces the overhead of manual effort to extract utterances related to required domain.

Developing a robust NLP system takes a lot of iterations of training and testing. It is required to stay up to date with the current trends and events, so devising an easy way to compensate for these missing patterns is necessary. Utterances with such patterns can be collated by querying our system for key phrases or bag of words, which can then be used for testing and training.

2.4.2. Automatic data extension of QnA Systems

We have a question answer system trained with some example questions and answers, it should then learn to answer other questions from unstructured data. Eg: We train a system on “Who is the president of the US?” and it learns to answer “Who is the president of India?” or “Who is the vice president of the US” without explicit training. Our system can account for such missing data by providing domain related corpus for the initial question.

2.4.3. Misrecognition Detection and Correction

ASR systems can sometimes output incorrect words or phrases which can be either an acoustic modeling or a language modeling error. Our system can be used to fix the ASR model by automatic or manual correction of incorrect data in the corpus. Because elasticsearch is a full-text search and analytics engine with very low latency, it allows us to store, search and analyze big volumes of text data. This tool uses these features to quickly search for a word/phrase that needs replacement and change all its related instances quickly. The workflow of this system is shown in Figure 3. This feature is an extension to one of our previous works, CACTAS [2] with very low latency.

3. Conclusion

With the increasing support for native apps on mobile phones, the constant need to adapt to new data is very crucial. Simplifying the process of data augmentation, normalization and correlation extraction can be achieved using the proposed methodology. Not only this helps in training purposes, this method will be efficient to prepare test cases for specific patterns which can be easily searched using keyword, homophone or phrases. This prunes the different modules and the time consumed in data preprocessing. To conclude, this system can be used to help other NLP systems adapt and become robust in terms of data or knowledge coverage.

4. Acknowledgement

This work was funded by the AI RnD Group in Samsung Electronics. We thank VP Kyoung-Gu Woo and Dr. Sky Jung for their support and guidance on this project.

5. References

[1] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[2] M. Mathivanan, K. Saranu, A. Pandey, and J. Vepa, “Cactas-collaborative audio categorization and transcription for asr systems.” in *Interspeech*, 2018, pp. 1495–1496.