May 2019

Streaming Infrastructure and Natural Language Modeling with Application to Streaming Big Data

Yuheng Du
Clemson University, yuheng.du.hust@gmail.com

Streaming Infrastructure and Natural Language Modeling with Application to Streaming Big Data

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Computer Science

by
Yuheng Du
May 2019

Accepted by:
Dr. Amy Apon, Committee Chair
Dr. Alexander Herzog
Dr. Mashrur Chowdhury
Dr. Andre Luckow
Dr. Ilya Safro

Abstract

Streaming data are produced in great velocity and diverse variety. The vision of this research is to build an end-to-end system that handles the collection, curation and analysis of streaming data. The streaming data used in this thesis contain both numeric type data and text type data. First, in the field of data collection, we design and evaluate a data delivery framework that handles the real-time nature of streaming data. In this component, we use streaming data in automotive domain since it is suitable for testing and evaluating our data delivery system. Secondly, in the field of data curation, we use a language model to analyze two online automotive forums as an example for streaming text data curation. Last but not least, we present our approach for automated query expansion on Twitter data as an example of streaming social media data analysis. This thesis provides a holistic view of the end-to-end system we have designed, built and analyzed.

To study the streaming data in automotive domain, a complex and massive amount of data is being collected from on-board sensors of operational connected vehicles (CVs), infrastructure data sources such as roadway sensors and traffic signals, mobile data sources such as cell phones, social media sources such as Twitter, and news and weather data services. Unfortunately, these data create a bottleneck at data centers for processing and retrievals of collected data, and require the deployment of additional message transfer infrastructure between data producers and consumers to support diverse CV applications. The first part of this dissertation, we present a strategy for creating an efficient and low-latency distributed message delivery system for CV systems using a distributed message delivery platform. This strategy enables large-scale ingestion, curation, and transformation of unstructured data (roadway traffic-related and roadway non-traffic-related data) into labeled and customized topics for a large number of subscribers or consumers, such as CVs, mobile devices, and data centers. We evaluate the performance of this strategy by developing a prototype infrastructure using Apache Kafka, an open source message delivery system, and compared its performance with the latency requirements of CV applications. We present experimental results of the message delivery infrastructure on two different distributed computing testbeds at Clemson University. Experiments were performed to measure the latency of the message delivery system for a variety of testing scenarios. These experiments reveal that measured latencies are less than the U.S. Department of Transportation recommended latency requirements for CV applications, which provides evidence that the system is capable for managing CV related data distribution tasks.

Human-generated streaming data are large in volume and noisy in content. Direct acquisition of the full scope of human-generated data is often ineffective. In our research, we try to find an alternative resource to study such data. Common Crawl is a massive multi-petabyte dataset hosted by Amazon. It contains archived HTML web page data from 2008 to date. Common Crawl has been widely used for text mining purposes. Using data extracted from Common Crawl has several advantages over a direct crawl of web data, among which is removing the likelihood of a user's home IP address becoming blacklisted for accessing a given web site too frequently. However, Common Crawl is a data sample, and so questions arise about the quality of Common Crawl as a representative sample of the original data. We perform systematic tests on the similarity of topics estimated from Common Crawl compared to topics estimated from the full data of online forums. Our target is online discussions from a user forum for car enthusiasts, but our research strategy can be applied to other domains and samples to evaluate the representativeness of topic models. We show that topic proportions estimated from Common Crawl are not significantly different than those estimated on the full data. We also show that topics are similar in terms of their word compositions, and not worse than topic similarity estimated under true random sampling, which we simulate through a series of experiments. Our research will be of interest to analysts who wish to use Common Crawl to study topics of interest in user forum data, and analysts applying topic models to other data samples.

Twitter data is another example of high-velocity streaming data. We use it as an example to study the query expansion application in streaming social media data analysis. Query expansion is a problem concerned with gathering more relevant documents from a given set that cover a certain topic. Here in this thesis we outline a number of tools for a query expansion system that will allow its user to gather more relevant documents (in this case, tweets from the Twitter social media system), while discriminating from irrelevant documents. These tools include a method for triggering a given query expansion using a Jaccard similarity threshold between keywords, and a query expansion method using archived news reports to create a vector space of novel keywords. As the nature of streaming data, Twitter stream contains emerging events that are constantly changing and therefore not predictable using static queries. Since keywords used in static query method often mismatch the words used in topics around emerging events. To solve this problem, our proposed approach of automated query expansion detects the emerging events in the first place. Then we combine both local analysis and global analysis methods to generate queries for capturing the emerging topics. Experiment results show that by combining the global analysis and local analysis method, our approach can capture the semantic information in the emerging events with high efficiency.

Acknowledgments

The author would like to thank Dr. John Holt with LexisNexis Risk Solutions for the suggestion to use the Dice coefficient to compare top-ranked keywords from two topics. This work is funded in part by NSF #1228312. The query expansion part of the work is funded by NSF MRI program 1725573 and NSF #1405767.

The author would also like to thank BMW Group Information Technology Research Center in Greenville, SC for funding the research project on user forum data analysis.

Part of this dissertation is based upon research work supported by the U.S. Department of Transportations Southeastern Transportation Center under O/E Grant 2016-1017. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Southeastern Transportation Center (STC), and the U.S. Government assumes no liability for the contents or use thereof.

The author would like to thank Dr. Amy Apon for her help and advice throughout the period of the Ph.D study. The author would like to thank Dr. Alexander Herzog for his help on the language modeling algorithms applied throughout the text data curation and text data analysis projects. The author would also like to thank Dr. Ilya Safro for his advice and help in the automated query expansion project in graph analysis and emergent event detection algorithms. The author would like to thank Dr. Andre Luckow for his help and advice in the data collection project and in the forum data analysis project.

Part of the research in the data collection component is from a collaboration with Dr. Mashrur Chowdhury's team in Civil Engineering department. The author would like to thank Dr. Chowdhury's advice in the distributed data delivery framework project.

Finally, the author would like to thank Clemson University's Palmetto Cluster for providing the computation resources to support all of his research projects.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction and Thesis Statement

As the foremost component of our proposed end-to-end streaming data analysis system, the data delivery framework lays the foundation of the system. It directly handles the collection and distribution of the streaming data being studied. Since the movement of data in the end-to-end pipeline is unidirectional, the quality of data analysis results highly depends on how fast and accurate the data collection component can distribute data into the curation and analysis component of the pipeline. Therefore, the quality of the data collection component is crucial in our research.

The second component of our proposed system is the data curation system that handles streaming text data. Our experiment focus on an active automotive user forum where users post technical user experiments about automotive products. Our curation component use a language model to capture the underneath discussion themes in these streaming text data. However, applying such model directly to large streaming data can be problematic due to the tedious computation involved. Our approach uses a data sample of the full data to speed up the computation. We also study the problem of statistical bias of sampling which is often over-looked in big data curation method.

The third component of our proposed system is the data analysis component that automatically detects when an emerging event takes place in streaming social media. We then use a mixture of global analysis method and local analysis method to generate queries that adapt to the dynamic nature of words used in Tweet streams.

1.1 Data Collection and Format of Automotive Data

The real world deployment of connected vehicle (CV) technologies, promoted in the United States and some other countries, entails developing roadway transportation systems that involve wireless interconnections between vehicles, and transportation and information infrastructures to provide safe and reliable transportation [1]. The American Association of State Highway and Transportation Officials (AASHTO) footprint analysis report identifies three elements that are necessary for such a dynamic and complex connected transportation system. Specifically, the system must 1) improve safety through vehicle-to-vehicle communication; 2) improve traffic operations by providing travel time information to drivers, transit riders, and freight managers in real-time; and 3) reduce the adverse environmental effect using real-time information to enhance fuel efficiency [94].

Rather than being a separate environment, CV systems will become part of a bigger connected society, such as city wide smart disaster management systems that include smart traffic management, smart power grid, and smart healthcare, as described in the vision for Smart Networked Systems [82]. An explosion of CV data is expected when one considers how much data can be collected continuously via a large number of sensors (e.g., sensors in vehicles, cell phones, roadside units). These data will come in varied formats (e.g., PDF, CSV, and structured/unstructured XML) [58],[23],[2]. In addition, different types of applications and the numbers of users requiring specific subsets of CV data from different sources will increase significantly as the market penetration of CVs in the roadway traffic stream increases over time.

The primary challenges for a CV message delivery system involve redistribution of data at various stages of the data lifecycles (i.e., raw, integrated, and processed) while meeting specific functional requirements based on time and spatial contexts. These requirements and contexts can vary depending on which CV applications being considered. For example, traffic management applications alone can generate multiple streams of data from various sensors of thousands of CVs. These data must be correlated and analyzed in real-time to identify traffic conditions and to provide further operational actions. This analysis will then make it possible to establish a hierarchy of various operational actions to ensure proper traffic condition assessments and selection of management decisions. These challenges also lead to some potential issues, which include: 1) increased latency for delivery of data in an usable format from the raw data as per CV applications that may require specific time-sensitive data; 2) requirements for more data storage at transportation data processing centers; and 3) failure of data processing machines at specific transportation centers that could create a risk of large scale failures in the CV ecosystem. In a previous work, Lantz et al. [56] presented a framework for the data infrastructure that can support the handling of massive volume of data in urban transit networks. This study discussed that a high rate of data arrival and data access depend on the capability and capacity of the data infrastructure.

The primary objective of this component of the dissertation is to design a distributed message delivery infrastructure for connected vehicle systems. A key design assumption for this infrastructure is that the standard data processing centers (data warehouses), such as those managed by public agencies, are the core locations from which data is delivered to support hundreds of CV applications. In our design, the raw data are separated, tagged, and posted by a software component, which is a distributed message delivery platform. In this strategy, whichever entity requests data will contact the platform to identify the tags of interest and receive the relevant data via the platform. The conceptual vision for this infrastructure is illustrated in Figure 1.1, where different roadway traffic related (e.g., vehicles and centers) and non-traffic related (e.g., news and weather) entities serve as both data producers and consumers to support CV and other transportation related applications. The addition of the message delivery system enables a data-focused view of the entire CV ecosystem in which any single entity can deliver data as well as acquire data.

As there is no fully functional, non-experimental connected vehicle system to support the study of message delivery behavior, we rely on a study of Connected Vehicle Reference Implementation Architecture (CVRIA) metadata in order to estimate the potential demand of data for different CV applications [3]. More specifically, we are interested in the temporal distribution of CV data, which is very critical for modeling the various CV application scenarios and types of data flow, as the message delivery systems must maintain acceptable latency for CV applications. In this paper, we develop and evaluate the performance of a prototype infrastructure of a distributed message delivery system, which could support future CV systems and applications, using an open source distributed streaming platform, Apache Kafka [54].

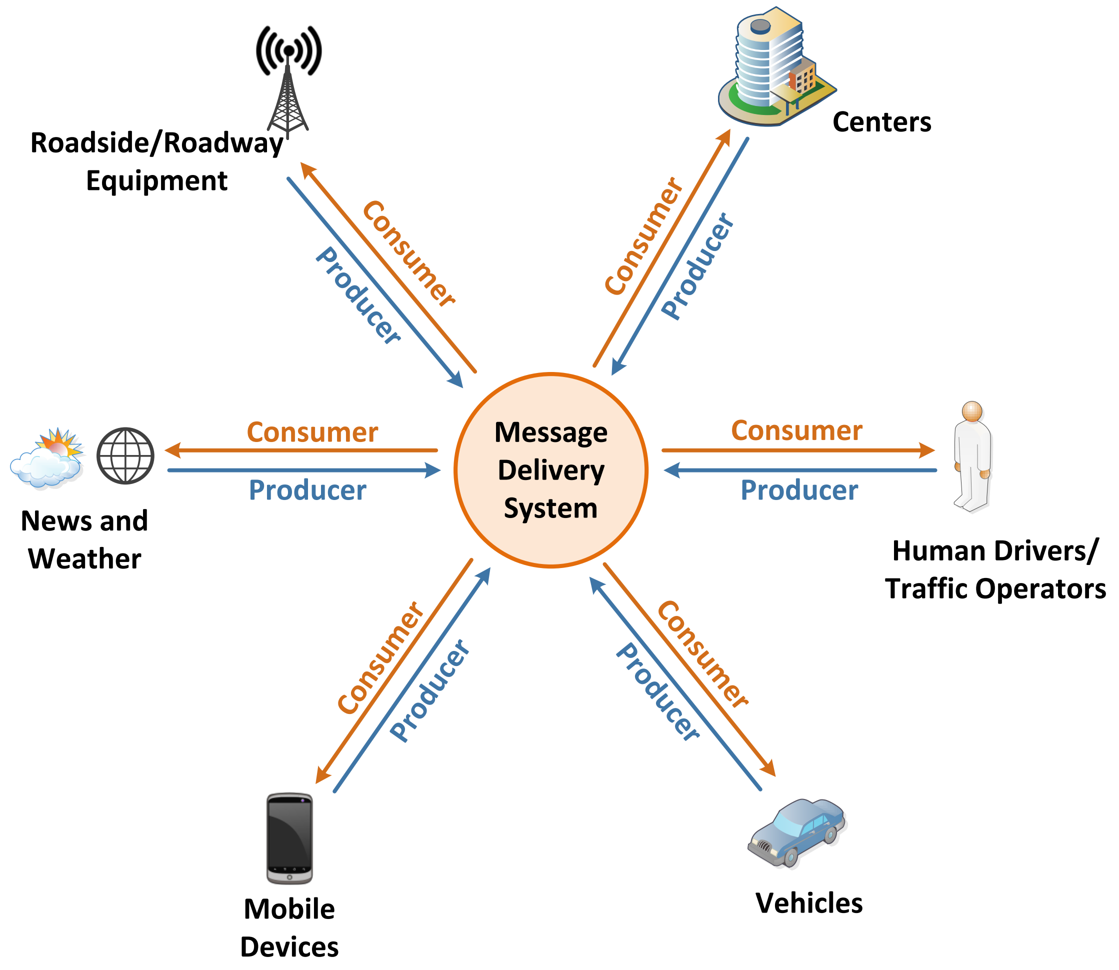

Figure 1.1: A conceptual vision for message delivery infrastructure for a connected vehicle system

1.2 Data Curation of Streaming Text Data in Automotive Applications

The second component of our end-to-end system is the data curation component which uses data coming out from the data collection component discussed in Section 1.1. In this section, we use text data collected from online forums to study the efficacy of our language model-based data curation component.

Social media and online forums provide a wealth of data to inform design,engineering, sales and marketing of consumer products. Increasingly, consumers use a wide variety of online services, e. g. Facebook, Twitter, forums, and blogs, to share information and experiences about products and services. Linking product development, sales and marketing to customer needs is a critical capability. The aim of this paper is to investigate the usage of advanced analytics, in particular, natural language understanding techniques, to detect main themes in online forums. Online discussions can help companies to better understand their customers' needs and to improve their products. In comparison to other social platforms, forums can contain very technical and detailed feedback information from advanced users.

Textual curation approaches are typically based on bag-of-words, n-grams and/or term frequency-inverse document frequency (TF-IDF) [52] data representations. In most cases, these text representations are used in conjunction with classification models, e. g., sentiment classifiers. However, this approach does have some limitations: bag-of-word and TF-IDF representations do not capture themes within documents or semantic relationships. Further, classification approaches require that the data is labeled, which is time-consuming and expensive. In particular, for investigative and exploratory analytics other approaches are better suited, e. g., the ability to capture discussion themes or topics [89]. Topic models [26] are algorithms that can detect common topics across a corpus of documents. The most well-known algorithm is Latent Dirichlet Allocation (LDA) [29]. Topic modeling provides an unsupervised learning method to analyze the main themes in a collection of documents. It can reject the noise in the data and recover the underlying topics hidden in each document without manual tagging. In this paper, we use LDA topic modeling to analyze online forum discussions.

Despite the valuable information provided by online forums, these also have several characteristics that make them intractable to study directly. For example, forums contain a tremendous amount of historical data. Massive online forums such as Gaia Online [17] contain more than 1 billion posts, with a daily count of 20,000 active users. Crawling through the full data of these forums may take months. Continuous crawling of a forum website can also result in blocking of IP addresses. Besides the large size, the directly crawled data are often noisy in nature. Although a forum has structured formats, such as threads and tags to guide discussions, users tend to go off-topic in a thread and spawn multiple discussion themes. Capturing these representative discussion themes requires complex natural language understanding algorithms ([29, 60]). Performing these algorithms on full forum data is a very time-consuming task. Therefore, analyzing large, noisy and complex forum data needs a more efficient strategy.

Common Crawl [36] is an open repository that contains petabytes of web crawl data covering over nine billion web pages [32]. For efficiency purposes, it does not provide the full data of the webpages being crawled. Instead, it provides samples of the online forums in the form of static snapshots. Common Crawl currently performs a monthly crawl based on a two tier crawling strategy which insures that pages with higher page ranks are visited and the overlap between each crawl is minimized. A crawl takes a snapshot of the pages being visited and saves the crawled data into a structured format.

Common Crawl provides a sample of the original online forum data with unknown biases. It is not an independent dataset with respect to the original forum data we are studying. Using Common Crawl as a sample of the full forum data for topic modeling has several advantages: the data is public accessible and ready to use. It avoids many pitfalls that are involved in creating a custom crawler (e. g., the prioritization of web page, blacklisting of the crawler, etc.). Common Crawl snapshots are static, which means they provide consistent data when an analysis requires repeatability. However, one drawback of Common Crawl is the uncertainty with respect to data quality and completeness and thus the ability of using these data for topic modeling. On average, we observed that Common Crawl only contained about 22 % of the data of interest. This is in line with other investigations of the Common Crawl dataset, e. g., by Stolz and Hepp [85]. As the precise collection algorithm of Common Crawl is not known, the data cannot be assumed a true random sample as it may be subject to sampling bias. Thus, it is also not possible to define for which class of data analysis algorithms the data is appropriate and how the results generalize.

In this dissertation, the use of Common Crawl data as a sample for extracting representative LDA topics from online forum textual data is evaluated. Our focus is on a very active car owner forum that is organized into 14 subforums representing different car models. For each subforum, we collected all available data from Common Crawl – a total of about 280 GB of raw data files. In addition, we developed a customized web crawler to collect the full 2.16 TB data from the online forum. Having access to the full data for each subforum allows us to systematically evaluate the representativeness of topics estimated from the Common Crawl samples compared to the full data. Further, because the subforum samples differ in terms of document count, sample proportion, and other features, we can investigate the relationship between data features and topic representativeness.

1.3 Automated Query Expansion by Expanding Topic Models of Streaming Social Media Data

Query expansion (QE) is a type of analysis used to enhance the results of a given information retrieval problem. Types of QE usually identify a method of substituting or adding words to a given filter to collect relevant, and exclude irrelevant, data in a set of documents [87]. Furthermore, the application of QE to the realm of social media and micro-blogging streams is applicable given the high-frequency and noisy nature of these documents, and the real-time information and usefulness they provide for developing events (i.e., public events, states of emergency). Therefore, developing new methods of QE to analyze noisy and high-volume streams is valuable in discriminating useful from non-useful information.

Our motivation for this type of QE lies in helping to overcome the personal, professional, or demographic biases people often carry with them into a manual information retrieval problem. Usually, selecting keywords to narrow a large set of documents for analysis is biased by political sentiment [24]. The purpose of this paper and its proposed system, for instance lies in overcoming the biases people may bring into an information retrieval problem from expertise (or lack thereof). We primarily address this by building a high-dimensional feature space from archival news, and using its proximity to the emergent words in the stream of interest to addend novel keywords into our queries.

As shown in Figure 1.2, the proposed parallel query expansion pipeline adopts the keywords from both primary stream (initial tweets stream) and from external corpus (secondary stream). Since our focus is the emergent topics in a small time span of the primary stream, we split the primary stream into a series of time windows of same length (15 minutes) in our analysis. A dynamic semantic graph is used to monitor the emerging keywords in each time window. When we detect new emerging keywords, LDA topic modeling is performed to extract candidates of emerging topics in the time window being identified. External corpus are represented as secondary stream in this figure. It consists of news articles collected in the same time span as of the primary stream(s). The external corpus are useful as it provides historical information about semantic relationship between keywords used in the query expansion pipeline. This auxiliary information combines with the semantic information learned through topic modeling the dynamic semantic graph (dynamic eigenvector centralities) gives an comprehensive view of the query keywords to generate, which is

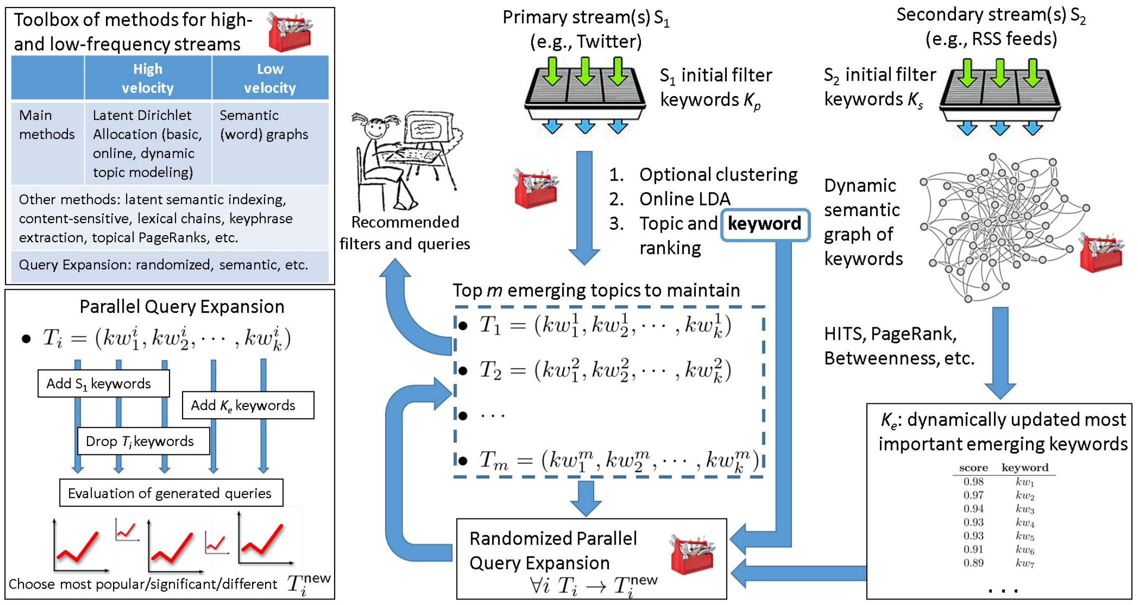
Toolbox of methods for high- and low-frequency streams
High velocity
Low velocity
Main methods
Latent Dirichlet Allocation (basic, online, dynamic topic modeling)
Semantic (word) graphs
Other methods: latent semantic indexing, content-sensitive, lexical chains, keyphrase extraction, topical PageRanks, etc.
Query Expansion: randomized, semantic, etc.
Parallel Query Expansion
$T_i = (kw_1^i, kw_2^i, \cdots, kw_k^i)$
Add S_1 keywords
Add K_e keywords
Drop T_i keywords
Evaluation of generated queries
Choose most popular/significant/different T_i^{new}
Recommended filters and queries
Primary stream(s) S_1 (e.g., Twitter)
S_1 initial filter keywords K_p
1. Optional clustering
2. Online LDA
3. Topic and keyword ranking
Top m emerging topics to maintain
$T_1 = (kw_1^1, kw_2^1, \cdots, kw_k^1)$
$T_2 = (kw_1^2, kw_2^2, \cdots, kw_k^2)$
$\cdots$
$T_m = (kw_1^m, kw_2^m, \cdots, kw_k^m)$
Randomized Parallel Query Expansion
$\forall i\ T_i \to T_i^{\text{new}}$
Secondary stream(s) S_2 (e.g., RSS feeds)
S_2 initial filter keywords K_s
Dynamic semantic graph of keywords
HITS, PageRank, Betweenness, etc.
K_e: dynamically updated most important emerging keywords
score keyword
0.98 kw_1
0.97 kw_2
0.94 kw_3
0.93 kw_4
0.93 kw_5
0.91 kw_6
0.89 kw_7
...

Figure 1.2: Illustration of QE with Data Fusion

crucial in constructing our proposed query expansion method.

1.4 Thesis Statement

The goal of this thesis is to provide a holistic study of the end-to-end system we have designed, built and analyzed to collect and analyze streaming data. For the data collection component, we investigate the hypothesis that a proposed distributed data delivery framework can handle the real-time requirements for Connected Vehicle applications. A major contribution of this component is that we provide evidence to show our data delivery framework is capable of handling rapidly produced streaming data in automotive domain. For the data curation component, we investigate the hypothesis that using a data sample of online forums to support the language modeling task, we can get comparable results as from full data. A major contribution of this component is that we provided evidence that the data sample we study is capable of support topic modeling task on online forums as good as any random sample, which gives results comparable to the whole dataset. For the data analysis component, the key hypothesis we investigate is that the proposed query expansion algorithm is capable of detecting emerging event and provide effective query words suggestions. A major contribution of this component is that we give evidence that our query expansion mechanism is capable of detecting emerging event in streaming socia media data. Also by combining the local analysis method and global analysis method in query expansion, we show that the proposed approach captures the semantic information of emergent events more efficiently than either the static query method or the local analysis method alone.

The remainder of this thesis is organized as follows. We first discuss background work in Chapter 2. In Chapter 3, we provide an overview of the data curation model we proposed and the query expansion algorithm we designed. We discuss our experiment results in Chapter 4, where we evaluate the performance of our data collection pipeline; discuss the topic similarity between the full automotive text data and the sample data under proposed sampling techniques; and analyze the efficacy of our proposed query expansion algorithms using a tweet dataset we collected. In Chapter 4, we also demonstrate business insights that can be drawn from our estimated topics. Chapter 5 summarizes the contributions of each component of our end-to-end system in details.

Chapter 2

Background

This chapter introduces the background work related to the three components of our proposed end-to-end system. First we focus on the distributed data delivery frameworks which the data collection component is built upon. Then we introduce the language model used in our data curation component. Finally we introduce the query expansion problem studied in the data analysis component.

2.1 Distributed Data Delivery Frameworks

A review of previous studies related to the evaluation of message delivery systems for connected vehicles is discussed in the following two sub-sections. The first sub-section reviews the current status of connected vehicle systems deployment. The second subsection explores existing message delivery systems for connected vehicle applications.

2.1.1 Connected Vehicle Systems Deployment

Connected vehicle systems demand an integrated deployment plan both from private sectors, such as automobile manufacturers, and public sectors, such as U.S. Department of Transportation (USDOT) in the US. Several large-scale pilot deployments, sponsored by the USDOT, are underway to develop market ready technologies before mandating Vehicle-to-Vehicle (V2V) technologies for new vehicle models in the next few years [4]. The Southeast Michigan Test Bed is one such federally funded publically available center for analyzing CV technologies [10]. This test bed provides a real-world laboratory to evaluate CV applications. In 2012, two CV test beds were deployed in Virginia for connected vehicle research [11]. The first is located in Blacksburg, VA, at the Virginia Smart Road and along Route 460, in the southwestern part of the state. The second is located in Fairfax County along I-66 and the parallel Routes 29 and 50, which are in the northeastern part of the state. The Connected Vehicle Infrastructure University Transportation Center (CVI-UTC) is using both sites to develop a fully operational test bed for connected vehicle mobility applications, and the Fairfax site will be used for dynamic alternate route research [5]. In 2015, Florida, California, and New York were also selected as new sites for similar test beds [16]. While the majority of operating test beds are used to analyze CV application development and evaluation, there is no test bed for data infrastructure evaluation for real-world CV applications.

The Michigan connected vehicle testbed Proof of Concept (POC) test reports identified that latency to deliver a message from a vehicle to the application server (i.e., from a connected vehicle to a roadside unit (RSU)) is between 0.5 and 1.5 seconds for CV application [46] [40]. This time range depends on the technologies for wireless communication and backhaul data transfer, and data congestion in the communication network. As the POC test used a limited number of connected vehicles, it was not possible to evaluate latency in the presence of network congestion. Using IntelliDrive probe vehicle data to analyze CV system performance, Dion et al. [40] observed an average 65s latency when Vehicle-RSU protocols permitted the sequential upload of messages, while an average latency of 30 seconds was observed when vehicles were allowed to immediately transmit all messages simultaneously to RSU. In a connected vehicle system, multiple CV applications could run simultaneously with a massive amount of data sent and received between vehicles and RSUs. This would increase resource contention and latency significantly, which would eventually lead to data loss if the data infrastructure design cannot handle a massive amount of data. Furthermore, no study evaluated message delivery systems between RSU and data centers of different transportation centers.

2.1.2 Message Delivery System for Connected Vehicle Applications

Although a significant body of work is available on message delivery among CVs and RSUs via vehicle ad hoc networks (VANETs), the focus of those studies was on direct vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication that use direct and localized wireless communication technologies [99, 59, 19]. The message delivery system using VANETs lacks support for aggregation and processing of raw data from multiple sources, including those that are not from CVs and RSUs. This can limit the applicability of VANETs to CVRIAs applications [3], where sources, time contexts, and spatial contexts of input data can vary. As a result, there is a need to create a middleware layer that can manage these distributed data streams independently.

Such middleware components, often described as distributed message delivery frameworks, are integral to cyberinfrastructure in various areas, including social media, manufacturing, and e-commerce [6, 7, 8, 9, 50, 80, 88, 71, 41]. These frameworks often consist of a cluster of interconnected computers that rapidly ingest streaming messages generated by data producers (e.g., sensors, roadside units). Different frameworks distribute the data to consumers (e.g., traffic management centers, mobile applications) via push (i.e., the data are pushed to the consumers) or pull (i.e., the consumers pull data from the frameworks storage space) models. The individual computers in the frameworks are often called brokers. The ingestion and distribution processes are distributed among the message brokers to ensure a parallel operation of both processes. There exist a number of such frameworks, including RabbitMQ [20], ActiveMQ [6], WebsphereMQ [7], and MQTT [10]. Of these, RabbitMQ is one of the most popular open source solutions and is widely used in academic and industry settings [69].

RabbitMQ [10, 69, 78] uses volatile storage as the primary means for storing incoming data, with messages collected and stored in an abstraction named queue. The queues reside in memory and allow consumers to pull messages from them directly. Since the queues in memory are limited in space, messages not acquired by consumers are moved to a secondary storage level on the disk prior to being purged. Such purging can increase in data retrieval time and data loss due to message delivery latency [78, 48, 84]. This is a concern for CVRIA applications, as message delivery latency can increase when there is an increased number of consumers during peak hours or more data are being streamed to brokers in critical events. This characteristic of RabbitMQ could negatively affect some critical CV applications (e.g., cooperative adaptive cruise control) that are sensitive to data accuracy and data loss.

The rapid development of data-driven infrastructures over the last decade has led to the creation of another distributed message delivery system implementation, Apache Kafka [54]. Apache Kafka was originally used as a distributed aggregation framework for massive amount of system log messages. Apache Kafka is highly scalable and fault tolerant, and it uses the concept of topics to represent data streams. Topics are viewed as data files stored in persistent storages during message delivery. A topic can be divided into several partitions that are located across different brokers, enabling both streaming data sources and data consumers to produce and consume messages in parallel [69]. This leads to Kafkas capability to deliver large volume of messages simultaneously [63, 93]. A performance evaluation of Kafka shows that it outperforms similar alternatives in terms of messages delivered per second by both data-generating and data-requesting entities [54]. Thus, Kafka appears more suitable for CV applications that need high throughput and persistent message delivery. Given that there has been no previous study on the performance of distributed message delivery system based on Kafka for CV systems, our paper studies this middleware for designing distributed message delivery system. There is a difference between distributed message delivery systems (e.g., Apache Kafka) and distributed stream processing systems. As the names suggest, the former focuses on the delivery of messages, and the latter focuses on the processing of streaming messages.

Our study focuses on a message delivery infrastructure, based on Apache Kafka, for the distributed delivery of streaming messages. The streaming message delivery infrastructure creates a preliminary data acquisition and ingestion in which stream processing and batch processing tasked with various service level agreement (SLA) can extract the relevant message streams based on their own capacity and run-time requirements. Distributed message delivery systems and distributed stream processing systems are used to support large-scale data infrastructure for interactive data analytics [98, 86]. Examples of distributed stream processing systems include the work by Biem et al. in which they developed a data stream-processing infrastructure that supports the analytical tasks for real-time streaming data [34] and other work on advanced streaming platforms, such as, Spark [100], Flink [33], and Storm [76].

More recently, a new area of research in message delivery in computer network has been explored with the focus on replacing the TCP/IP bottleneck, which requires mapping from users requested contents to the specific location of the contents. Specifically, the use of a content-centric networking paradigm makes it possible to decouple the location from a data users identity, network security, and data access [91, 73, 74]. Preliminary work in a content-centric network has shown to be applicable for data collection and dissemination at ranges longer than the standard VANET [91], [92, 35, 57]. The message delivery concept of our distributed message delivery infrastructure is similar to the content-centric networking, where the message streams are made available and accessible to users via the streams content tags only, and the users are not required to know to the origin of the message streams.

2.2 Online Automotive Forum Data Analysis

One problem that has often been overlooked in big data research is the quality of bias of the dataset being studied [45, 42]. Our approach evaluates the quality of the data sample under a language modeling task both quantitatively and qualitatively.

2.2.1 Topic Modeling on Online Forums

The value of online forum data has been broadly studied in behavioral research (e.g., [55, 68]). For example, Wu et al. collect data from one of the largest online discussion forums in China to identify the principal users who contribute to a discussion topic [95].

Topic modeling enables research on online forums by identifying underlying topics in forum discussions. Chen et al. use a two tier model to identify popular topics in a large online forum that contains 881,190 posts [34]. The topics identified with a topic model can also serve as data labels because topic models are a form of mixed clustering. Zhou et al. take advantage of the commonly seen Question-and-Answer discussion style in online forums and apply topic modeling to assist the task of suggesting semantically similar questions to a user query [102]. Ramesh et al. use topic modeling to analyze student discussions in three massive open online courses from Coursera [75].

Analysis of vehicle online forums can provide business insights to manufacturers and vendors, such as market structure information. Netzer et al. apply text mining methods to a sedan car forum to estimate sentiment relations between different car models [70]. One finding is that these sentiments are not always explicit and often comprise only a small portion in all forum discussions. That is, car owners generally discuss problems encountered or modifications to their cars without using strong sentimental words. Human tagging is used to evaluate effectiveness of the text mining approach in [70], which is labor intensive and may not be feasible when dealing with massive datasets.

Wu et al. estimate topics from a Honda car owner online forum, which they use to predict how likely a user will participate in a future discussion on a specific topic [96]. They demonstrate that this prediction performs better for regular and active users, and that participation willingness is affected by peer participation in a topic. Shi et al. find that this peer-to-peer relation can rely on other more subtle behaviors, such as browsing [81].

2.2.2 Common Crawl Dataset

The Common Crawl data archive [36] is a gigantic public repository of web crawled data, collected and maintained by a non-profit organization "dedicated to providing a copy of the Internet to Internet researchers, companies and individuals at no cost for the purpose of research and analysis" [37]. Previous research has used the data repository to analyze the graph structure of the web over time [65]. Since a large proportion of the data included in Common Crawl is in the form of text, Common Crawl has also been used in Natural Language Processing (NLP) research, including machine translation ([32, 83, 22]), text classification [49], and taxonomy development [79].

Buck et al., for example, use Common Crawl to build 5-gram counts and language models that improve statistical machine translation [32]. Smith et al. crawl lateral contents of different language pairs from the Common Crawl corpus and use the results to facilitate language translation approaches [83]. Iyyer et al. use Common Crawl data for evaluating a sentiment classification algorithm based on a deep neural network [49]. Seitner et al. build a tuple database from Common Crawl where each tuple represents a "is-a" relationship between two words, which can be used to analyze more complex taxonomies [79].

2.2.3 Comparing Topics

Several methods exist to compare the quality of estimated topics with each other, including perplexity [90], semantic coherence [67], and exclusivity [25]. These methods apply to the comparison of topics estimated on the same data, but using different model parameters (e.g., different number of topics). Our goal is different. We compare LDA topics estimated from two different data sets – Common Crawl and the full data.

To the best of our knowledge, there has been little research on the comparison of topics estimated from different data sources. An exception is [101], which measures the similarity between topics estimated from Twitter and traditional news using the Jensen–Shannon (JS) divergence. This measure compares two topics based on their full word distributions. Our approach (explained in more detail below), in contrast, relies on a set-based comparison between the top keywords from each topic. The top keywords are determined from the word-topic probabilities, but the probabilities themselves are not being compared. We use this measure instead of the JS divergence because our goal is to mimic human evaluation of topic similarity, which would be based on a visual inspection of the top keywords between topics.

2.3 Automated Query Expansion by Expanding Topic Models of Streaming Data

Automated Query Expansion has long been suggested to cope with the word mismatch scenario in information retrieval tasks [97]. Xu [97] analyzed three representative techniques to perform automatic query expansion for a classic information retrieval scenario. The global analysis technique uses the method suggested by Jing [51]. It generates a thesaurus-like database that provides a ranked list of phrases related to the set of words in a given query. The method is referred as Global analysis approach since the association database it uses takes the whole collection of documents into account and the process is often computational heavy. Moreover, the task in Xu's work is different from our task, where the information to query are streaming data. This means the thesarus-like database method used in Jing's work is not directly applicable. The database needs to be updated in an online fashion for each tweet, which makes the method fail in large scale data.

The second approach Xu [97] mentioned uses local feedback method. Instead of using top-ranked phrases from the entire corpus to generate the thesaurus, this method simply uses the top-ranked terms and phrases from the documents in query results. Efficacy of this method is hugely dependent on the quality of the query result itself. Therefore, the reliability of the local feedback method remains an issue even it is less expensive to perform.

Using a combination of the global analysis approach and the local feedback approach, Xu [97] also introduced a novel method called local context analysis. In the first phase, top concepts (noun groups share the same semantic meaning) are identified from the ranked query result. Then in the second phase, new terms are picked from the concepts in phase one based on their distance from original query terms in the global thesaurus and their *tf idf* scores. This method peforms better than using the global analysis or local feedback analysis method independently. However, it also requires a static metric to rank the documents in query result. For streaming social data query scenario, the metrics to estimate the goodness of the query result is often dynamically changing and may comprise of a mixture of various sub-metrics. Hence, it is not feasible to directly use the local feedback method introduced in [97].

A more comparable task in query expansion is the one used in Massoudi [64]'s work, which also uses tweet data as the query platform. It is interesting to note that [64] uses a time dependent indicator to cope with the dynamic nature of streaming data. The idea is similar to our approach that uses a dynamic metric to estimate query quality. For example, [64] uses repost count and followers of post author as indicator of a tweet's quality which changes with the time. Our approach uses hashtags information and a graph based approach that takes into account the semantic space coverage of the tweet.

Chapter 3

Research Design and Methods

In this chapter, we give an overview of the design in each of the three components in our end-to-end streaming data analysis system. First we present the design and experiment setup of the data delivery framework, which is the data collection components in our system. We also give a case study to describe the experiment setup in detail for evaluating the data delivery framework. Secondly we present the method for the text data curation in application to streaming text data. Finally we present our proposed method for query expansion problem in application to Twitter data.

3.1 Emulation of Distributed Data Delivery Framework Setup

Our data collection component comprise primarily of a data delivery framework. We perform a series of experiments to emulate this framework and evaluate its performance in delivering the streaming data we collected in automotive domain. First we describe the format of the streaming data we collect by analyzing the metadata.

3.1.1 Metadata Analysis

Key in the use of Kafka is the identification of the topics that will represent the CV data streams. To identify the topics for our case study, we use CVRIA [3], which has been developed by the USDOT and which forms the basis for a common language definitions and early CV deployment concepts, as a starting point. CVRIA identifies key interfaces across different entities/stakeholders in a connected vehicle system and supports standard development activities. The CVRIA architecture is composed of four viewpoints: Physical, Functional, Communications and Enterprise. The physical view represents the connection between physical and application objects, whereas the functional view shows the logical interconnections between functions. The enterprise view represents the relationship among the organizations responsible for their respective roles in a CV environment. Finally, the communications view organizes the different communication protocols that enable communication between application objects. CVRIA has defined the concept of operations for ninety-five CV applications, and categorized as: 1) environmental, 2) mobility, 3) safety and 4) support [3]. For each application, a physical architecture was developed that identified the physical objects/centers, application objects/equipment packages and information flows between application objects to support each applications functional requirements. Each information flow of a connected vehicle application includes the following two contexts that are related to this study: 1) spatial and 2) time context. The spatial context is classified into 1) adjacent (0 m 300 m), 2) local (300 m- 3 Km), 3) regional (3 Km 30 Km), 4) national (30 Km - National) and 5) continental (Continental US) categories [3]. The time context is classified into 1) now (less than 1 second), 2) recent (1 second 30 minutes), 3) historical (30 minutes - 1 month) and 4) static (greater than 1 month) categories [3].

As each connected vehicle application has a specific data characteristic in terms of spatial and time context, we have analyzed metadata of different CV applications to understand the real-time service requirements in terms of spatial and time contexts, which provides the basis of designing distributed message delivery system to support different CV applications. For example, in the Speed Harmonization application, the 'variable speed limit status' information flows from the traffic management center to ITS roadway equipment. The characteristics of this information flow are local in 'spatial context' and recent in 'time context'. Consequently, it is challenging to deliver data concurrently for different CV applications. An analysis of CVRIA information flow is useful to support the design of data analytics infrastructure for connected vehicle systems. The number of unique information flow can be defined as the total number of information flows required for all the CVRIA applications without any redundancy. As the same information flow can be used by different CV applications, aggregation of all information flow is required to reduce redundant information flows. For example, 'traffic flow' is a unique information flow, which contains traffic flow variables (e.g., speed, volume, and density measures). The source of this information flow is 'ITS Roadway Equipment' and destination is 'Traffic Management Center'. This information flow is required in the following CV applications: 1) Cooperative Adaptive Cruise Control; 2) Eco-Cooperative Adaptive

Cruise Control; 3) Eco-Lanes Management; 4) Eco-Ramp Metering; 5) Eco-Speed Harmonization; 6) Eco-Traffic Signal Timing; 7) Intelligent Traffic Signal System; 8) Intermittent Bus Lanes; 9) Low Emissions Zone Management; 10) Queue Warning; 11) Roadside Lighting; 12) Speed Harmonization, and 13) Variable Speed Limits for Weather-Responsive Traffic Management. To avoid duplication of information flows, we count this information flow as a single flow for all CVRIA applications. To determine the number of unique information flows, we collected and stored all information flows related to 95 CV applications identified so far (at the time of writing this paper) in CVRIA. We then wrote a python script for analyzing all the information flows to determine the number of unique information flows based on the time and spatial context. All unique information flows (shown in Figure 3.1) that originated from different centers (e.g., Traffic Management Center) are coded as 'from center'; and all unique information flows that are received by the different centers are coded as to center.

All of the unique information flows in 'to center' and 'from center' categories are classified for spatial context (i.e., A) adjacent, B) local, C) regional, D) national, and E) continental), and time context (i.e., 1) now, 2) recent, 3) historical, and 4) static) and presented in Figure 3.2 and Figure 3.3, respectively. Figure 3.1 shows the comparison between the frequency of unique information flows to the centers and from the centers for all the CV applications in CVRIA. CVRIA identified 25 centers that are producing data, and 27 centers, which are receiving information. This figure also shows the number of unique information flows that need to be aggregated for each center. There are a total of 231 information flows received by all the centers with a total of 219 data flows sent from all centers. The Traffic Management Center receives and sends the highest total number of information flows. Figure 3.2 presents the distribution of information flows, based on time and spatial context, to the centers. Information flows are also classified in different combinations of spatial and time context categories for all CV applications. We observed that most of the data flows are in recent and local (2B), and recent and regional (2C) categories. Figure 3.3 presents distribution of data flow, based on time and spatial context, from the centers. We observed similar distribution in 'to center' information flows and most of the information flows are in 2B and 2C categories. The identification of unique information flows will reduce the total data volume by eliminating the redundancy of the information flow in a distributed message delivery system design.

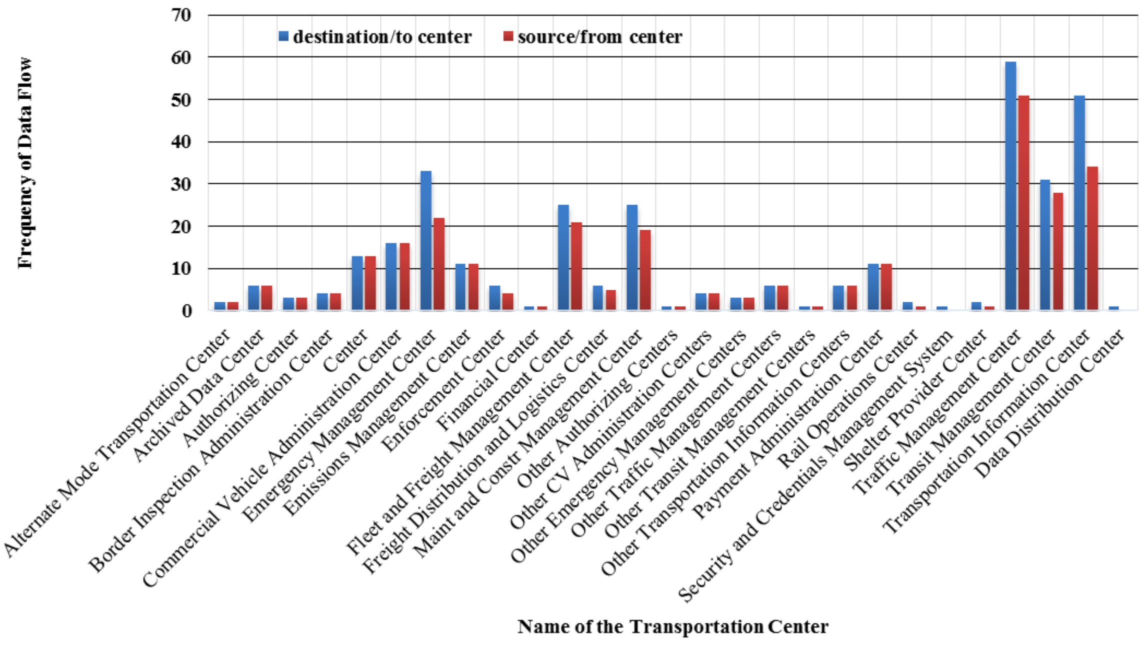

Figure 3.1: Frequency of data flow by "from center" and "to center".

3.1.2 Distributed Message Delivery System for Connected Vehicle Applications

As connected vehicle systems becomes more integrated with a smart and connected society through new technologies, such as, automated vehicles and large-scale sensor networks, the demand for access to data in a connected vehicle system will also increase rapidly. Therefore, it is beneficial to shift from an application-centric view to a data-centric view. In a data-centric view, different transportation centers can provide messages that is tagged, instead of having to support messages that fit the requirements of applications. Meanwhile, the applications are responsible for the acquisition process. This is achievable by creating a Kafka layer to manage the various tagged types of data. The other two types of entities within this infrastructure are producers, which create messages, and consumers, which acquire messages from brokers. The flow of data in a distributed message delivery infrastructure from RSU to center is illustrated in Figure 3.4. Raw data are first streamed from the producers to the brokers, where data are placed into queues. Each queue is labeled by a "topic tag". The consumers subscribe to the relevant individual queues in order to retrieve the required messages from the brokers.

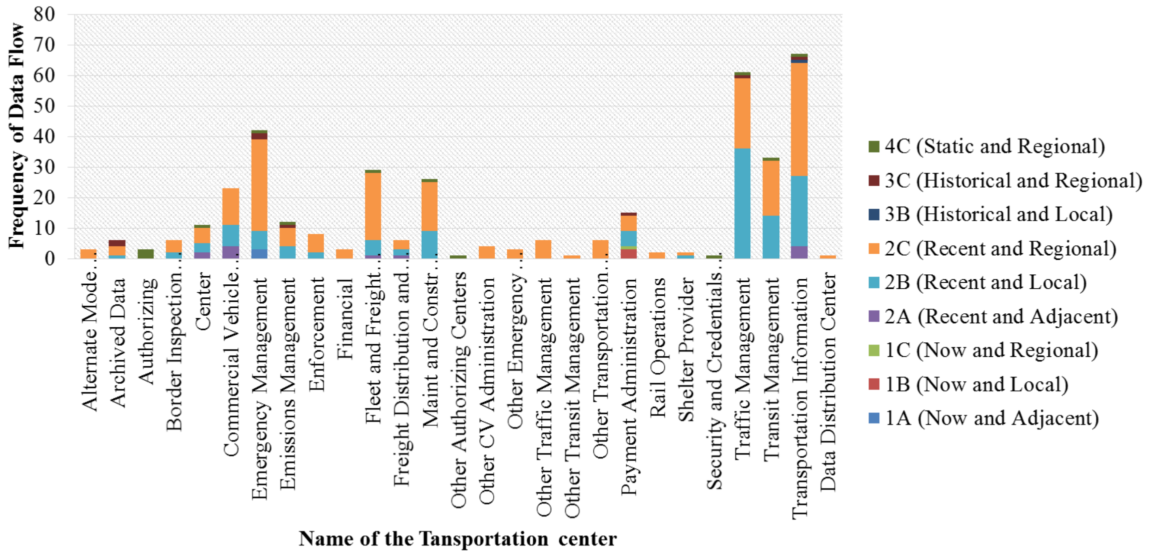

Figure 3.2: Distribution of data flow based on time and spatial context "to center from other physical objects.

The brokers, depending on the needs of the consumers, dynamically create the queues. All entities within a connected vehicle system (i.e. vehicles, traffic management centers, online news, weather, social media sites, and even the CV applications themselves) can be producers or consumers of messages. By using the Kafka brokers as a medium to facilitate data streaming, a distributed message delivery infrastructure can accomplish the following:

- Separation of content and location: Kafka brokers enable consumers to stream relevant data from producers without either party (consumer or producer) having to know about each others location. For example, an emergency management application can ingest a message packet tagged as a crash message without first establishing contact with the source of the message package. In this case, the emergency management application is the message consumer and the accident message source is the message producer. The message consumer and the message producer only talks directly to the brokers while their locations remain hidden with respect to each other.

- Optimization of data management and processing through the broker layer: The computing capability of the Kafka broker cluster enables preliminary curation of raw data before placing them into queues. By placing this responsibility into the Kafka middleware, the traffic man-

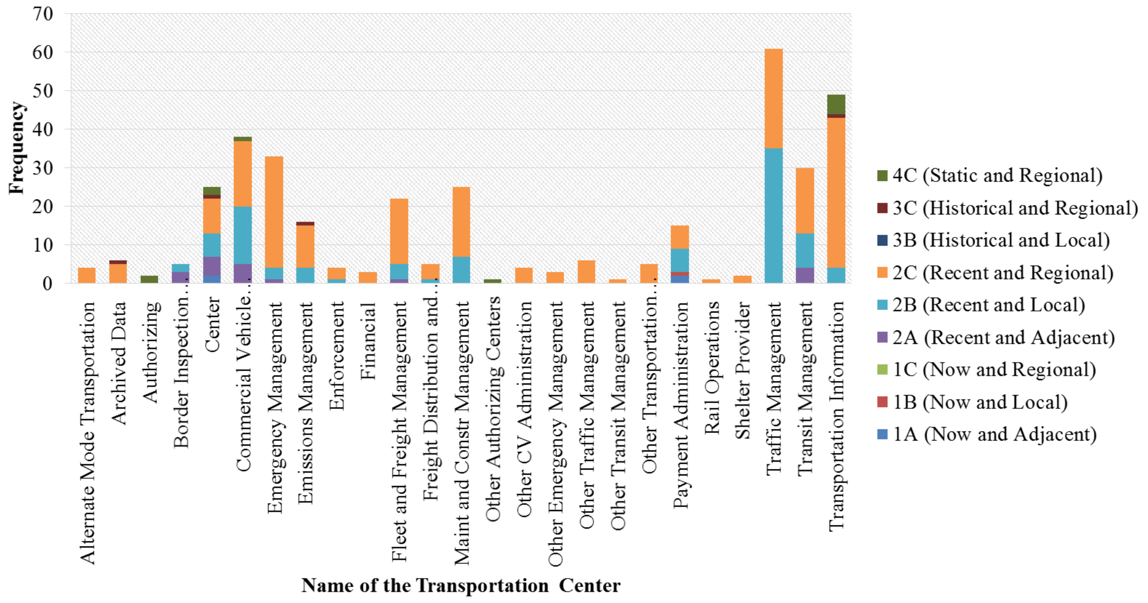

Figure 3.3: Distribution of data flow based on time and spatial context “from center to other physical objects.

agement centers can focus more on analyzing the stored message rather than cleaning the raw data and preparing message into a usable format.

- Dynamic balancing and scaling of message delivery infrastructure: As CV applications have different time requirements for data arrival [11, 12], this requires the applications to only consume data at an appropriate rate. Apache Kafka supports the dynamic addition and removal of new broker software instance into the existing cluster, allowing the infrastructure to scale up during peak demand hours and scale down during periods of reduced demand [54].

- Reduction of administrative and technical responsibilities for data maintenance: In a traditional approach, message is usually accessed directly from the data centers of public Departments of Transportation (DOTs) or similar public agencies. A high level of message redundancy is required not only for backup purposes but also to support large-scale message access (i.e., creating multiple copies of message to be accessed by a large number of users and applications). By placing the burden of facilitating message delivery on a brokerage layer, the DOT centers only need to focus on maintaining data for traditional in-house transportation applications. External applications that require message can have access to the message via the streams of

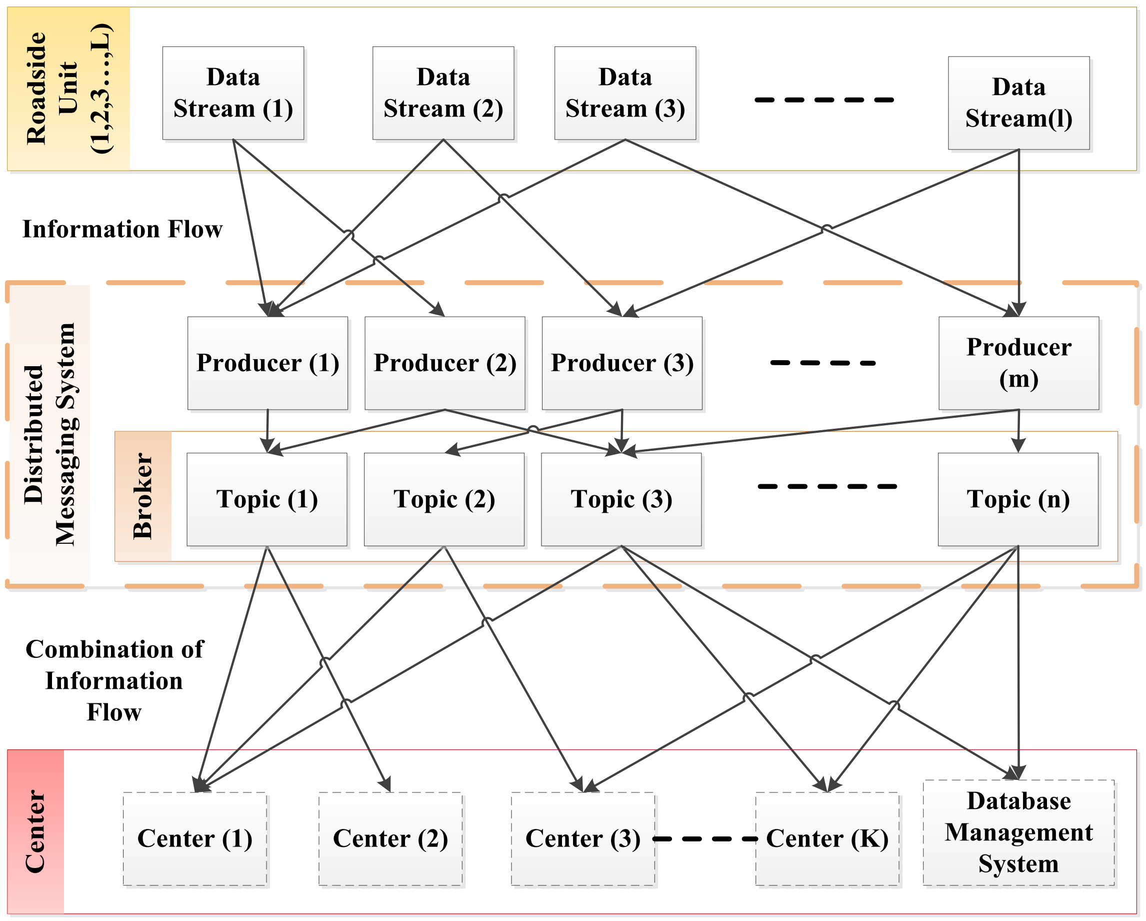

Figure 3.4: Conceptual design of distributed data-centric delivery infrastructure from RSU to different transportation centers.

the broker layer. While it is possible to duplicate these streams for performance purposes, the real-time nature of these streams will prevent the accumulation of message storage duplication.

3.2 Case Study: Evaluation of Distributed Message Delivery System

We design a prototype infrastructure of the distributed message delivery system using Kafka for CV applications. Synthetic data, which were generated using the VISSIM microscopic traffic simulator, were used to evaluate the prototype infrastructure. In the following subsections, we describe the details of the case study.

3.2.1 Synthetic Data Description

Microscopic traffic simulation is an efficient and economical solution for generating synthetic data to evaluate connected vehicle systems of any regional roadway network. We can model a connected vehicle system in a simulated roadway network representing drivers, vehicles, and roadways and perform experiments with different connected vehicle system architectures supporting different applications. Synthetic data generated from simulation experiments include input and output data to each subsystem (i.e., consumers, producers) within the respective message delivery architectures. For example, the vehicle position and velocity messages are transmitted from the vehicle on-board equipment (OBE) to RSU whereas RSU transmits recommended speed information to the vehicle OBEs for real-time speed management of CVs. In this case study, we simulated a roadway network along the I-26 corridor between Columbia and Charleston in South Carolina, which encompassed 91.5 miles of freeway with 19 interchanges. VISSIM (a microscopic traffic simulator) developed by PTV [18] was used to develop a simulated model of the I-26 corridor. According to the Cost Overview for Planning Ideas & Logical Organization Tool (CO-PILOT), a high-level planning tool for connected vehicle pilot deployments, we assume one RSU installation per mile on I-26 corridor [13]. There is an exception to place RSUs at the horizontal curves and the assumption is that two RSUs are required for each curve, one at each end of the curve. For the I-26 corridor, DSRC communication range of 900 ft was used for all the 92 RSUs. Thus, each RSU can collect data, if a vehicle is within RSU coverage (i.e., within the 900 ft radius). In our simulation, each RSU collected CV data for a highway segment of 1800 ft considering 900 ft DSRC coverage in both directions of traffic.

From the VISSIM simulation, we recorded 62 types of data (e.g., speed, position, acceleration, lane number, lane change) for each CV within the DSRC communication range. The CV data are collected using Vehicle Record output option from the VISSIM simulation. We then use these data in our experiments to evaluate the distributed message delivery infrastructure prototype that models RSUs connected to different transportation centers (e.g., traffic management center, transit management center, emergency management center, commercial vehicle administration center, transportation information center). In our connected vehicle system, each CV sends data to RSU, and each RSU sends CV data to the backend transportation centers. These CV data can support different types of CV applications, such as vehicle-to-infrastructure safety (e.g., curve speed warning), mobility (e.g., speed harmonization), and environmental (e.g., variable speed limits for weather-responsive traffic management) applications as depicted in CVRIA [3]. As shown in Figure 3.2 and 3.3, different type of information flows are sent from each RSU to different transportation centers for different CV applications. For this study, we consider only data generated from each CV and was sent to RSU; and data sent from RSU to different transportation centers. In the following, we list all 62 data types collected from the VISSIM simulation. However, our distributed message delivery system can also support other information flows (e.g., roadway sensors data) from other sources (such as roadway traffic sensors) along with vehicle-generated data. In our case study presented in the paper, other sources of traffic related data, such as traditional traffic sensors (e.g., loop detector) and mobile device (cell phone or GPS traces) data, were not considered. It is expected that the substantial penetration of CVs on the roadways will reduce or eliminate the need for traditional traffic sensor data for traffic condition assessment and prediction [62, 61, 21].

3.2.2 Data Infrastructure Experimental Platform Description

Our experimental platform spans two computing platforms based at Clemson University [2]. The first, the Holocron cluster, is a small-scale platform that consists of 20 machines provided to researchers in bare-metal fashion, enabling the installation of any customized software for experimentation with customized cloud infrastructures. In our experiments, we selected 16 machines from Holocron as our Kafka brokers. Each machine has 256 GB DDR4 RAM memories, 2TB 7,200 RPM SATA HDD hard disk and a 10Gbps Ethernet connection for data transfer between machines. This 16-machine broker cluster is the key component for the message delivery system, which comprises Kafka data brokers and supporting services [54]. Therefore, most of the computing resources in the message delivery system reside in this cluster. The brokers handle the incoming message traffic from message producers as well as route outgoing message traffic to message consumers.

The second platform, Palmetto, is a large-scale distributed testbed for scientific research applications consisting of 1,700 machines maintained by the Clemson University Cyberinfrastructure Technology Integration (CITI) group [2]. Since it provides a large number of physical machines, we used machines from the Palmetto cluster as both our data producers and consumers for CV applications. As the Palmetto cluster has over 1000 of the available 1700 machines for scientific research, we allocate a single physical machine from Palmetto for each producer or consumer. This guaranteed better resource isolation, thus ensuring the repeatability and reliability of our experiments. Each Palmetto machine functioned as either a producer or a consumer of the message delivery system by running a specific computer program. For instance, a producer machine runs the data generation program, representing the RSU in the connected vehicle applications. On the other hand, a consumer machine runs the data-ingesting program, representing a traffic management center or a mobile device, which runs certain CV applications that consume data from brokers. Each of the Palmetto machines had 16GB of DDR4 RAM and 100GB 7,200 RPM hard drive and 1Gbps Ethernet connection to the Holocron nodes. The distributed message delivery system for CV applications was then prototyped by setting up a cloud infrastructure between the producers and consumers in Palmetto and the brokers in Holocron. Holocron provides flexible networking among distributed nodes, which facilitated our experiments by assigning the same network configurations on each producer to broker link and each broker to consumer link. Palmetto's integrated batch system, Portable Batch System (PBS) [14], eases the parallel execution of large number of producer and consumer computer programs, which is essential in modeling the case where all consumers are "online".

3.2.3 Experimental Scenarios

To evaluate the performance of the distributed message delivery system, a baseline scenario of message distribution system without any message broker was established. We considered 92 producers in this baseline scenario, all of which were connected with 10 consumers sequentially (as shown in Table 3.1) using a direct Transmission Control Protocol (TCP) connection. We then developed four experimental scenarios to evaluate the distributed message delivery system with different number of brokers and consumers, as presented in Table 3.1. The number of the producers was fixed at 92 (i.e., number of RSUs installed in I-26 corridor). As various numbers of end users affect CV applications during different hours of the day, we varied the number of consumers for each broker size to 10, 20, 30, 40 and 50 to represent different times of the day. The 50-consumer scenario represents the peak hour scenario when most consumers attempt to access data simultaneously, while the 10-consumer scenario represents the off-peak hours when most users are offline. We use different number schemes for our brokers (e.g. 2, 4, 16, 32) to explore how the larger broker clusters benefit the performance of the message delivery system. We allocated the 32 brokers at 16 physical machines in Holocron. Two broker instances were managed simultaneously on each machine.

We examined the message delivery system experimentally considering the capacity of all

Table 3.1: Message Delivery System Evaluation Scenarios

Message Delivery System without Broker (Baseline Scenario)	
Number of Producer	Number of Consumer
92	10

Message Delivery System with Broker			
Experimental Scenario	Number of Producer	Number of Broker	Number of Consumer
Scenario 1	92	2	10,20,30,40,50
Scenario 2	92	4	10,20,30,40,50
Scenario 3	92	16	10,20,30,40,50
Scenario 4	92	32	10,20,30,40,50

producers. For this purpose, each of the 92 producers published 100,000 messages at the same time to overflow the network capacity. These messages were then published to 62 different topics, each representing a single message type as discussed in section 3.2.1. All the messages have a fixed size of 200 bytes. Here, a topic is an abstract container in Kafka that can be seen as a pipeline in which messages flow through. The configuration of each topic in the broker cluster reflects the trade-off between message reliability and latency for all the messages published to that topic. For example, in extreme cases where we generated messages from a certain number of producers to arrive at the brokers at its maximum speed, messages are sent to a topic that requires no acknowledgements to be sent from the brokers to the producers. All of those messages published to that topic thus exhibited the lowest latency.

In our experiment with a 4-node Kafka broker cluster, the average latency for sending one message dropped by 78 percent when switching from a topic, whose messages require acknowledgment ("ack" topic), to a topic whose messages that do not require acknowledgment ("no-ack" topic). However, the reliability of the "no-ack" topic is problematic. About 98.9 percent of all the messages published to the "no-ack" topic is received by the consumers in the 4-broker cluster scenario, which makes this topic unsuitable for the applications that require a highly reliable message transfer (i.e., where 100 percent of messages must be received by the consumers). Since we value the message reliability for our real-time CV applications, we configured all 62 topics in our experiment to be

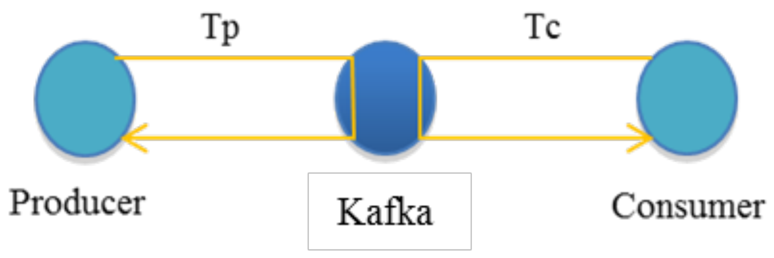

Figure 3.5: Breakdown of an end-to-end latency of a message delivery procedure using the developed message delivery system.

“ack” topics, which required explicit acknowledgment from the broker for each message. The end-to-end latency of a message passing through the message delivery system is comprised of two parts. The first part of this latency included the communication process between the producer and the broker, represented by Tp as shown in Figure 3.5. This Tp was measured from the time at which the message was generated by the producer to that time at which the broker indicated reception of the message.

The second part of the latency encompassed the communication process between the broker and the consumer, which is denoted by Tc (as shown in Figure 3.5). Here, Tc was measured from the time at which the consumer initialized the message-read request to the time at which the consumer successfully received the message. The parallel implementation of broker to consumer message delivery in Kafka makes the latency Tc much lower than Tp. Therefore, at the producer side, it was necessary to reduce the per-message delivery latency to the highest extent to better serve the real-time applications. This latency is associated with the queuing latency of each message in the producers send buffer. The average queuing time of each message at the producer end was proportional to the quotient of buffer size and message size. If the buffer size was too large, the average queuing time of each message was increased. To prevent this queuing delay, we assigned the send buffer size of each of our producers to 400 bytes, twice of the message size. Therefore, whenever a message was loaded into the send buffer, it only had to wait for dispatch of its only predecessor. The small size of the message prevented any overflow in the producer. This approach eliminated the unnecessary queuing delay at the producer end. A total volume of 40 GB messages were generated in the experiment (2 GB for each experimental scenario).

3.3 Data Collection of Automotive Forum and Common Crawl

To evaluate the data curation component of our proposed system, we first describe how the target online forum data and the sample data are collected.

3.3.1 Description of Datasets

We choose a very active car owner online forum as our target forum to evaluate Common Crawl's sample quality. This forum is organized into 14 subforums, each representing a different car model made by a specific car vendor. We refer to these by their car type, e.g., "suv-mid" (a mid-sized SUV), "sedan-full" (a full-sized sedan), "convertible-new" (a newer model of a convertible type), etc. For each subforum, we have collected all available data from Common Crawl as well as the full data from the online forum. Since Common Crawl is based on sampled data, it is not an independent dataset from the original forum data (the full data).

The Common Crawl dataset [36] consists of billions of HTML based web pages that are provided in two formats: WARC and WET. The WARC format contains meta data that describes the crawling process, storage hierarchy, HTTP response codes, and HTML tags. The WARC data is more noisy and hence requires filtering and preprocessing before it can be analyzed with LDA. Alternatively, Common Crawl provides extracted raw text data directly for text mining research called the WET format data. However, the WET format data cannot be used for our LDA experiments, since LDA requires detailed separation of texts from different posts and different threads in an online forum. We therefore used the 14 subforum URLs and gathered 280 GB WARC format data files from the publicly available Common Crawl images in AWS. After data preprocessing, we have grouped all posts in a thread together to form one document.

To collect the full data for each subforum, we have developed a customized web crawler based on Jsoup, a Java library for working with HTML. Jsoup provides an API for manipulating and extracting data, using the best of DOM, CSS, and Jquery-like methods. It can be used to scrape and parse HTML from a URL, file or string. In total, we collected 2.16 TB of raw data. Running the customized crawler on one subforum took, on average, 24 hours. However, a first run of the crawler resulted in the workstation's IP address being blacklisted, which required restarting the data collection with a less aggressive crawling strategy that would decrease the frequency with which the website was accessed. In total, the data collection on the full forum data took over four

Table 3.2: Overview of Online Forums Data Collected from Common Crawl (CC) and Full Data (FD).

Subforum	**Doc. Count CC**	**Doc. Count FD**	**Doc. Frac. (CC/FD)**	**Time-stamp Var.**	**Size of Raw Data (GB) CC**	**Size of Raw Data (GB) FD**
sedan-mid	3,249	27,649	0.12	38	49.4	267.5
sport-new	1,869	16,553	0.11	33	46.2	216.6
convertible	1,521	28,525	0.05	23	29.7	250.5
suv-compact	1,467	10,187	0.14	30	29.6	128.4
suv-mid	1,397	20,217	0.07	21	17.4	145.8
convertible-new	1,181	8,671	0.14	19	18.2	102.2
sport	901	7,765	0.12	54	19.2	74.4
hatchback	703	3,190	0.22	52	17.2	44.0
sport-full	639	17,110	0.04	34	22.8	493.1
sedan-full	436	3,527	0.12	90	5.3	28.1
coupe-compact	212	15,635	0.01	65	11.3	229.2
electric	193	2,177	0.09	181	7.2	38.2
suv-mid-new	139	9,687	0.01	123	4.2	129.0
suv-small	8	851	0.01	1,014	0.2	13.2

weeks, while the data collection from Common Crawl was completed in less than one day, which illustrates another advantage of using the sampled data.

We present an overview of the data collected from Common Crawl (CC) and the full data (FD) in Table 3.2, including the number of documents in each subforum and data set, document fraction in the sample compared to the full data, and standard deviation of timestamp gap (in hours) in each subforum. The latter measures variation in the spread of the data over time. Because Common Crawl data is sampled at irregular time intervals, and because subforums differ in their daily user activity, there are differences in the way the data is spread over time between subforums. We capture this variation by first ordering the timestamps of posts in a subforum, then calculating the standard deviation of the intervals between consecutive timestamps of posts.

3.4 Online Automotive Forum Text Data Curation

After the streaming text data are collected from the online forum, we use the Latent Dirichlet Allocation (LDA) language model to discover the discussion themes contained in the corpus. To evaluate the sample bias in drawing these themes, we introduce both a quantitative approach and a qualitative approach for comparing the topics drawn from the samples to the topics drawn from the full dataset.

3.4.1 Description of LDA Topic Modeling

Topic modeling approach uses unsupervised machine learning method for extracting latent topics form a large set of documents. It is often applied in text mining to facilitate tasks like information retrieval, text classification and sentiment analysis. Topic modeling models documents by using a generative statistical model that takes each document as a mixture of a fixed number of topics. A representative model used in many topic modeling research is the Latent Dirichlet Allocation (LDA) model [29].

In topic modeling, a corpus is a set of documents while each document is indicated as a sequence of words. The key idea in topic modeling is adding a layer of hidden topics between words and documents by assuming a generative process between topics and documents.

More formally, LDA uses a three layer Bayesian model to fit the generative process. In the top layer, each document is represented by a finite mixture over a set of latent topics. In the middle layer, each latent topic is represented by a distribution over words in the entire corpus. Finally, the bottom layer consists of the words in a corpus, which are the basic elements of the generative process. Therefore, a topic's meaning is best conceived by looking at its word rankings.

When applying LDA to the user forums we are analyzing, each thread is assumed to be one document. The number of threads from the Common Crawl Bimmerpost forum data being analyzed in our experiment is 21,247. These threads come from 14 distinct subforums with each subforum symbols one specific car model. In our experiment, we treat 14 subforums independently when performing LDA topics analysis, as different subforums have various data characteristics in terms of topic modeling.

In the insight of forum user behavior section, we categorized the 14 car models into 5 classes, convertible, electric vehicle, sports car, sedan and SUV. We are interested to see how the users' discussions about each of these car classes are different from each other by observing and comparing the topics across different car classes. Therefore, we regroup the documents from the 14 sub forums into 5 segments and performed LDA individually.

When performing LDA on the 4 car classes, we trimmed the words in our preliminary segments using the following strategy. The words that appear less than 5 times in each segment or in fewer than 5 documents in that segment are removed. Removing the words that appear less than 5 times in each segment helps filter out the low frequency words and focus on the core words in the vocabularies. This trimming strategy and threshold is commonly seen in text mining researches that use topic modeling [31]. Removing the words that appear in fewer than 5 documents helps discard the potential off-topic discussions about weather, social events, etc. After word trimming, the vocabulary size shrinks down by 75 percent on average in each segment.

Latent Dirichlet Allocation (LDA) [29] is a generative model that estimates latent groups ("topics") from a corpus. Its main assumption is that documents are random mixtures of corpus-wide topics, where each topic is a probability distribution over the entire vocabulary. Following the notation in [29], the generative process of LDA for each document w in a corpus D is described as follows:

1. Choose $N \sim \text{Poisson}(\xi)$.
2. Choose $\theta \sim \text{Dir}(\alpha)$.
3. For each of the N words w_n:
    (a) Choose a topic $z_n \sim \text{Multinomial}(\theta)$.
    (b) Choose a word w_n from $p(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic z_n.

Where N is the number of works in a document drawn from a possion distribution. θ is the topic distribution over a document sampled from a Dirichlet distribution $\text{Dir}(\alpha)$. A key output of LDA is an estimate of each document's topic proportions, which can be used to calculate the proportion of each topic in the entire corpus. We denote these global topic mixtures as $M = \{m_1, \ldots, m_k\}$, where k is the number of topics, and $\sum_1^k m_i = 1$.

For all of the following analysis, we set $k = 10$ and the Dirichlet parameter $\alpha = 0.1$. An exception is the "suv-small" subforum, where we set $k = 5$ because of the small number of documents. We use the original C code provided by [29] available at [39]. On the largest data set in our analysis ("sedan-mid" with 27,649 documents in the full data), the algorithm converges after about eight hours.

3.4.2 Description of Similarity Comparison Between Topics

Our goal is to compare topics estimated from Common Crawl to those estimated on the full data. A common metric for evaluating a topic model's quality is perplexity, which is a measure of a model's predictive likelihood calculated from a held-out set [90]. Perplexity is not a suitable measure in our case because we need to evaluate the similarity of two models estimated on different data rather than comparing their performance on the same data set. We instead use two metrics that capture both the quantitative and qualitative similarity between two topic models. First, we compare topic mixtures estimated from the sampled and full data, using the Kolmogorov–Smirnov (KS) test. Secondly, we compare the top-ranked words in different topics using the Sørensen–Dice coefficient. In this section, we motivate and explain both measures in more detail.

3.4.2.1 Evaluating Topic Proportion Similarity

A key output of LDA is an estimate of the proportion that each latent topic is represented in the corpus, which we denote as M (see Section 3.4.1). An estimate of 0.25 for a topic, for example, means that 25% of the text in a corpus is estimated to fall under this topic. These mixtures are important measures for business analysts because they provide insights into the relative importance of estimated topics.

To give an example, consider the following two topic mixtures estimated for the "suv-compact" forum from Common Crawl (CC) and the full data (FD):

$$M_{CC} = \{0.16, 0.15, 0.11, 0.11, 0.11, 0.09, 0.09, 0.08, 0.06, 0.05\}$$
$$M_{FD} = \{0.16, 0.13, 0.11, 0.11, 0.11, 0.09, 0.09, 0.08, 0.07, 0.07\}$$

The topics are sorted from largest to smallest topic for both data sets. This comparison only examines topic mixtures; the semantic alignment of the topics is captured by the other comparison using the Dice coefficient. Based on these mixtures, the two models produce very similar results.

We formally evaluate topic mixture similarity with a two-sample KS test. Let $F_{CC}(x)$ and $F_{FD}(x)$ denote the empirical distribution functions calculated from M_{CC} and M_{FD}, respectively. The two-sample KS test statistics D is then calculated as

$$D = \sup_x |F_{CC}(x) - F_{FD}(x)|. \tag{3.1}$$

It is a test of the null hypothesis that $F_{CC}(x)$ and $F_{FD}(x)$ come from the same distribution.

For the above example, $D = 0.2$ (the largest absolute difference between topic proportions) with $p = 0.99$. Because p is clearly above the standard 0.05 threshold, we cannot reject the null hypothesis that the topic proportions are drawn from the same distribution. While this result does not prove that the two mixtures are the same, there is no statistical evidence that they are different from each other.

The KS test statistic is calculated by matching topics based on their rank. That is, one compares the two largest topic proportions with each other, followed by the second largest proportion, etc. This does not take into account that topics with similar proportions may differ qualitatively, i.e., in terms of their top ranked words that define the topics. We therefore introduce a measure that compares topics qualitatively.

3.4.2.2 Evaluating Topic Meaning Similarity

Topics estimated with LDA are probability distributions over the vocabulary. It is the analyst's job to label the topics, that is, to decide their substantive meaning. This is usually done by sorting the vocabulary by their estimated topic-word probabilities and looking at the top k words, where typical values for k are in the range of 5–20. To provide an example, Table 4.4 in Section 4.3, which we will discuss in greater detail below, shows the top ten keywords for the two largest topics estimated from Common Crawl for four selected forums.

For Common Crawl to be a useful sample of the population data, it should produce topics that are substantively similar to those estimated on the full data. We evaluate this criteria with a metric that mimics human evaluation of topic similarity, which would be based on comparing the top keywords of two topics and judging their similarity in terms of the words they include. More precisely, we follow an approach suggested in [44] that uses the Sørensen–Dice coefficient to measure the overlap between two keyword lists. Let X denote the set of k top keywords from a topic estimated from Common Crawl, and let Y denote keywords estimated from the full data. The Sørensen–Dice coefficient (or short, Dice coefficient) is calculated as

$$D(X, Y) = \frac{2|X \cap Y|}{|X| + |Y|}, \tag{3.2}$$

where $X \cap Y$ is the set of common words from both word lists, and $|X|$ and $|Y|$ are the numbers of words in each list.

To provide an example, consider the following two word lists:
$X = \{\text{looks, sport, interior, trim, wheels}\}$ and $Y = \{\text{wheels, trim, color, price, sport}\}$ (these are actual words that appear in the "suv-compact" forum). Each set includes five words, and they share three elements, "sport", "trim" and "wheels". The Dice coefficient for this example is $D(X, Y) = \frac{2\times3}{5+5} = 0.6$. If the two word lists were identical, the Dice coefficient would be 1; if their intersection were empty, the coefficient would be 0. In our calculations below, we will use the same number of the top twenty keywords in each set. The Dice coefficient is then equivalent to the proportion of words that appear in both topics.

The Dice coefficient is a comparison between two topics. When comparing topics from two models estimated on different data sets, we need to assign each topic from one model to a topic from the other model in order to calculate the coefficient. This is a classic matching problem for which well-known solutions exist. We here follow an approach suggested in [44] for this particular case. Because we expect that each topic estimated on the full data has a corresponding topic in the sample, we greedily match topics to each others based on the maximum Dice value. More precisely, for two equally-sized sets of topics, we first match the topic pair with the highest Dice coefficient, then repeat this process with the unassigned topics until all topics are matched. Our measure of model similarity is then the average Dice coefficient over all selected topic pairs.

3.5 Query Expansion of Streaming Social Media Data

In this section, we introduce our design of an automated query expansion system that detects and monitors emergent events in Twitter streams.

3.5.1 Query Expansion in Social Media Data System Overview

The overview of our proposed query expansion component is described in the following algorithm: In order to take a large corpora of documents and relate them in a conceptual space, FastText from Facebook's research group was used. FastText uses subword information to enrich its vector space, so that even stemmed or misspelled words are close to each other in conceptual distance [30]. With calculating emergent events from a document stream such as Twitter, Dynamic Eigenvector Centrality (DEC) captures not only the importance of words in a stream for a given window, but how their centrality has increased (or decreased) in a number of windows prior [22].

Algorithm 1 System Overview

```
 1: $S_m$, primary Tweet stream at window m
 2: $js$, Jaccard Similarity between top DEC words, 3 windows apart
 3: procedure AutomatedQueryExpansion($S_m, w$)
 4:     Calculate DEC values on $S_m$ every $w$ minutes.
 5:     $js \leftarrow$ Calculate Jaccard Similarity between top DEC words, 3 windows apart
 6:     if $js \leq 15$ then
 7:         Construct Topic Model on current window of $S_m$
 8:         Expand Topic Model with Vector Space to make new Query
 9:     end if
10:     $W_t$, query expansion keywords from local analysis
11:     $W_v$, query expansion keywords from global analysis
12:     Collect Tweets from $S_m$ using $W_t$ and $W_v$
13: end procedure
```

To calculate when to expand a query, we determine the jaccard similarity between the top 200 DEC keywords between two windows that are 45 minutes apart. When this window is below a certain threshold (for our purposes, 15 percent), we expand a query using the given algorithm.

To experiment on our automated query generation approach, we studies 4 different methods of query generation.

Q1: Static Stream. As a baseline scenario, we use a static stream which is generated by using one fixed keyword “police” as query condition.

Q2: Query Expansion Stream Using Topic Modeling Keywords. The first proposed approach for query expansion is to use Topic Modeling (LDA model) keywords. LDA topic modeling produces n topics with each topic represented by a distribution of the words in the vocabulary. The distribution of words reveals the discussion theme associated with the topic. The algorithm is described as follows: To generate query Q, we give the LDA topic modeling output T, and a stream of Tweets S. T consists of n topics each representing one discussion theme. In a given Tweet s, if we can find any combination of two keywords in topic t, then we add Tweet s into query result q_t corresponding to topic t. Finally the procedure returns the aggregated query result for all topics, Q.

Q3: Query Expansion Stream Using Topic Modeling Keywords and Dynamic Eigenvector Centralities As a comparison to Query 2, we introduce another query which combines Topic Modeling keywords and Dynamic Eigenvector Centralities (DEC) keywords to generate the query condition. This procedure is described in Algorithm 2. For each of the n topic t, we also add another condition in the query so that it needs to include the DEC keyword that is furthest apart

Algorithm 2 Query Expansion Using Topic Modeling Keywords

```
1: $Q$, query result
2: $S$, primary Tweet stream
3: $s$, a Tweet in the primary Tweet stream
4: $T$, a set of topics result from LDA topic modeling
5: $t$, a given topic in $T$
6: $q_t$, query result associated with a given topic t
7: procedure QUERY EXPANSION($Q, T, S$)
8:     $Q \leftarrow \emptyset$
9:     for $t$ in $T$ do
10:        $q_t \leftarrow \emptyset$
11:        for $s$ in $S$ do
12:            $(w_i, w_j)$, a pair of words from the same Tweet
13:            for $(w_i, w_j)$ in $s$ do
14:                if $w_i \cap w_j$ in $t$ then
15:                    $q_t \leftarrow q_t \cup s$
16:                end if
17:            end for
18:        end for
19:        $Q \leftarrow Q \cup q_t$
20:    end for
21:    return $Q$
22: end procedure
```

Algorithm 3 Query Expansion Using Topic Modeling Keywords and DEC keywords

```
1: $Q$, query result
2: $T$, a set of topics result from LDA topic modeling
3: $t$, a given topic in $T$
4: $q_t$, query result associated with a given topic t
5: $V$, historical semantic vector space from word embeddings
6: $S$, primary Tweet stream
7: $s$, a Tweet in the primary Tweet stream
8: $Dec_w$, word with top DEC value that have furthest semantic distance from a set of LDA keywords
9: $(w_i, w_j)$, a pair of words from the same Tweet
10: procedure QUERY EXPANSION($Q, T, V, S$)
11:    $Q \leftarrow \emptyset$
12:    for $t$ in $T$ do
13:        $q_t \leftarrow \emptyset$
14:        $Dec_w \leftarrow furthestDec(V, q_t)$
15:        for $s$ in $S$ do
16:            for $(w_i, w_j)$ in $s$ do
17:                if $w_i$ is $Dec_w \wedge w_j$ in $t$ then
18:                    $q_t \leftarrow q_t \cup s$
19:                end if
20:            end for
21:        end for
22:        $Q \leftarrow Q \cup q_t$
23:    end for
24:    return $Q$
25: end procedure
```

Table 3.3: Characteristics of Proposed Query Generation Methods

	Static	**Themes**	**Emergent**	**Historical**
Q1	Yes	No	No	No
Q2	Yes	Yes	No	No
Q3	Yes	Yes	Yes	No
Q4	No	No	Yes	Yes

from the set of keywords in topic t in the vector space V. This will ensure that the topic is most emerging get represented in the query result. In the above procedure, function $futhestDec(V, q_t)$ find the DEC word that has the most average Euclidean distance from the keywords in topic q_t and save in in Dec_w. We use this combination because normally the top DEC keyword is contained in the top keywords in the LDA topic modeling results. Therefore to ensure we can create a new topic that contains DEC keyword, we select the DEC word that has the most distance in the semantic vector space to the LDA topic keywords. The semantic vector space is generated using the Tweets collected during a two week time window around the target event.

Q4: Query Expansion Stream Using Topic Modeling Keywords and Vector Space. The fourth proposed algorithm is a method for expanding a generic topic model (in our case, a parallel LDA model) with an external vector space to add to the topics, making novel queries). As mentioned, our vector space is trained on the past year of news and press releases, and is represented with V. Using this vector space, a nearest neighbor efficiently calculates those words closest in space to a given word. Thus, we expand the query by taking a topic t from a topic model, find a set of neighboring $neighbors$ that are closest to t in the vector space to the topic. The resulting set of keywords included in the query is $neighbors$. The idea behind this algorithm is that by brining the external vector space, we can add new keywords of historical information about the query intent which is not included in the vocabulary of the available Tweets stream.

The key goal of our query generation method are to combine keywords relation information from various semantic levels to provide the best quality query suggestion. To reach this goal we perform different natural language modeling techniques to provide semantic information of words at such heterogeneous levels. LDA topic modeling uses a generative model to give the topical (whether a set of words are associated with the same underneath discussion theme) information of keywords. Dynamic Eigenvector Centralities(DEC) uses a semantic graph model to indicates the level of emergence of a keyword at a given time. The word embedding model uses a neural network

Algorithm 4 Query Expansion Using Topic Modeling Keywords and Vector Space

```
1: $Q$, query result
2: $T$, a set of topics result from LDA topic modeling
3: $t$, a given topic in $T$
4: $q_t$, query result associated with a given topic t
5: $V$, historical semantic vector space from word embeddings
6: $S$, primary Tweet stream
7: $s$, a Tweet in the primary Tweet stream
8: $neighbors$, neighboring words in historical semantic space $V$ from a LDA topic $t$
9: $nearest$, a set of neighboring words in historical semantic space from word $w_t$
10: $(w_i, w_j)$, a pair of words from the same Tweet
11: procedure QUERY EXPANSION($Q, T, V, S$)
12:     $Q \leftarrow \emptyset$
13:     $neighbors \leftarrow \emptyset$
14:     for $t$ in $T$ do
15:         $q_t \leftarrow \emptyset$
16:         for $w_t$ in $t$ do
17:             $nearest \leftarrow$ nearestNeighbors($V$, $w_t$)
18:             for $newWord$ in $nearest$ do
19:                 if $newWord$ not in $t$ then
20:                     $neighbors \leftarrow neighbors \cup newWord$
21:                 end if
22:             end for
23:         end for
24:         for $s$ in $S$ do
25:             for $(w_i, w_j)$ in $s$ do
26:                 if $w_i \wedge w_j$ in $neighbors$ then
27:                     $q_t \leftarrow q_t \cup s$
28:                 end if
29:             end for
30:         end for
31:         $Q \leftarrow Q \cup q_t$
32:     end for
33:     return $Q$
34: end procedure
```

model to minimize the probability of neighboring words in a corpus given a certain word [72, 77, 53]. Each word are represented by a dense vector called word embeddings that represents a word using lower dimensional vector than one-hot representation [43]. It learns a semantic vector space to describe the semantic distance between words, therefore serves as a global thesaurus to indicate the semantic similarity between words. The word embedding model are built on preexisting corpus and hence utilizes historical knowledge of the word-to-word relationship.

Table 3.3 summarizes the key differences between the 4 query generation method described above. For static stream (Q1) it doesn't adopt any word semantic information except the static keywords used for query. For Q2, the theme relationship between keywords are used since topic modeling keywords are the candidates for query generation. For Q3, both DEC keywords and topic modeling keywords are used as candidates, hence semantic information in both theme and emergent level are included. For Q4, the candidate keywords are a mixture of nearest neighbors obtained from vector space model and DEC keywords. Hence the semantic information in historical level and emergent level are used.

For each Tweet in the primary stream, if it contains more than a threshold of n keywords in a given query method (Q1, Q2, Q3, Q4), then we add that Tweet into the result set associated with that query. The result set may contain many subset, with each subset of Tweets corresponding to query result associated with a LDA topic. We analyze the quality of queries at subset level, meaning that we aggregate the query result associated with each LDA topic. The experiments and result discussion sections discuss this matter in details.

3.5.2 Query Expansion in Social Media Data Experiment Set up

We ran two experiments to compare these proposed tools for query expansion: **(1)** Tweets collected from a topic (”police”) on the raw stream, and **(2)** Topic models with vector space words.

Raw Stream. To test our system, we began with a stream of 20.5 million Tweets that cover the timeline of the 2015 Baltimore protests, when Freddie Gray, a young African-American Baltimore citizen, was killed from a police altercation. While this stream was not a full "firehose" of the Twitter social media stream, it was collected with sufficient noise for it to be treated as such.

Static Stream. For a baseline query approach, we took queried the Raw Stream for all Tweets containing the word ”police,” for a total of 5.1 million Tweets.From these Tweets, we calculated Dynamic Eigenvector Centralities on the stream in 15-minute intervals. When we observed a 15-percent Jaccard similarity (or smaller) between windows three intervals apart, we began a new query expansion.

Expansion Query Stream Using Topic Modeling Keywords To generate the query expansion terms for Query 2 (discussed in section III), we applied Latent Dirichlet Allocation (LDA) topic modeling on the intervals that contains emergent event as identified in Static Stream. For each interval, we use 5 topics and take the top 20 keywords from each topic to form the top-ranked keywords to expand the original query. For each topic, the Tweets that contain at least two top LDA keywords from that topic are added to the expanded query stream associated with that topic.

Chapter 4

Results

In this chapter, we discuss the results we obtained from experiment setups for each of the three components we mentioned in Chapter 3. In Section 4.1, we focus on the latency and throughput performance for our distributed data delivery component. In Section 4.2, we use both quantitative and qualitative metrics to evaluate the performance of our data curation component. In Section 4.3 we present our observations in terms of business insights using our data curation model on the online automotive forums. In Section 4.4, we analyze the query results using metrics that measuring the efficacy of our proposed queries at different semantic levels.

4.1 Distributed Data Delivery System Experiment Results and Analysis

For the baseline scenario, we consider 92 producers (i.e., RSUs), where each producer was sending 50,000 messages to 10 consumers sequentially using a direct TCP connection. Each consumer opened a TCP socket and listened for connections from the producers, and each producer made a socket connection for every message prior to transmission. Each message was 200 bytes in size with 62 types of CV data (see section 3.2.1) to simulate the baseline scenario and distributed message delivery system. We used two Kafka brokers in our comparison of the baseline scenario, which is the minimum system requirement to avoid single point machine failure, for the distributed message delivery system. The average end-to-end and the maximum end-to-end message delivery latency for the baseline scenario (without any brokers) were 12 milliseconds and 3751 milliseconds, respectively, which were significantly greater than the corresponding latencies with a distributed message delivery system using two Kafka brokers (as shown in Table 4.1). This difference is mainly due to the smaller overhead of the distributed message delivery system when serving multiple consumers compared to the baseline scenario. For serving multiple consumers, the Kafka distributed message delivery system handles all incoming messages in parallel, where in the baseline scenario only one message is handled at a time between a producer and a consumer.

The brokers cached message pool then processes multiple message deliveries to different consumers simultaneously. Our analyses for distributed message delivery system with four different experimental scenarios are presented in Table III and Figure 4.1, 4.2, 4.3 and 4.4. In Table 4.2, we presented average latency between producer and broker Tp and average latency between broker and consumer Tc in addition to total average latency for different experimental scenarios. Note the smaller Tc compared to Tp, because of the efficient implementation of the message delivery function in Apache Kafka. When a consumer needs to read messages from the brokers, the Kafka brokers use the Linux sendfile Application Programming Interface (API) to transfer messages to the consumer socket without further buffering or copying the messages. This reduces the transmission latency, and therefore affects Tc. Moreover, when consumers simultaneously read a large message set, Kafka batched the messages in groups to leverage the receiving buffer at the consumer end in order to improve the latency performance. The data collected from our experiment regarding the total average latency of each message for the distributed message delivery system with varying number of consumer are shown in Figure 4.1. For real-time CV applications, latency is the most important performance factor of a delivery system. An increase in the number of brokers (see Figure 4.1) causes a decrease in average end-to-end latency, a trend that is more obvious at peak hours when additional consumers utilize most of the processing power of the broker cluster. Therefore, a larger broker cluster must be available during peak hours, since the increased processing power is amortized by the added workload from consumer requests. Even during these peak hours, the use of two brokers for our experimental scenario (considering 50 consumers) will give an approximate end-to-end latency average of 7.95 milliseconds, providing ample room for a stream processing latency to meet all the time and spatial contexts requirements for real-time applications specified in CVRIA [3]. Increasing the number of brokers in the broker cluster makes this latency even more insignificant. However, increasing the number of brokers provides the most for the applications with a large number of users

Table 4.1: Comparison of the Message Delivery Latency between the Baseline and Distributed Message Delivery System

Experimental Scenario (92 producers and 10 consumer)	**Average Latency** (ms)	**Maximum Latency** (ms)
Baseline (without broker)	12.00	3751
Distributed message delivery system with broker (2 brokers)	2.03	1463

at peak hours; otherwise, the performance gain is minimal, as shown in the case of the 10 consumers.

Another essential metric, in addition to latency, is producer and consumer throughput. Throughput can be defined as the number of records sent or received per second. The throughput at the producer end (as shown in Figure 4.2) helps establish the upper limit of RSUs report frequency, which is the maximum velocity of message producing events that the delivery system can handle if no message buffering occurs. The y-axis shows the MB/s throughput for each of the 92 producers. Even though the total message volume of the 92 producers overflows the network bandwidth, the actual accumulated throughput of each producer is far less than network bandwidth. The handshake time between the producers and the brokers, which slowed the message transfer and reduced the throughput could be the cause of this reduction. During the off peak hours, the best throughput was 0.35MB/s for each producer when using 32 brokers, creating a 1500 record-per-second limit for processing message flow from either RSUs or CVs. The number of consumers less likely affects the improvement in a producers throughput when the numbers of brokers increases, as shown in Figure 4.2. Thus, increasing the number of brokers appeared to exhibit a similar throughput improvement pattern among all consumer cases, regardless of peak or off-peak hours.

While in latency case as shown in Figure 4.1, higher performance improvements are observed when the number of consumers are large (peak hours). Figure 4.3 shows the throughput of consumers with different number of broker and consumer scenarios. The throughput of each consumer is much larger than the throughput of each producer. This discrepancy is due to Kafkas efficient implementation of consumer read process [54]. For the 32-broker scenario, the per-consumer throughput dropped linearly as the number of consumers increased from 10 to 50, indicating that the accumulated consumer throughput was approximate to the network bandwidth limit. For other

Table 4.2: Tp and Tc for Distributed Message Delivery System in Different Experimental Scenarios

Number of Consumers	**Average Tp (ms)**	**Average Tc (ms)**	**Average Total Latency (ms)**
Experimental Scenario 1: **Number of Producers: 92 Number of Brokers: 2**			
10	1.55	0.48	2.03
20	3.19	0.83	4.02
30	4.23	1.06	5.29
40	5.42	1.34	6.76
50	5.97	1.98	7.95
Experimental Scenario 2: **Number of Producers: 92 Number of Brokers: 4**			
10	1.36	0.31	1.67
20	2.23	0.49	2.72
30	3.20	0.72	3.92
40	3.97	0.89	4.86
50	5.02	1.16	6.18
Experimental Scenario 3: **Number of Producers: 92 Number of Brokers:16**			
10	1.21	0.28	1.49
20	1.59	0.32	1.91
30	2.03	0.44	2.47
40	2.54	0.49	3.03
50	3.10	0.61	3.71
Experimental Scenario 4: **Number of Producers: 92 Number of Brokers: 32**			
10	1.18	0.26	1.41
20	1.80	0.29	2.09
30	2.24	0.36	2.60
40	2.73	0.43	3.16
50	3.14	0.51	3.65

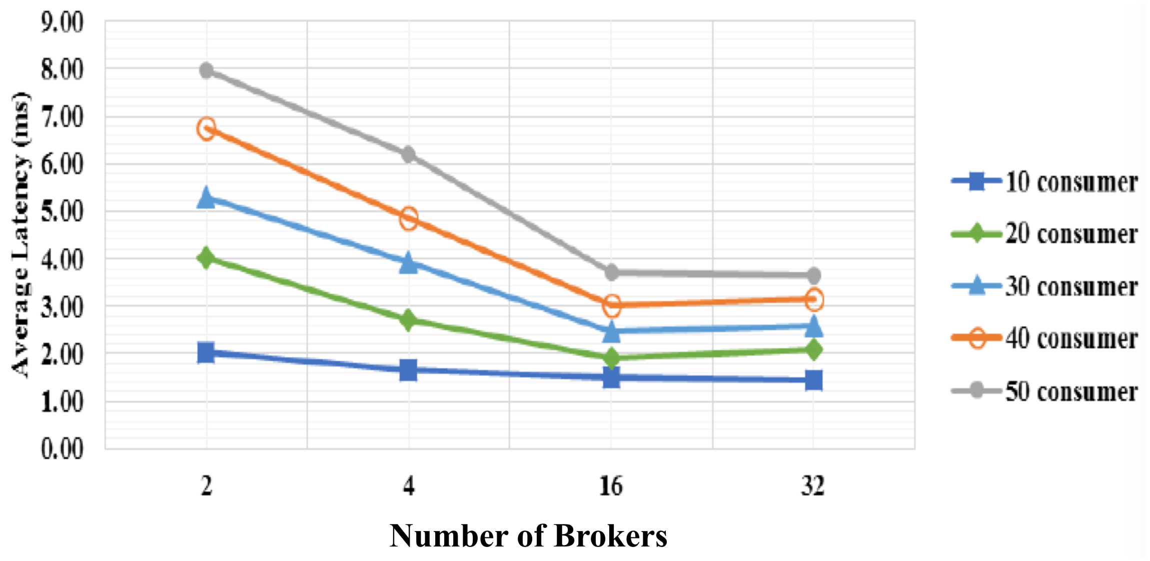

Figure 4.1: Total average latency for the distributed message delivery system by varying the number of consumers and brokers.

cases, the consumer throughput was satisfactory even with low numbers of broker clusters, indicating that the improved performance in consumer throughput was insensitive to the broker cluster size.

In most real-time application, consumers cannot receive large batches of data simultaneously. Therefore, when the number of brokers increases, the percentage of latency reduction is higher than the percentage of increment in consumer throughput. Specifically as shown in Figures 4.1, 4.2, 4.3, it is desirable to use a larger broker cluster for CV applications managing many consumers at peak hours, while a broker cluster size between 4 or 16 for applications with few consumers ensures performance without additional expense on the larger cluster. We did however notice a significantly higher maximum latency than average latency in all of our experimental scenarios. Note the end-to-end latency distribution of the 16-broker system scenario shown in Figure 4.4.

Here, the average latencies for all consumer cases were less than 5 milliseconds with the 99th percentile latencies, which is still less than 12 milliseconds while the maximum latencies jumped to 500 milliseconds. Although there was a less than 1 percent occurrence of these maximum latencies, they nonetheless determined the degree to which a delivery system served a real-time application. This discrepancy may be due to a slow File Input/Output (I/O) thread, a networking congestion

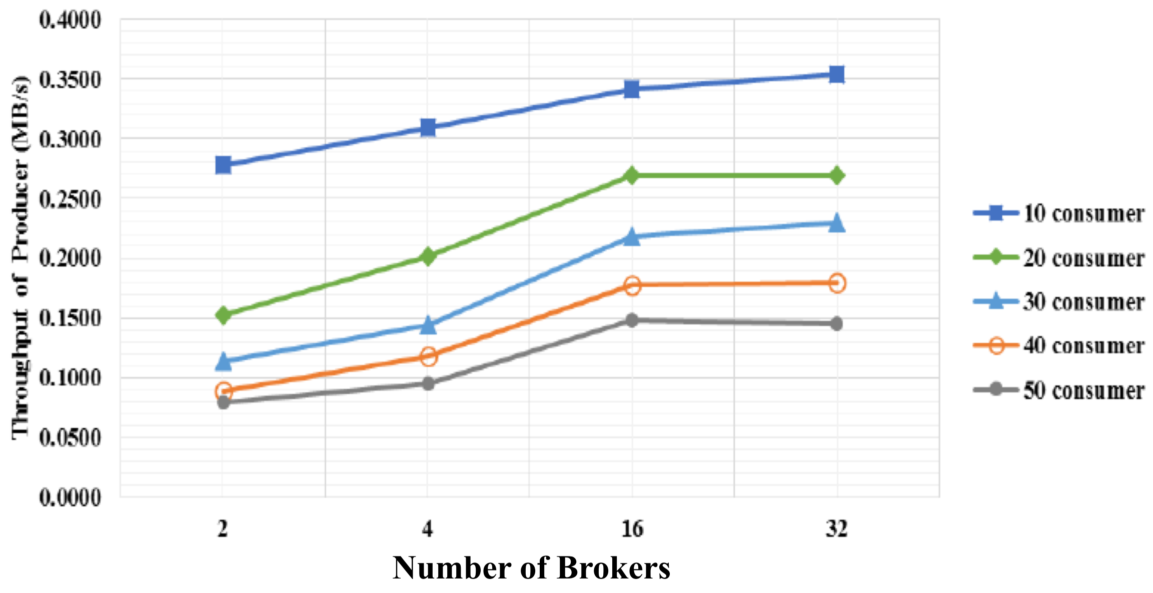

Figure 4.2: Throughput of producer for the distributed message delivery system by varying the number of consumers and brokers.

error, broker unavailability, or a slow partition of a topic. Determining the precise origin of this long maximum latency problem is difficult due to the lack of fine-grained tools to monitor such behaviors. It is also difficult to determine the correlation between these minor events and their root causes. Although it is possible to send multiple duplicate messages through different producers simultaneously to avoid the maximum latency ceiling of sending each message, a message redundancy will occur in the brokers with this strategy. Therefore, we will conduct further studies to determine the possible causes of this long maximum latency problem to derive a solution.

USDOT's system requirements of Intelligent Network Flow Optimization (INFLO) Prototype for CV Dynamic Mobility Applications (DMA) state that the Traffic Management Entity (TME) shall have the capability to obtain data from the traffic sensors on every 20 seconds [15]. The highest total average end-to-end latency is 7.95 milliseconds in our experiments. These latencies were much lower than the recommended values for the USDOT system requirements for CV pilot deployments. However, the minimum system requirement, such as the minimum number of brokers, is decided by the message delivery workload for the Kafka message delivery system, e.g., how many producers sending messages at the same time and how many consumers are reading those messages

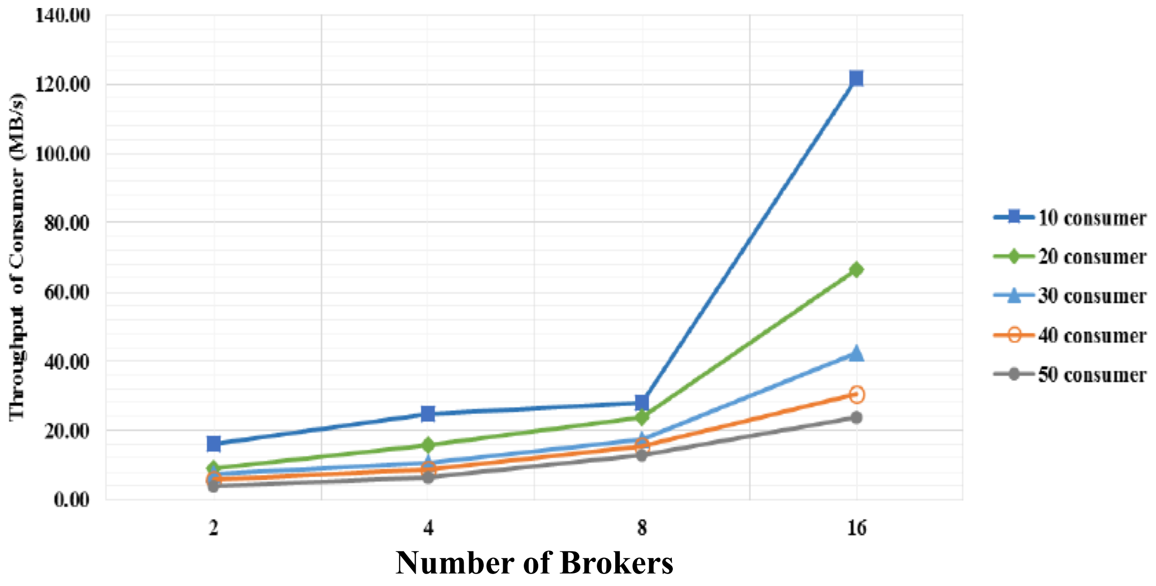

Figure 4.3: Throughput of different number of consumers for the distributed message delivery system by varying the number of brokers.

at the same time. The strength of our distributed message delivery system is that the delivery capability of the brokering system is dynamic and adaptable to workload in run time. For example, we can always start with the minimal number of Kafka brokers (i.e., 2) and add up the number of broker as needed when the latency performance rises above the USDOT requirement. Our experiments demonstrated that we could satisfy the USDOT latency requirement for different CV applications with the least number of machines (i.e., 2). In addition, our experiments also demonstrated that the latency performance improves further when we add more number of brokers under experimental scenarios 2, 3 and 4 (as shown in Table 4.2 and Figure 4.1). Also, we found that in moving the design of ITS data infrastructure from a vertical, top-down approach as shown in [56] (Figure 4.5) to a horizontal, one-level approach (Figure 4.6) using publish/subscribe streaming message delivery model improved the flexibility, scalability, and resiliency of the entire data infrastructure. Publish and subscribe streaming messages represent sending and receiving of streaming messages, respectively. Moreover, our new approach simultaneously maintains the different storage components required at different scales of operation, and significantly reduces the level of dependency among the different storage components. As connected vehicle systems can support multiple applications simultaneously

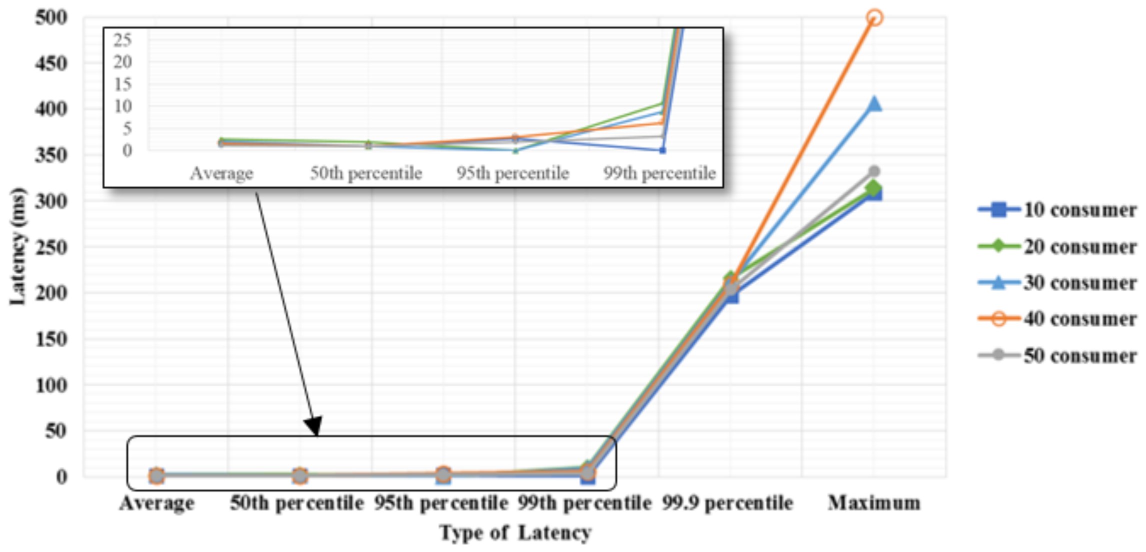

Figure 4.4: The latency distribution for the distributed message delivery system with 16 brokers by varying the number of consumers.

(such as multiple safety, mobility, environmental and energy applications could be supported by the same CV infrastructure on a roadway) and public investments will only include infrastructure investments (such as investments in roadside units and backend computing infrastructure), connected vehicle systems can potentially provide significant economic benefits compared to its cost [47].

4.2 Results of Data Curation using Topic Modeling on Sampled Dataset

In this section, we analyze the representativeness of topics estimated from Common Crawl compared to those estimated on the full data. We first conduct a comparison between LDA topic proportions between the two data sets, showing that there is no statistical evidence that the topic proportions are not drawn from the same distribution. We then analyze similarity in terms of word ranking, demonstrating that the average Dice values are within a range that would be expected under random sampling in 13 out of the 14 forums. Using a multivariate beta regression model, we show that there is evidence that larger sample proportions and the number of threads in a sample are positively correlated with average topic similarity. Finally, we conduct a series of experiments that generalize our findings to sample sizes not observed in Common Crawl.

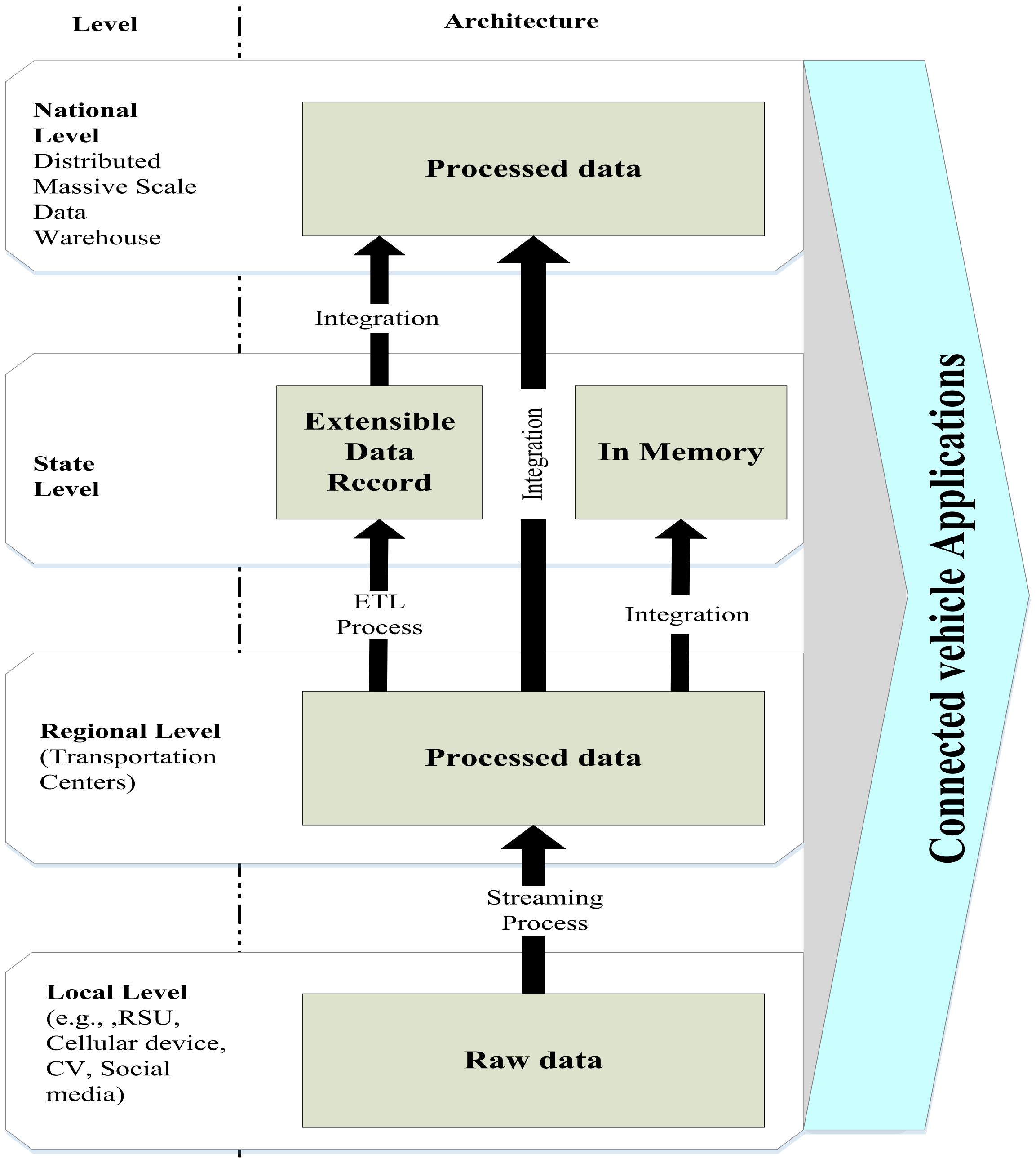
Level
Architecture
National Level Distributed Massive Scale Data Warehouse
Processed data
Integration
Integration
State Level
Extensible Data Record
In Memory
ETL Process
Integration
Regional Level (Transportation Centers)
Processed data
Streaming Process
Local Level (e.g., ,RSU, Cellular device, CV, Social media)
Raw data
Connected vehicle Applications

Figure 4.5: Hierarchical multi-level data infrastructure architecture for Big Data/Data Intensive computing.

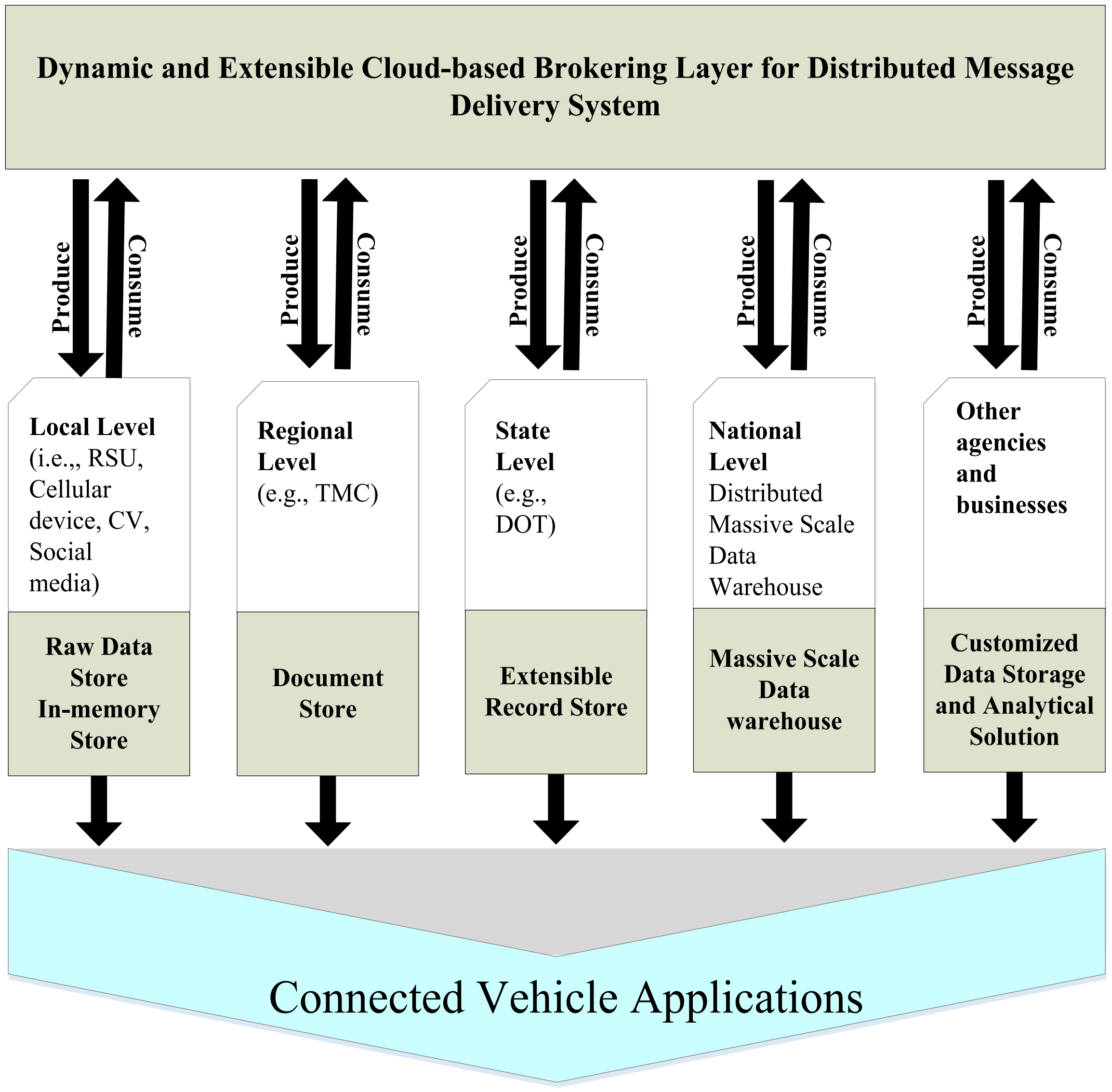
Dynamic and Extensible Cloud-based Brokering Layer for Distributed Message Delivery System
Produce
Consume
Produce
Consume
Produce
Consume
Produce
Consume
Produce
Consume
Local Level (i.e.,, RSU, Cellular device, CV, Social media)
Regional Level (e.g., TMC)
State Level (e.g., DOT)
National Level Distributed Massive Scale Data Warehouse
Other agencies and businesses
Raw Data Store In-memory Store
Document Store
Extensible Record Store
Massive Scale Data warehouse
Customized Data Storage and Analytical Solution
Connected Vehicle Applications

Figure 4.6: Single-level distributed message delivery infrastructure architecture for Big Data/Data Intensive computing.

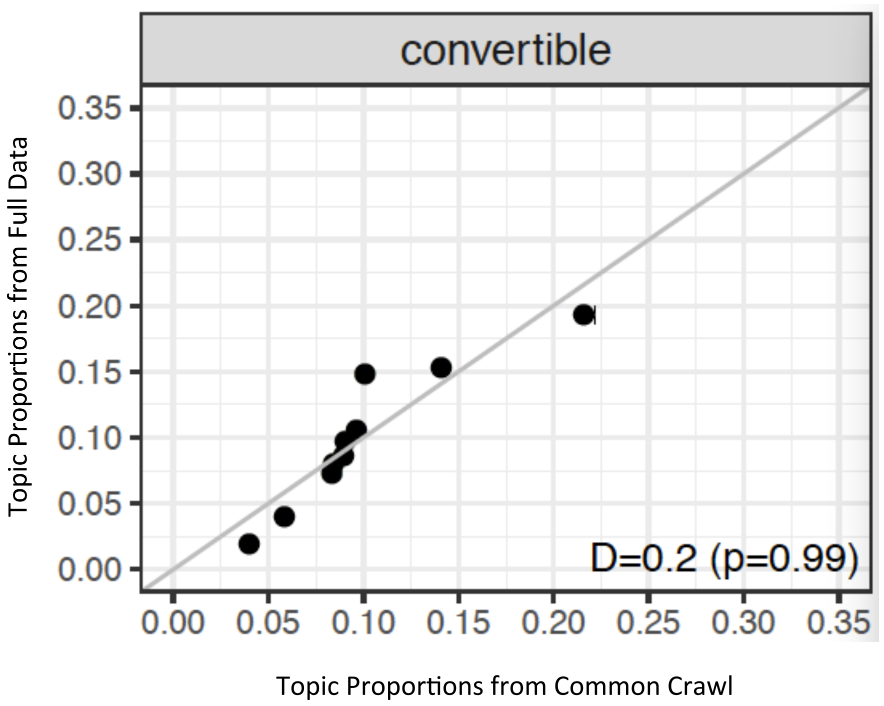

Figure 4.7: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Convertible subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

4.2.1 Topic Proportion Similarity

For each subforum i, we have estimated global topic proportions $M_{\text{CC},i}$ and $M_{\text{FD},i}$. Figures 4.7-4.20 use scatter plots for comparing the two proportions. Most data points are close to the 45-degree line, indicating a high degree of similarity. Their average Pearson correlation is 0.92, with min=0.80 and max=0.97. To formally test the differences between the proportions, we calculate the Kolmogorov–Smirnov (KS) test statistic discussed in Section 3.5.2 for each subforum. These statistics, which are printed in the bottom-right of each figures in Figure 4.7-4.20, range from 0.2 to 0.6, with p-values well above the typical 0.05 threshold. Based on these results, we cannot reject the null hypotheses that the proportions are the same. That is, we do not find statistical evidence that, in terms of their topic proportions, the Common Crawl samples differ from the full data.

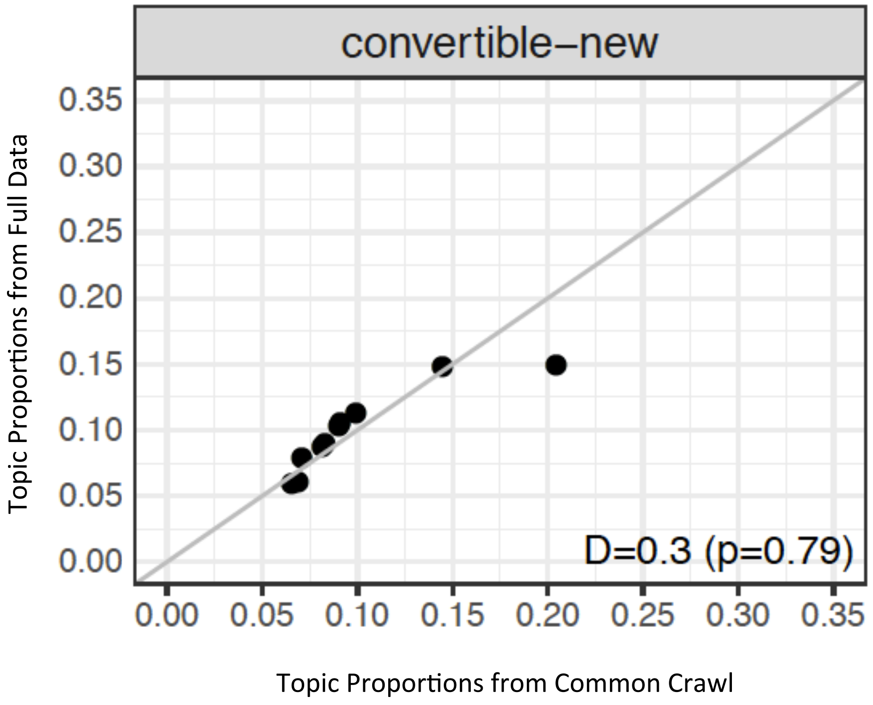

Figure 4.8: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Convertible-New subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

4.2.2 Word Rank Similarity

We next evaluate the LDA topics estimated from Common Crawl in terms of their substantive similarity with the topics estimated from the full data. Figure 4.21 shows the average Dice coefficient for each subforum as black dots, ordered from smallest to largest. The average Dice values range from 0.37 ("suv-small") to 0.64 ("suv-compact"), with an average value across subforums of 0.50. In terms of topic similarity, this means that the average matched topic pair between Common Crawl and the full data overlap by, on average, 50% of their top 20 keywords. In 7 out of the 14 subforums, the average Dice value is above 0.5, indicating that the average Common Crawl topic overlaps with more than half of its words with its matched topic from the full data.

We further quantify the similarity comparisons by considering the size of the average Dice

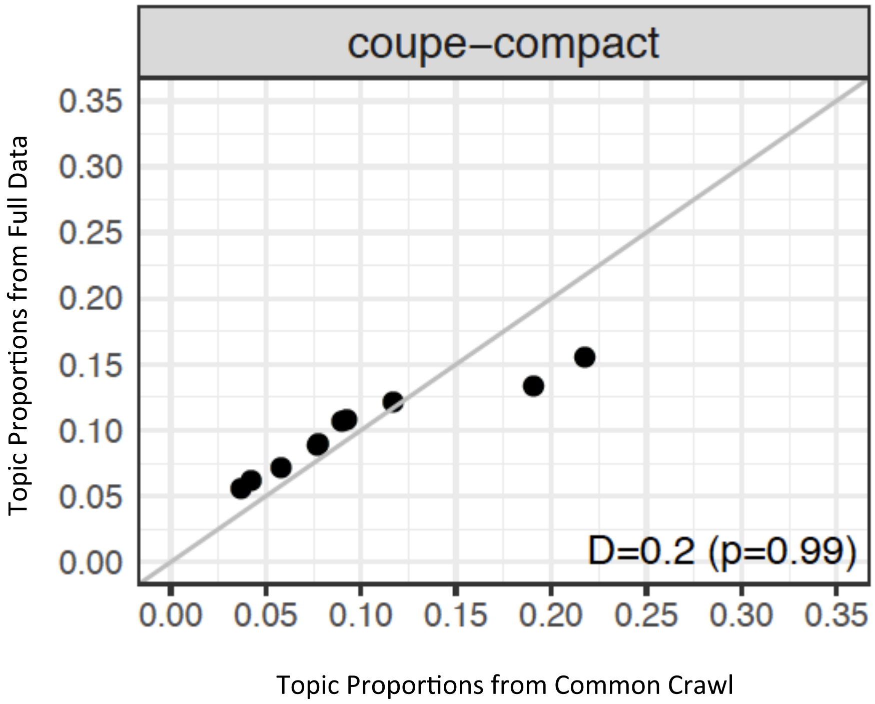

Figure 4.9: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Coupe Compact subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

value one would expect if the Common Crawl data were a true random sample of the full data. This answers the question to what extent the results estimated on the Common Crawl samples behave the same or differently than under true random sampling. To this end, we conduct the following simulation: for each subforum in the full data, we draw 100 random samples (without replacement) of the same size than the subforum in Common Crawl. For each sample we estimate 10 topics and calculate their average Dice value with the same method we applied to the Common Crawl data. These simulations result in 100 average Dice values for each subforum that correspond to possible results one would obtain under random sampling.

The large black dots in Figure 4.21 show the average Dice values calculated from Common Craw. The small gray dots show the average Dice value from each of the 100 simulations together

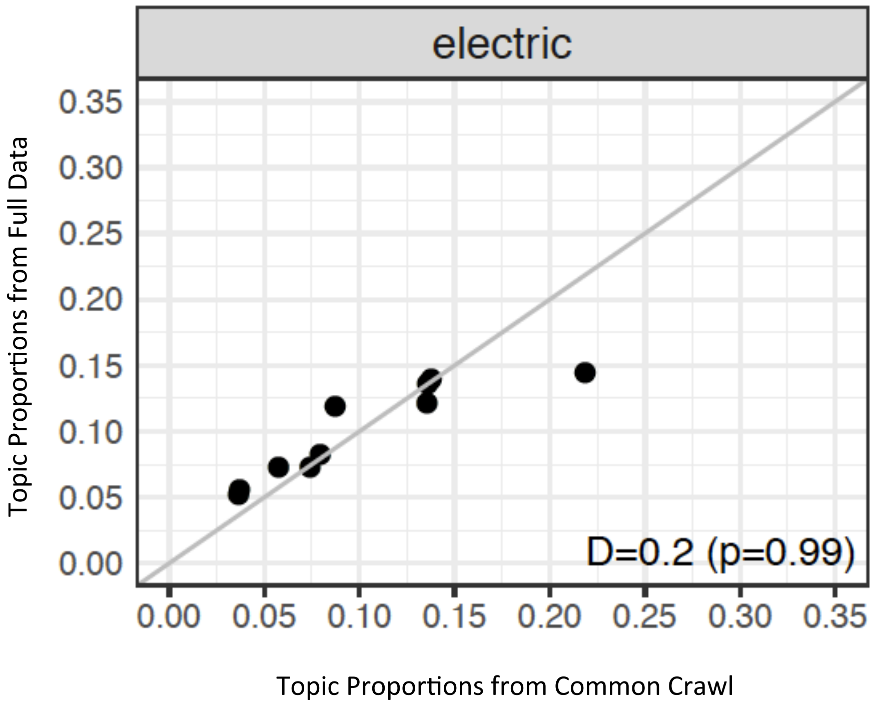

Figure 4.10: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Electric subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

with the 95% intervals of estimated values. In 13 out of the 14 subforums, the average Dice value calculated from Common Crawl falls within the 95% interval of Dice values calculated from the random samples. We can conclude that for these 13 subforums, the topic similarity between Common Crawl and the full data is not significantly different than what one would expect to find under true random sampling. The average Dice value is outside the 95% interval in only one case, the "sport-full" forum, which has the fourth smallest sample proportion. This result indicates that samples with a document fraction below 0.05 might not be suitable for topic modeling because of the possibility that the observed sample might be biased.

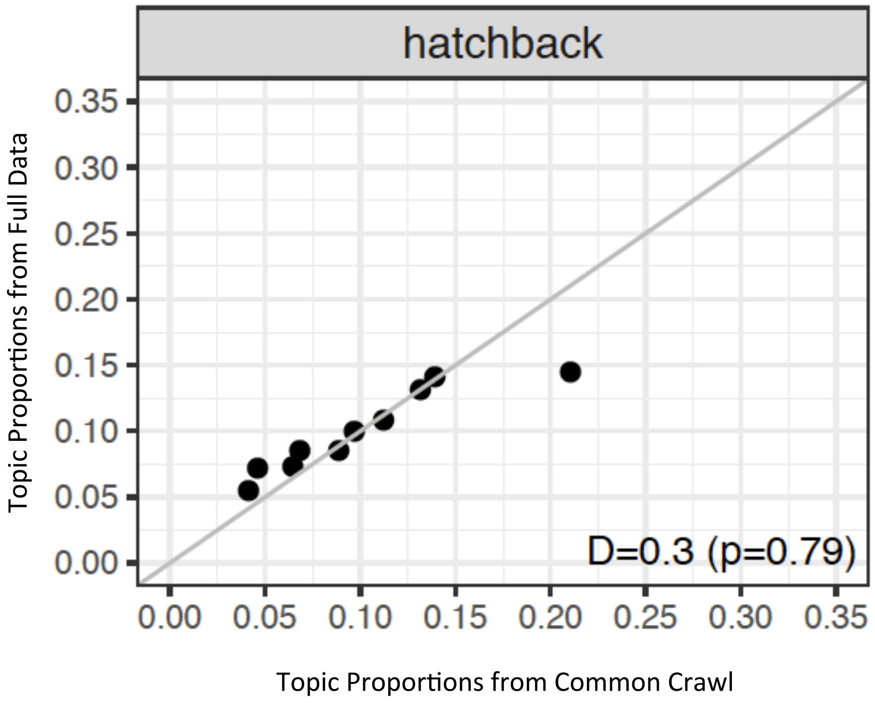

Figure 4.11: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Hatchback subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

4.2.3 Multivariate Regression

We have established in the previous section that topic similarity between Common Crawl and the full data does not significantly differ in 13 out of the 14 forums. In this section, we formally test whether differences in data characteristics are systematically linked to topic similarity. We estimate the joint effect of document fraction, document number, and logged time interval variation on average Dice coefficient with a beta regression [38], which accounts for the response being bound in the $(0, 1)$ interval.

Table 4.3 shows that document fraction and document number have a positive and significant association with the average Dice coefficient at the 0.1 threshold or below. The estimated coefficients represent additional changes in the log-odds ratio of the response. To facilitate their interpretation,

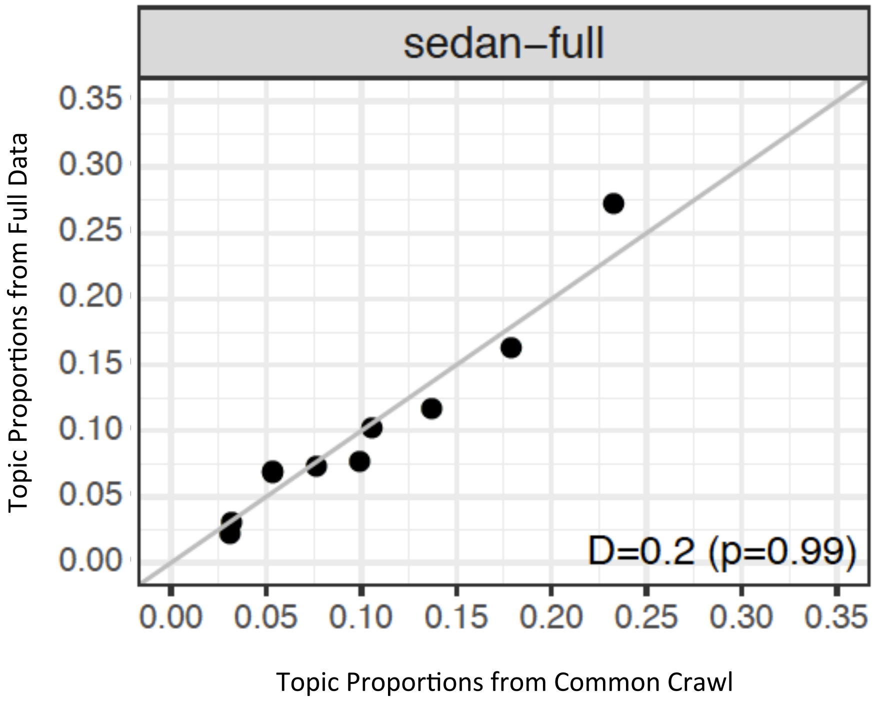

Figure 4.12: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Full-sized Sedan subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

we compute predicted effects when changing a predictor from its empirical minimum to maximum, holding all other predictors at their means. For document fraction, an increase from 0.01 to 0.22 is associated with an average increase in the Dice coefficient by 0.093. For document number, an increase by about 3,200 documents is estimated to increase the response by 0.089.

4.2.4 Experiments on Larger Sample Sizes

The largest sample we observe in Common Crawl includes 22% of the full data. We here conduct a series of experiments to evaluate how the quality of topics estimated on sampled data depends on sample sizes outside our observed range. To this end, we draw 100 random samples (without replacement) from the full data of each subforum at six sample sizes: 0.01, 0.1, 0.2, 0.4,

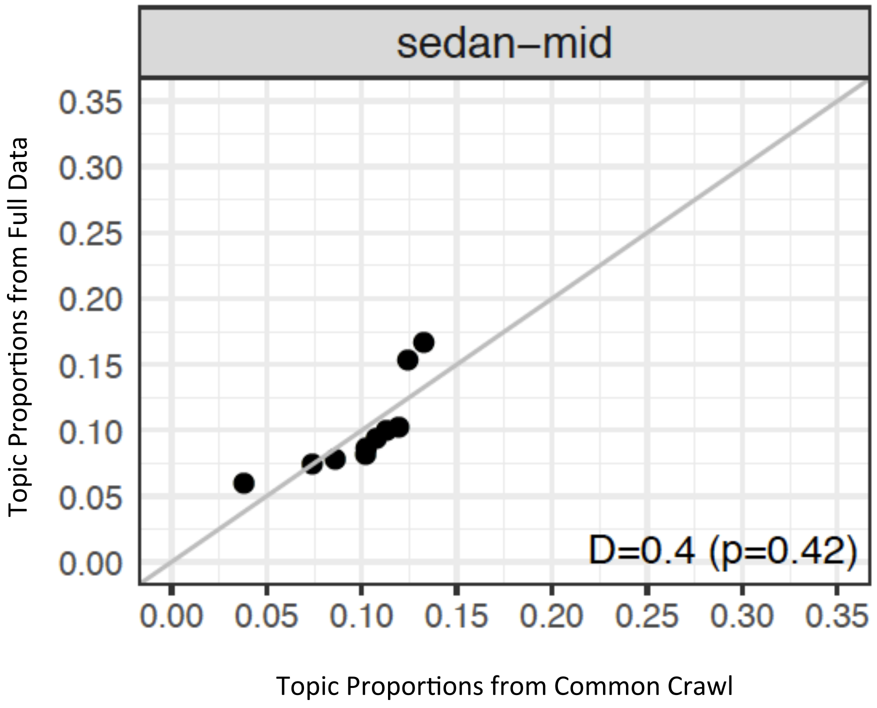

Figure 4.13: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Mid-sized Sedan subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

0.6 and 0.8 of the full data. This results in a total of 8,400 samples (14 subforums $\times$ 6 sample sizes $\times$ 100 random samples). For each sample, we estimate 10 LDA topics (and again 5 topics on “suv-small”) and calculate the average Dice value between the sampled and the full data. We then calculate the average Dice value and 95% interval range for each set of 100 random samples.

The results are shown in Figures 4.22-4.35. We observe that as a general trend, the average Dice value increases with larger samples. We observe the largest increase when the sample proportion is increased from 0.01 to 0.1 and from 0.1 to 0.2. The line then flattens out at sample proportions above 0.2. The largest average similarity measure we observe in our experiments is 0.71. Considering the full range of values within their 95% intervals, we find a maximum value of 0.83.

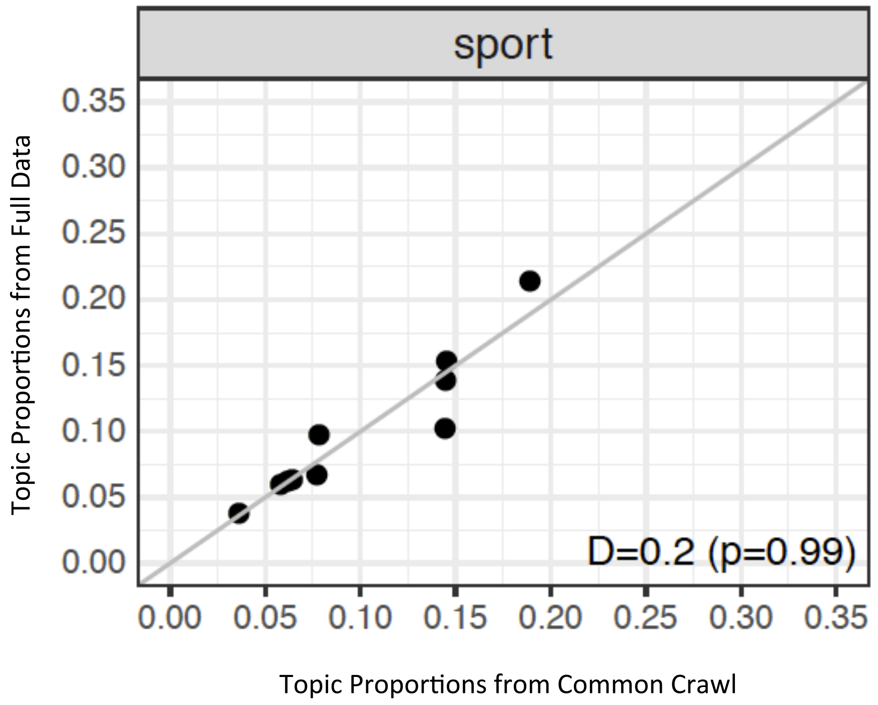

Figure 4.14: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Sport subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

Table 4.3: Results from Beta Regression

	Coef. (std.dev.)
Document Fraction	1.759** (0.795)
Document Number (1,000s)	0.111* (0.064)
Time interval variation (log)	−0.068 (0.054)
Constant	−0.011 (0.280)
Observations	14
Pseudo R^2	0.643
Log Likelihood	25.563

Note: *p<0.1; **p<0.05; ***p<0.01

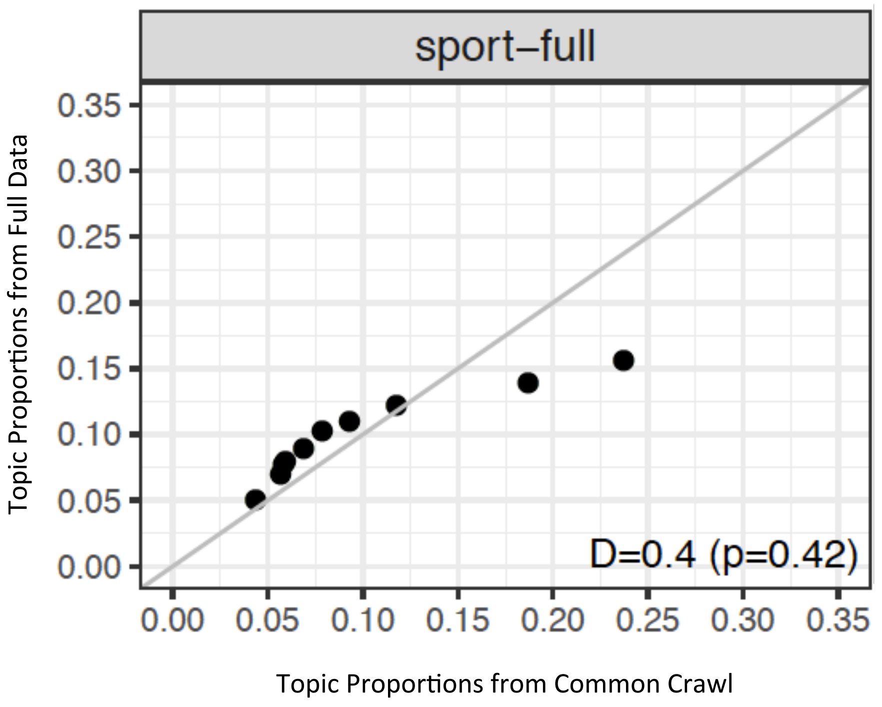

Figure 4.15: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Full-sized Sport subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

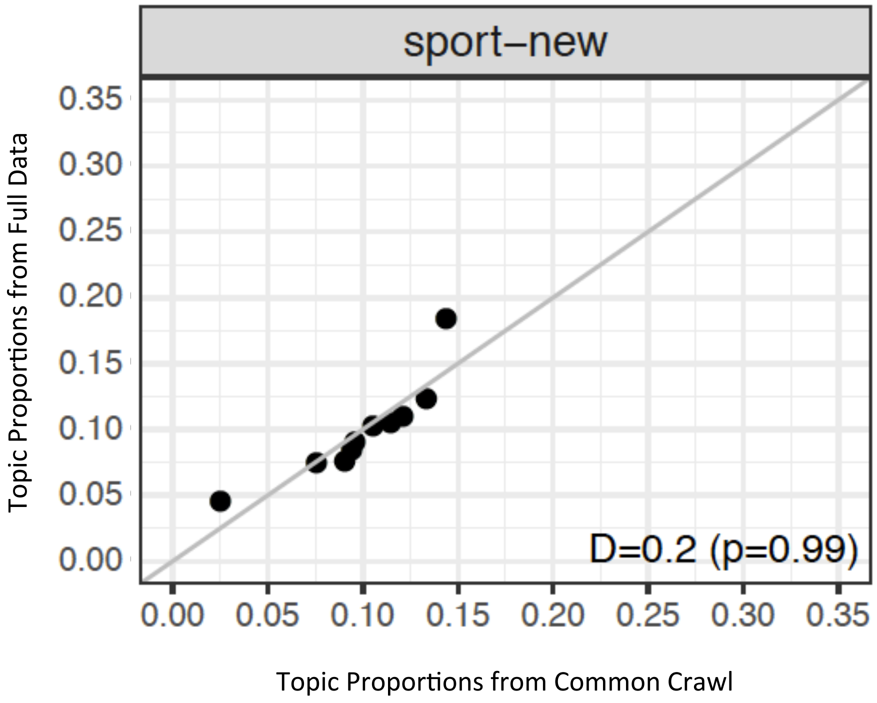

Figure 4.16: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for New Sport subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

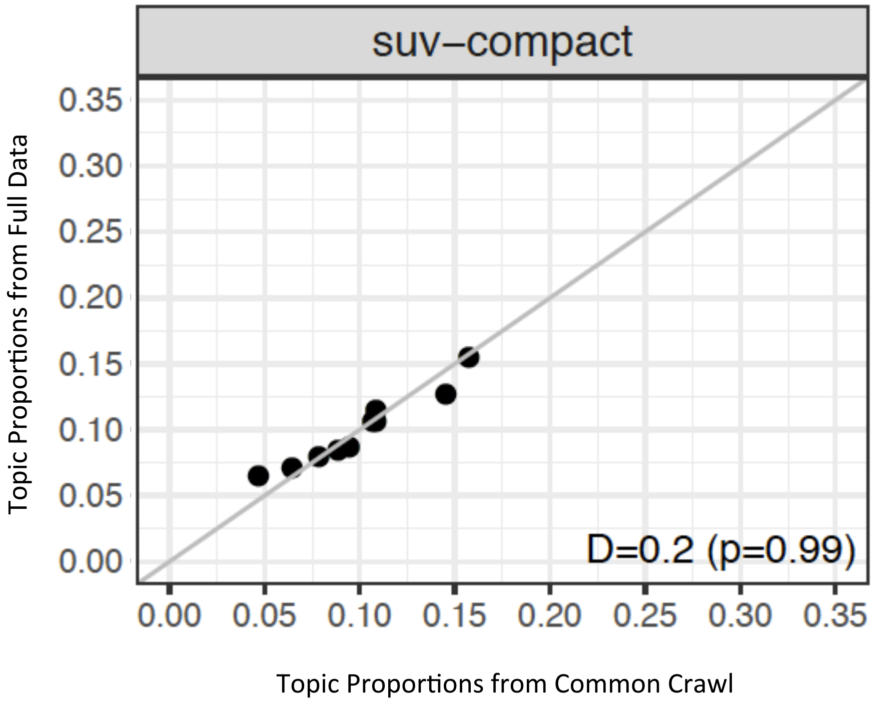

Figure 4.17: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Compact SUV subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

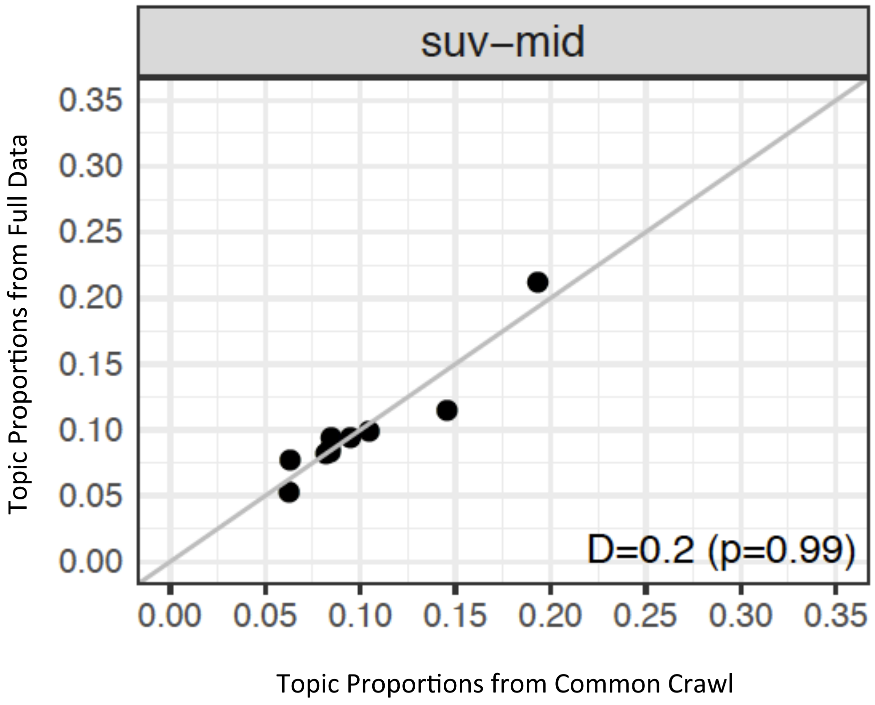

Figure 4.18: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Mid-sized SUV subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

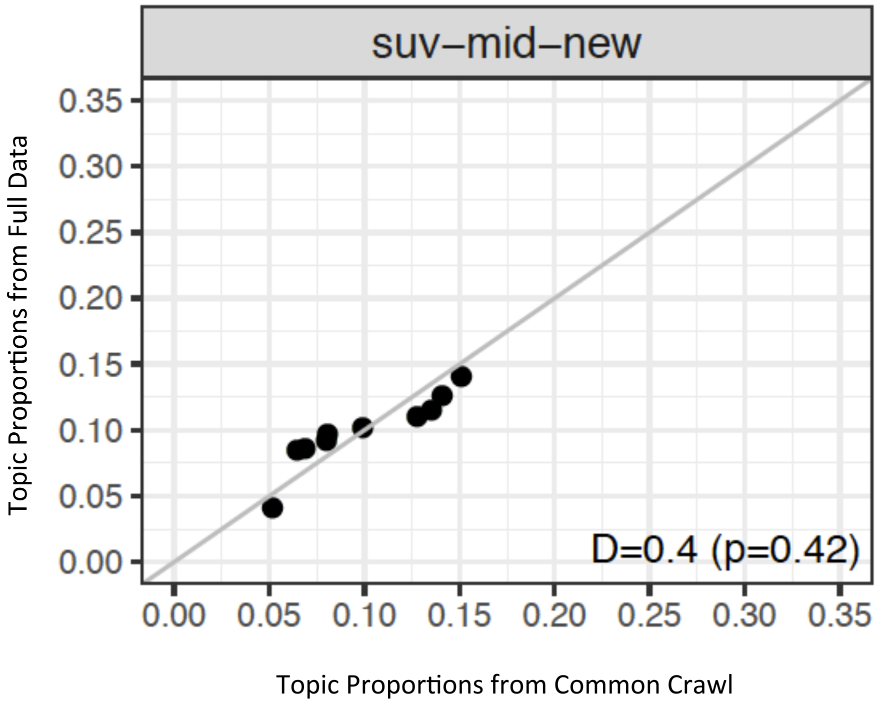

Figure 4.19: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for New Mid-sized SUV subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

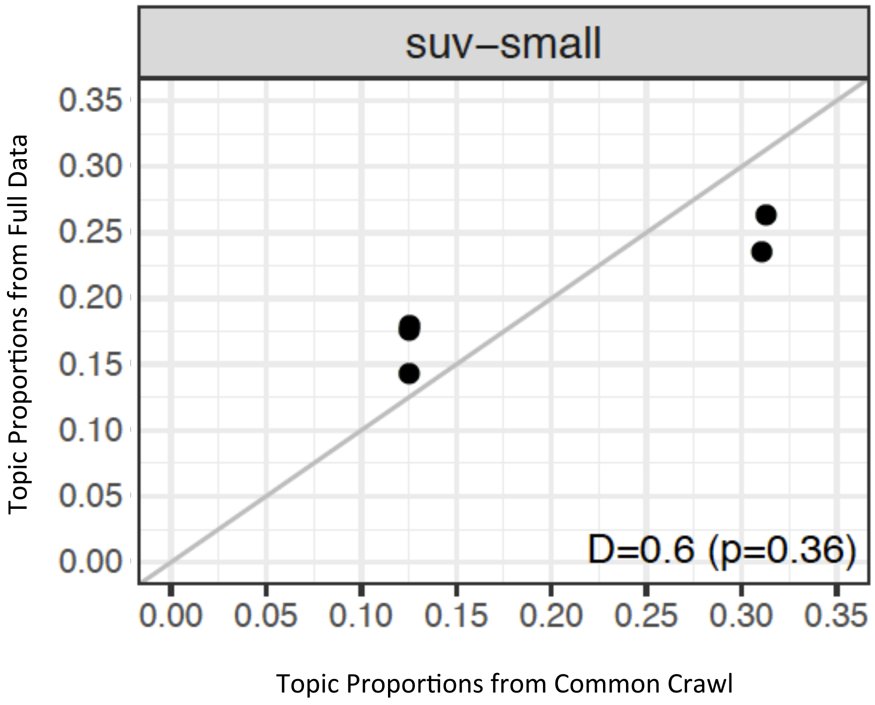

Figure 4.20: Topic proportions estimated from Common Crawl (x-axis) compared to topic proportions estimated from the full data (y-axis) for Small-sized SUV subforum. Gray lines are 45-degree lines. D values in bottom-right are Kolmogorov–Smirnov (KS) test statistics with p-values in parentheses.

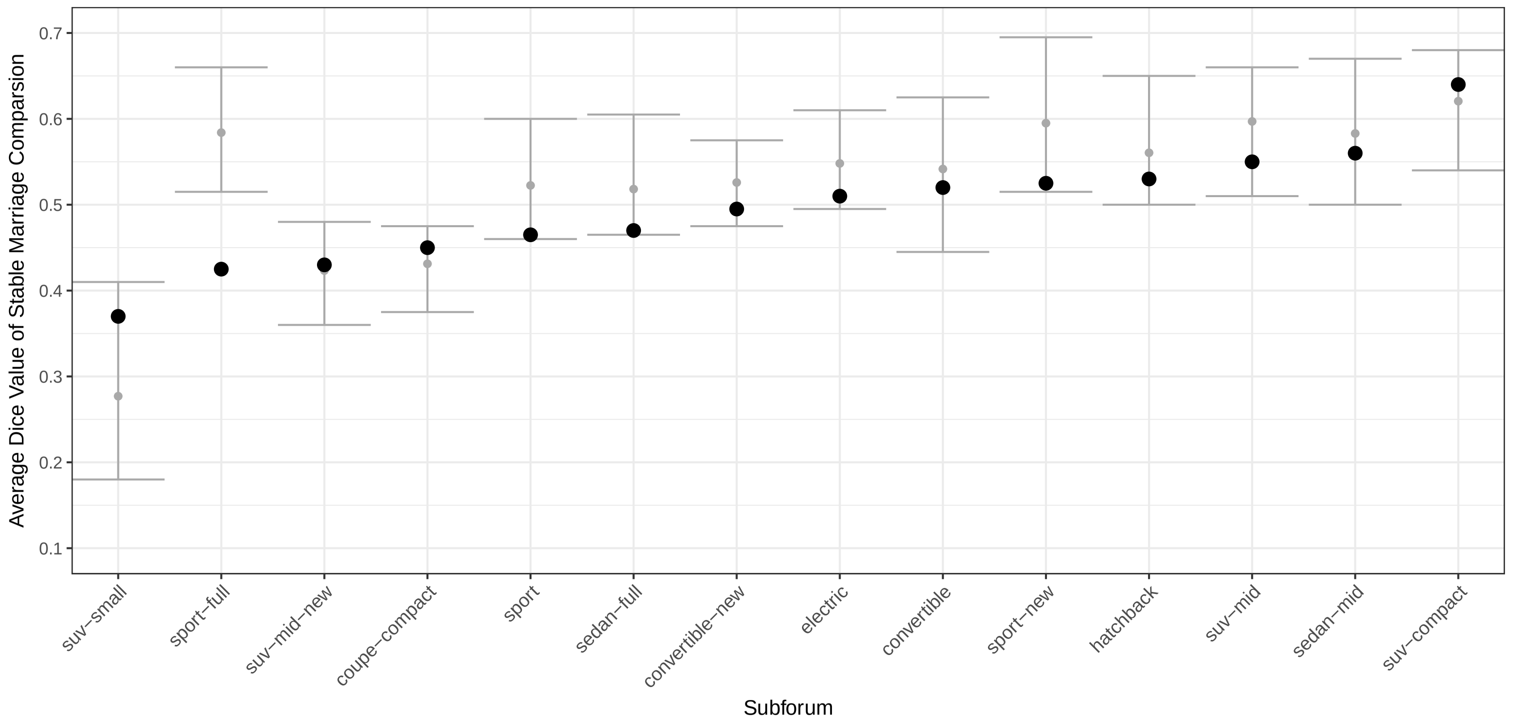

Figure 4.21: Large black dots are average Dice coefficients calculated from matched topic pairs between Common Crawl and the full data. Small gray dots are average Dice coefficients estimated from 100 random samples drawn from the full data of each subforum and with the same sample size as the Common Crawl subforums. Gray lines indicate the 95% interval of values estimated from the random samples.

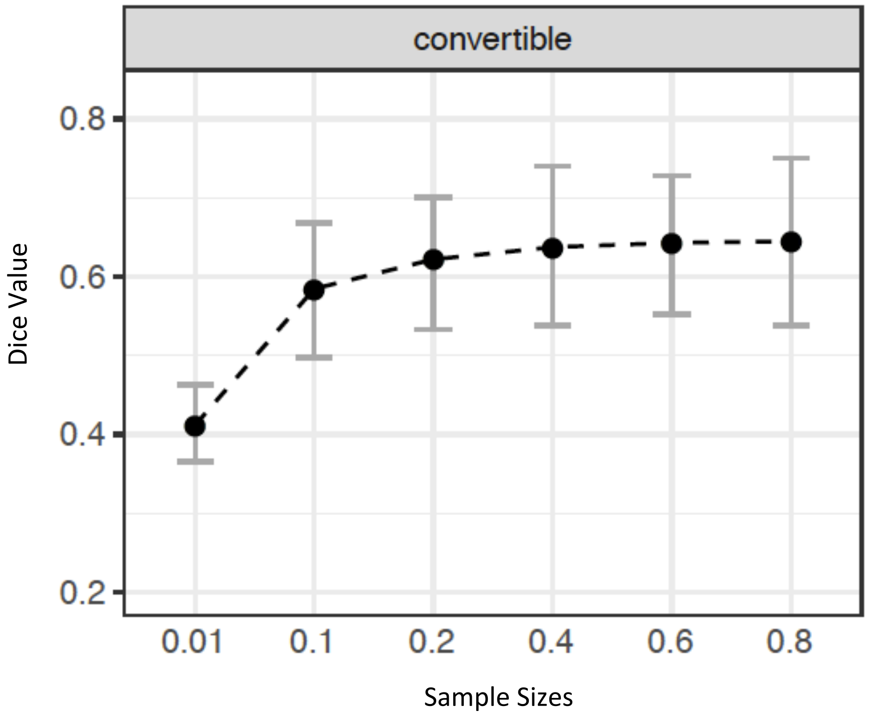

Figure 4.22: Distribution of the Dice coefficient under random sampling in Convertible subforum.

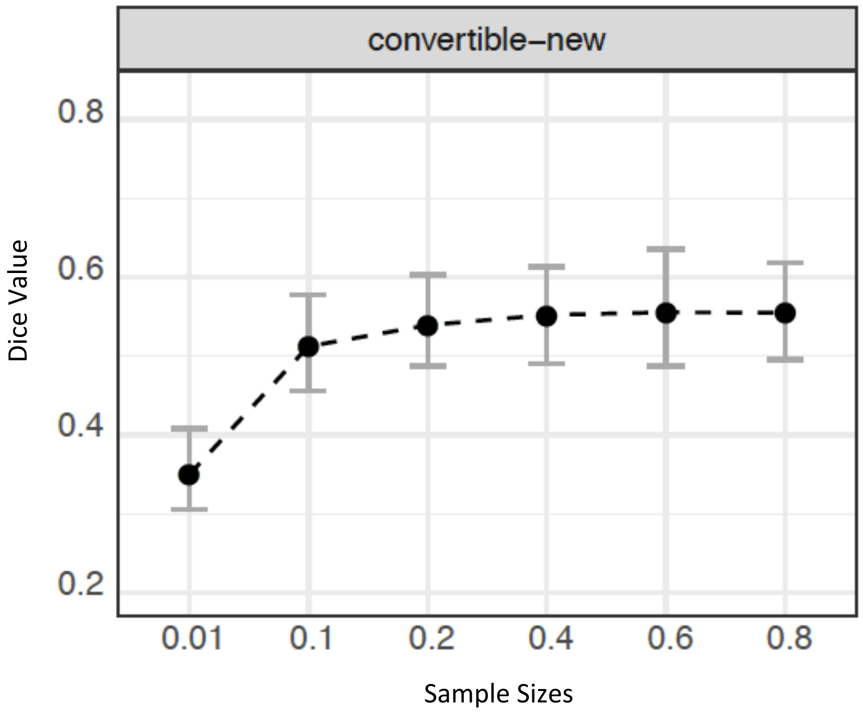

Figure 4.23: Distribution of the Dice coefficient under random sampling in New Convertible subforum.

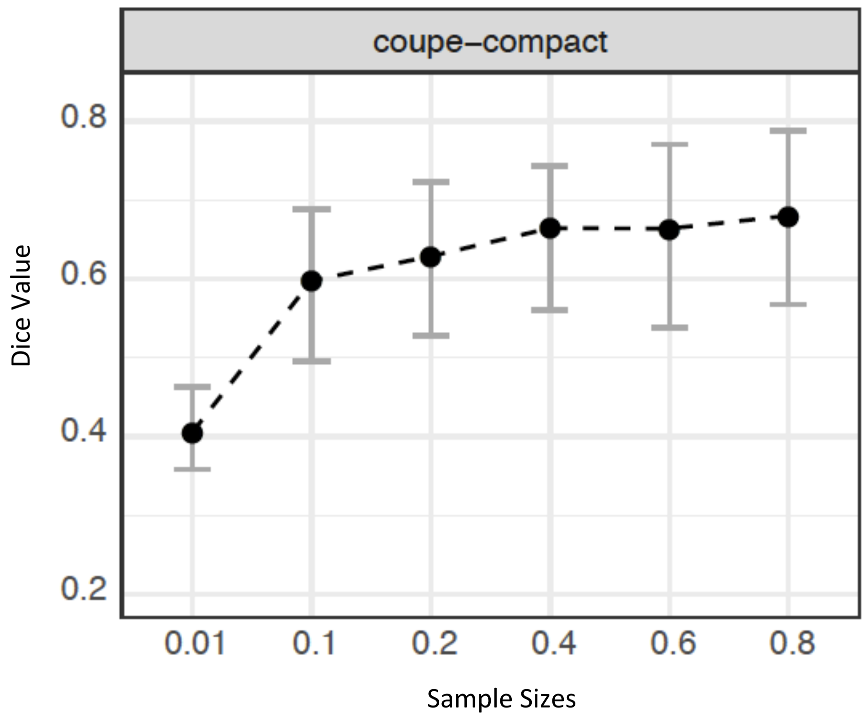

Figure 4.24: Distribution of the Dice coefficient under random sampling in Compact Coupe subforum.

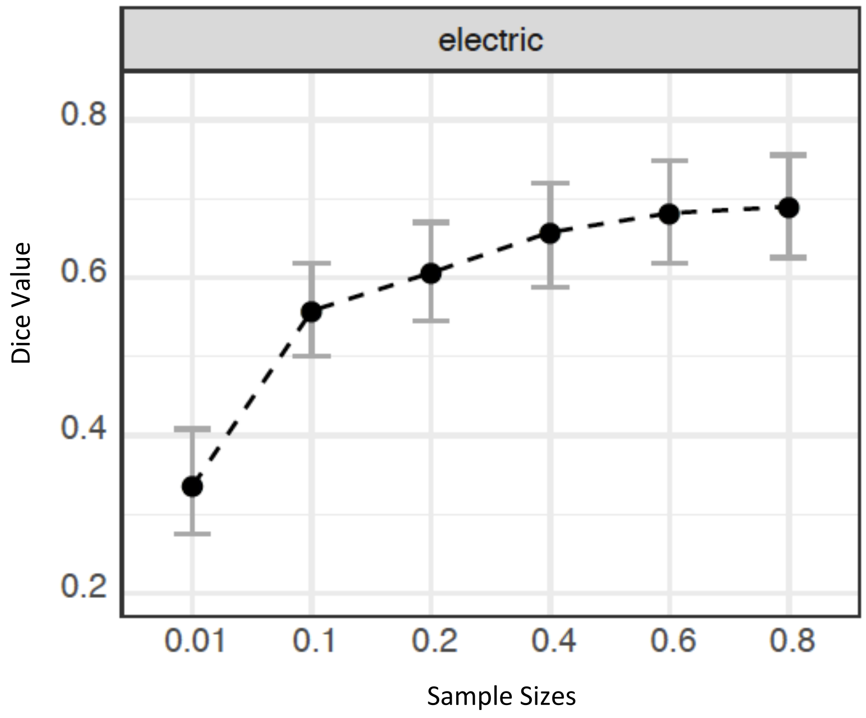

Figure 4.25: Distribution of the Dice coefficient under random sampling in Electric subforum.

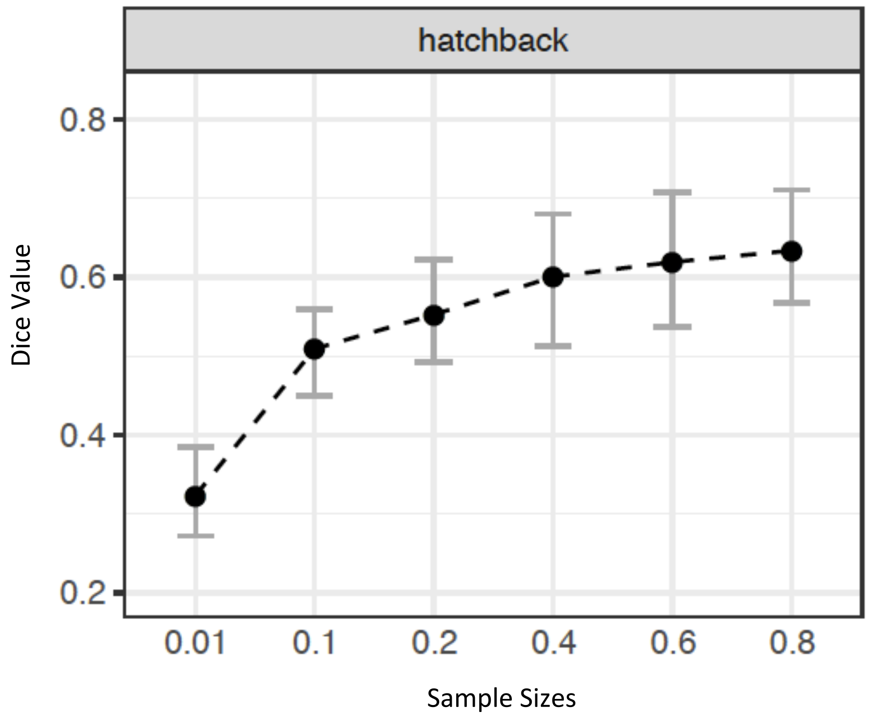

Figure 4.26: Distribution of the Dice coefficient under random sampling in Hatchback subforum.

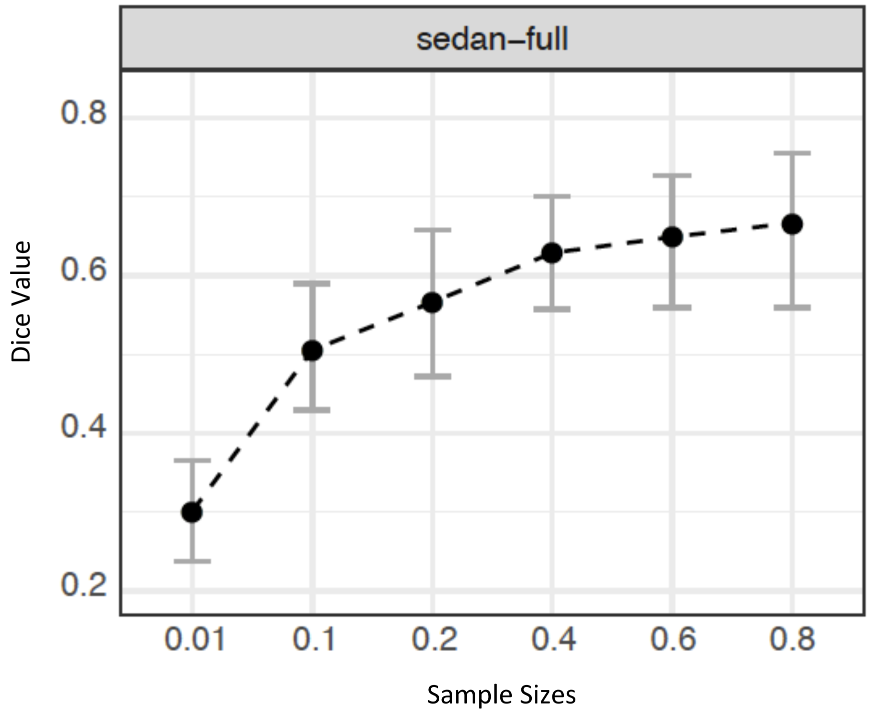

Figure 4.27: Distribution of the Dice coefficient under random sampling in Full-sized Sedan subforum.

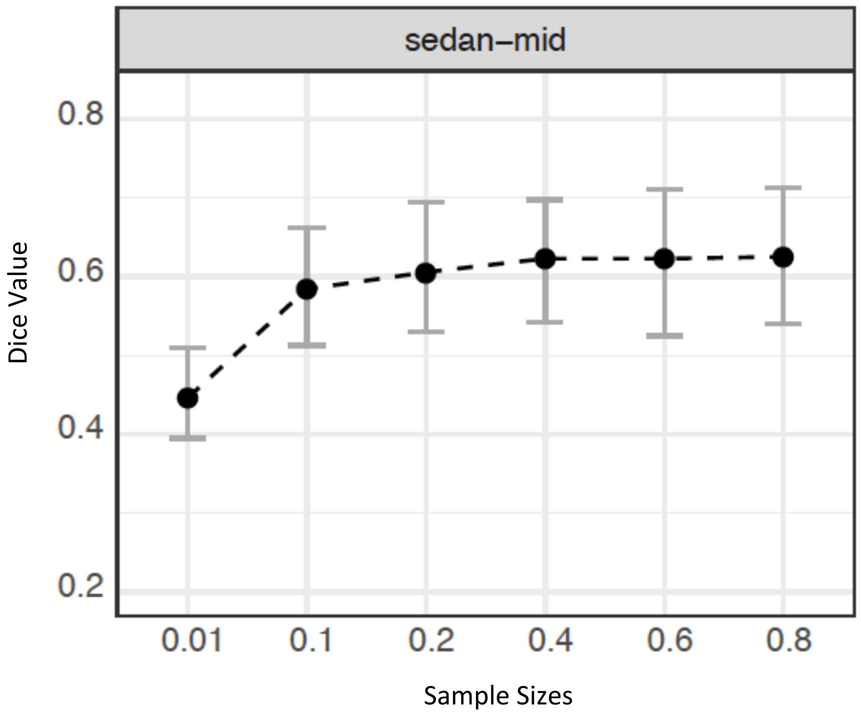

Figure 4.28: Distribution of the Dice coefficient under random sampling in Mid-sized Sedan subforum.

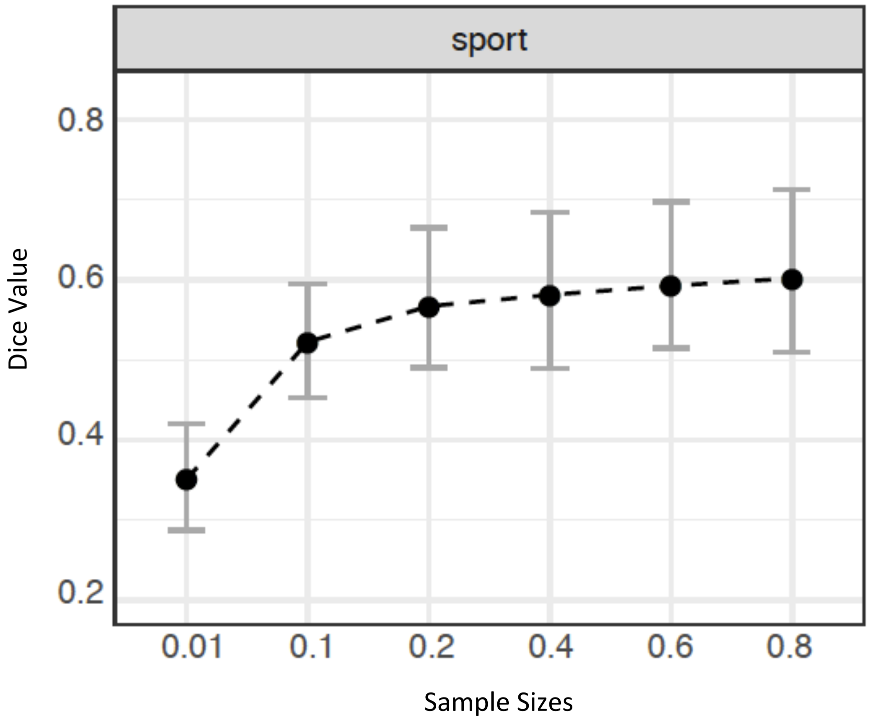

Figure 4.29: Distribution of the Dice coefficient under random sampling in Sport subforum.

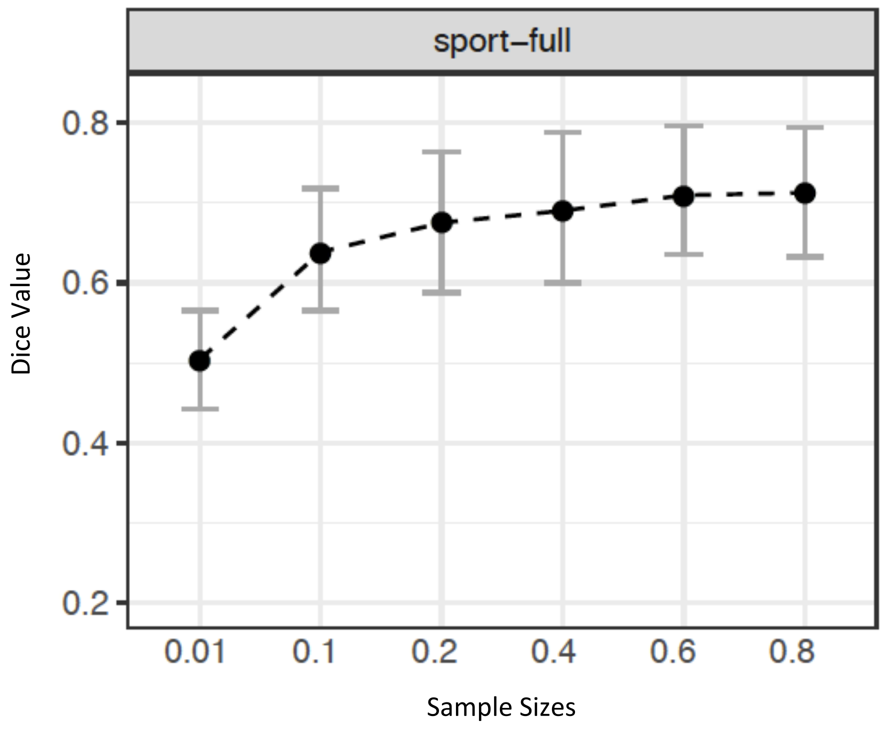

Figure 4.30: Distribution of the Dice coefficient under random sampling in Full-sized Sport subforum.

The results in Figures 4.22-4.35 provide two important insights for the application of LDA on sampled data. First, we observe that even for samples that include 80% of the full data, the average Dice value does not exceed 0.71 (or 0.83 if we take the full range of values within the 95% interval into account). This indicates that, at least for the data included in our analysis, topics estimated with LDA are sensitive to the types of documents included in the sample. Second, there is a diminishing return in topic similarity for increasing sample sizes. In a majority of cases we observe that topic similarity only increases by a small fraction beyond a 0.2 or 0.4 sample size, indicating that samples at these sizes may be sufficient for the estimation of LDA topics if collecting the full data is too costly or too time intensive.

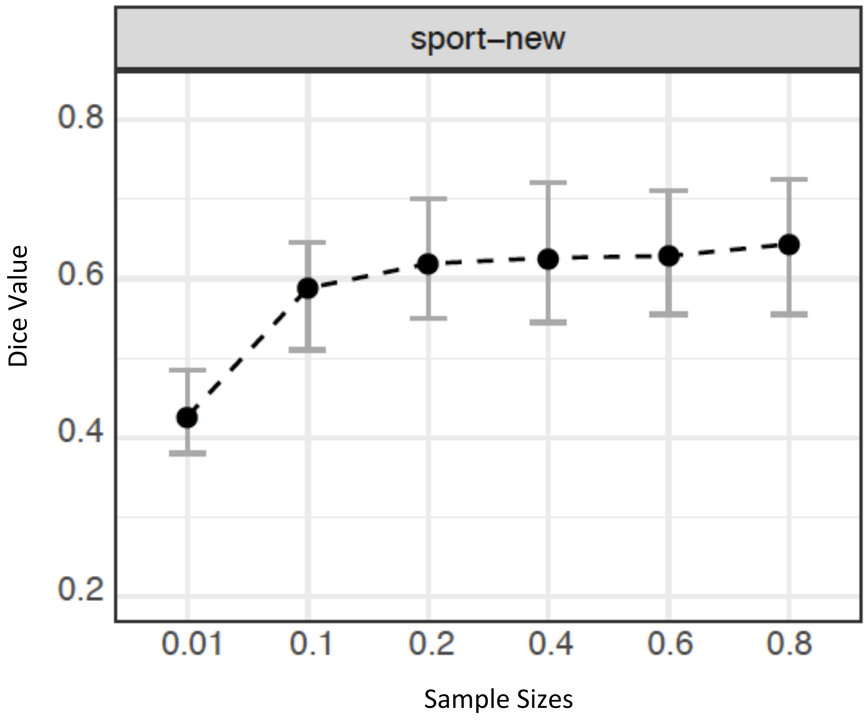

Figure 4.31: Distribution of the Dice coefficient under random sampling in New Sport subforum.

4.3 Insights Into Online Forum User Behavior

In this section, we use our estimated topics from Common Crawl to provide insights into customer behavior as expressed in online forum discussions. We focus our discussion on four subforums that we selected because they represent different car classes: “electric”, “sedan-mid”, “sport”, and “suv-compact”. Table 4.4 shows the top 10 keywords for the two largest topics from each of the car classes.

We observe that for all four classes, the look of the car is the most dominant topic. While the color black is mostly mentioned on Sedan-mid, Sport, and SUV-compact classes, the color blue is mostly mentioned on the Electric class.

The second largest topic in each subforum shows distinctive characteristics across the four car classes. In the Electric subforum, the topic represents a focus on battery performance-related

Table 4.4: Top ten keywords from largest and second largest topics from four selected subforums

Electric		**Sedan-mid**	
First topic	Second topic	First topic	Second topic
car	electric	looks	car
looks	charging	car	vendor
twitter	solar	nice	iphone
electric	battery	sedan-mid	new
blue	level	black	dealer
look	time	look	phone
concept	vehicles	wheels	usb
interior	available	great	need
first	fast	front	system
tesla	use	pics	problem
Sport		**SUV-compact**	
First topic	Second topic	First topic	Second topic
sport	sport	suv-compact	tires
coupe	better	black	suv-compact
sport-new	power	looks	wheels
convertible	performance	sport	winter
nice	weight	interior	snow
love	drive	color	rims
wheels	brakes	pics	suv-mid
black	track	package	need
vendor	steering	trim	price
looks	even	wheels	set

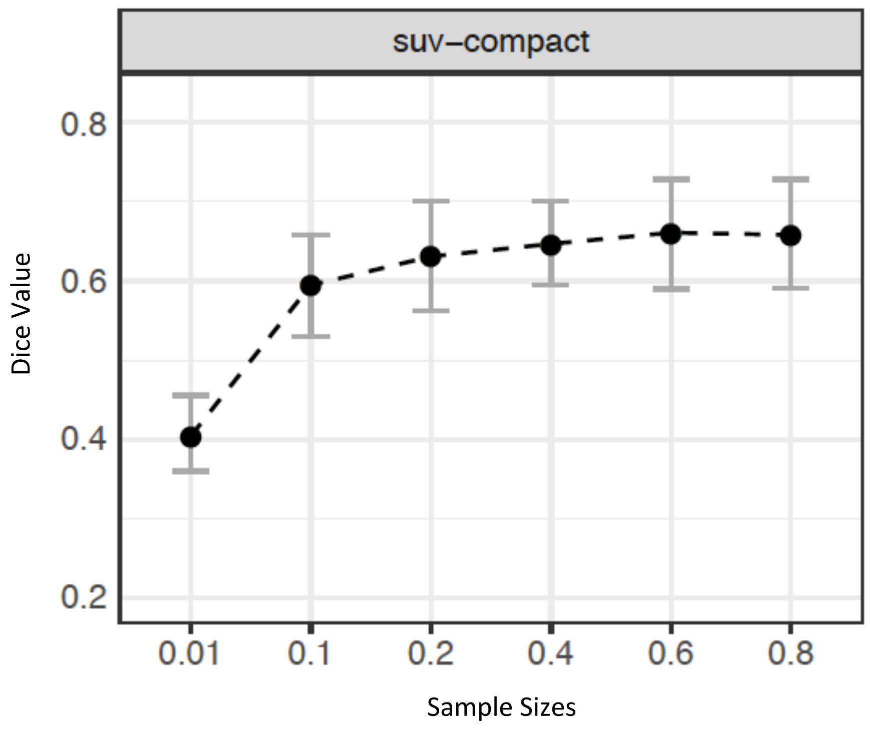

Figure 4.32: Distribution of the Dice coefficient under random sampling in Compact SUV subforum.

keywords covering facets like charging and battery level. In the Sedan-mid class, the second largest topic focuses on phone and Iphone associated user experience, suggesting the need for studying compatibility of phone usage inside the car for improved user experience. In the Sport subforum, the second largest topic is around handling and control aspects of the driving experience, covering facets like steering and brakes. Finally, in the SUV-compact subforum, the second largest topic focuses on driving experiences during the winter season, suggesting that the utility of wheels and tires should be of concern for the specific car vendor.

From a business perspective, topic models enable analysts to infer themes from documents in an unsupervised, automated way. By annotating documents with topics, navigation and processing of the text data is improved. Another important application is the combination of topic models and supervised classification approaches, e. g., to sort documents into a fixed set of categories (e. g., a model or defect category). Topic models provide a condensed document representation that is also

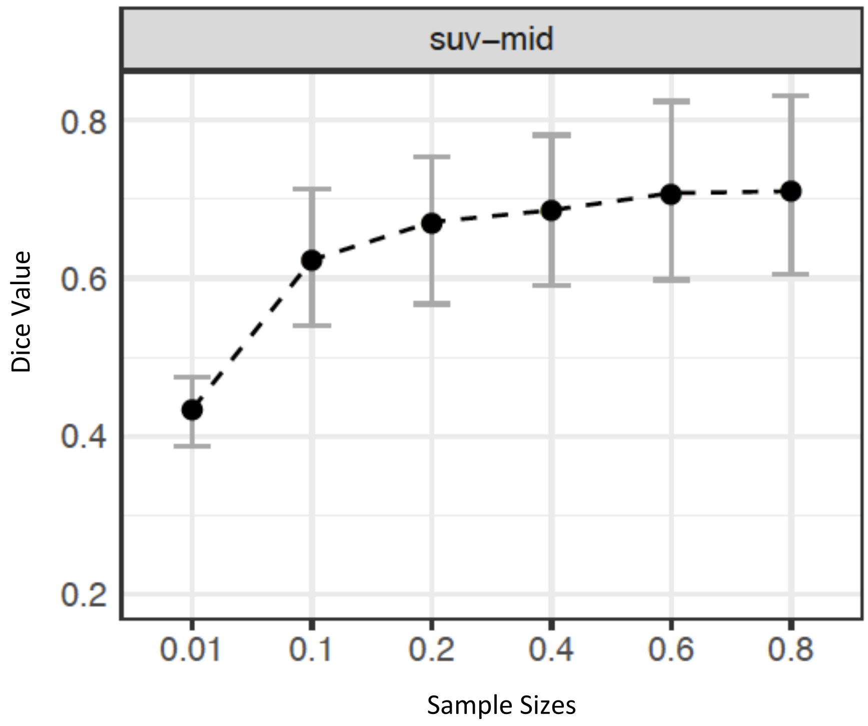

Figure 4.33: Distribution of the Dice coefficient under random sampling in Mid-sized SUV subforum.

well suited as input for classification algorithms.

We first show the top topics and key words of two car classes in the following table, the sedan-mid class and the electric class.

4.3.1 Common and Unique Topics Across Vehicle Classes

Topics estimated from LDA can be summarized by calculating the total proportion that each topic is represented in the corpus. Figure 4.36 summarizes these topic mixtures for each vehicle class as a stacked bar chart. Each cell in the bar chart represents one topic, with the height of the cell indicating the proportion that a topic is represented in the threads and posts in a segment.

In order to calculate the extent to which a topic is unique to its vehicle class, we calculated the Dice coefficients for all pairwise comparisons between a topic and all topics outside its own segment. The cell numbers in Figure 4.36 and corresponding color coding represent the maximum

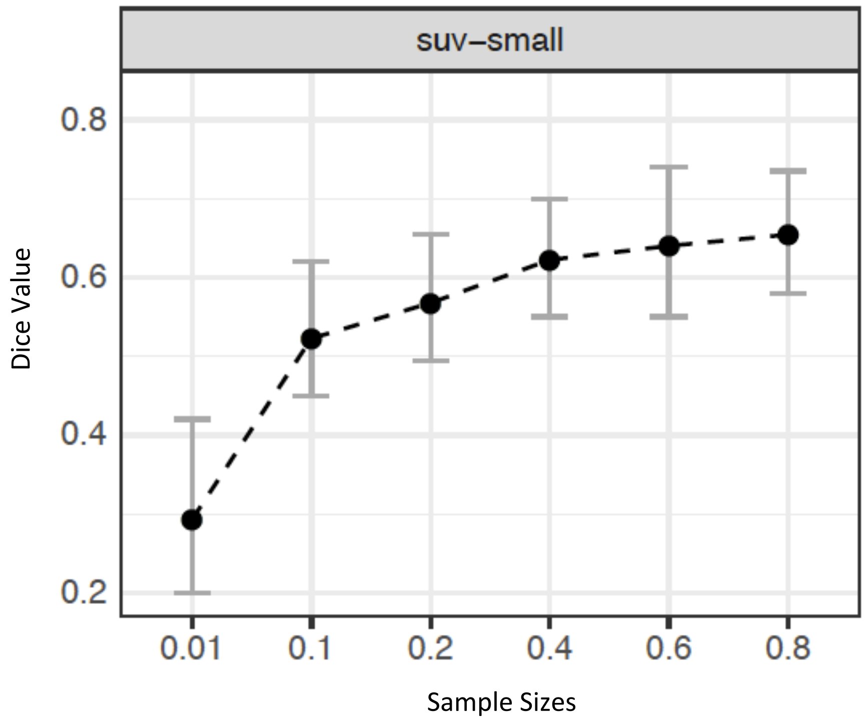

Figure 4.34: Distribution of the Dice coefficient under random sampling in Small-sized SUV subforum.

Dice coefficient calculated from these pairwise comparisons. This coefficient is bound between 0 and 1, with larger values (and color approaching red) indicating a larger overlap.

Looking first at the average maximum Dice value for each vehicle class, which is printed on top of each bar, we find that discussions around electric vehicles are characterized by the most unique topics (mean = 0.46), followed by convertibles (mean = 0.51), sports cars and SUVs (both with a mean of 0.53), and finally sedans (mean = 0.6). Our results therefore provide evidence that user interactions around electric vehicle topics are very different from discussions around other vehicle types, possibly because the technological differences between electric and gasoline vehicles lead to different challenges and questions that motivate users to interact in an online forum.

Next, looking at the distribution of maximum Dice values within each vehicle class, we find that the most unique topics tend to be located on either the top or bottom of the bars, with the least

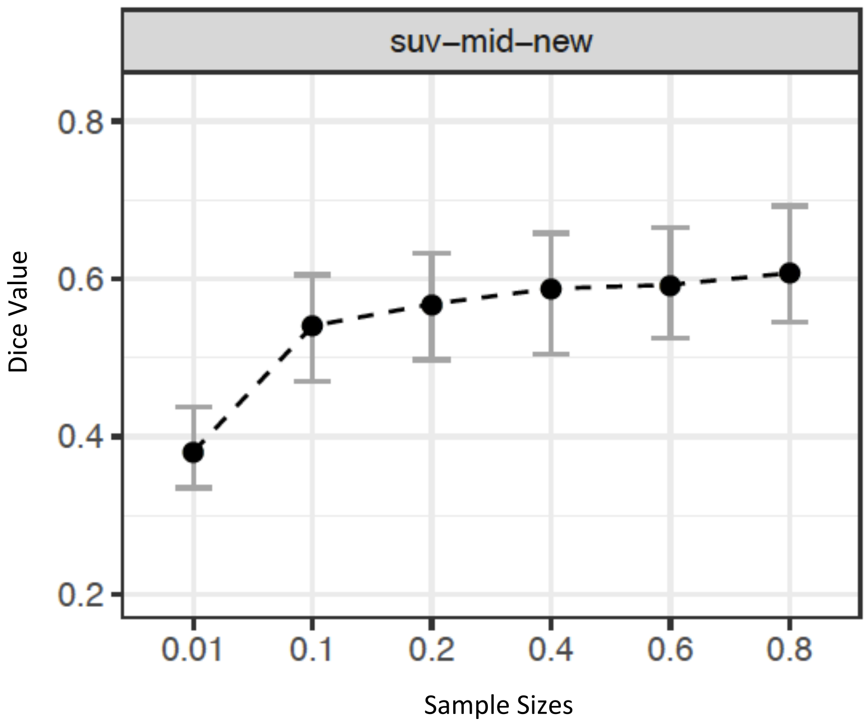

Figure 4.35: Distribution of the Dice coefficient under random sampling in New Small-sized SUV subforum.

unique (i.e., the most common) topics located somewhere in the middle. That is, the largest and smallest topics (as measured by their proportions) that characterize the discussions in a subforum tend to be those that are substantively different from topics in other forums.

Finally, Figure 4.36 shows that the two most common topics, with a maximum Dice value of 0.85, can be found among electric vehicles and sedans. However, looking at the corresponding word lists, we find that this is an artifact of the forums data, with each topic including a large number of html commands that were not captured in the cleaning process.

4.3.2 Topic Similarity Between Vehicle Classes

In the previous section we have looked at the general uniqueness of topics as determined by their pairwise comparisons with topics outside their own segment. In this section we answer the

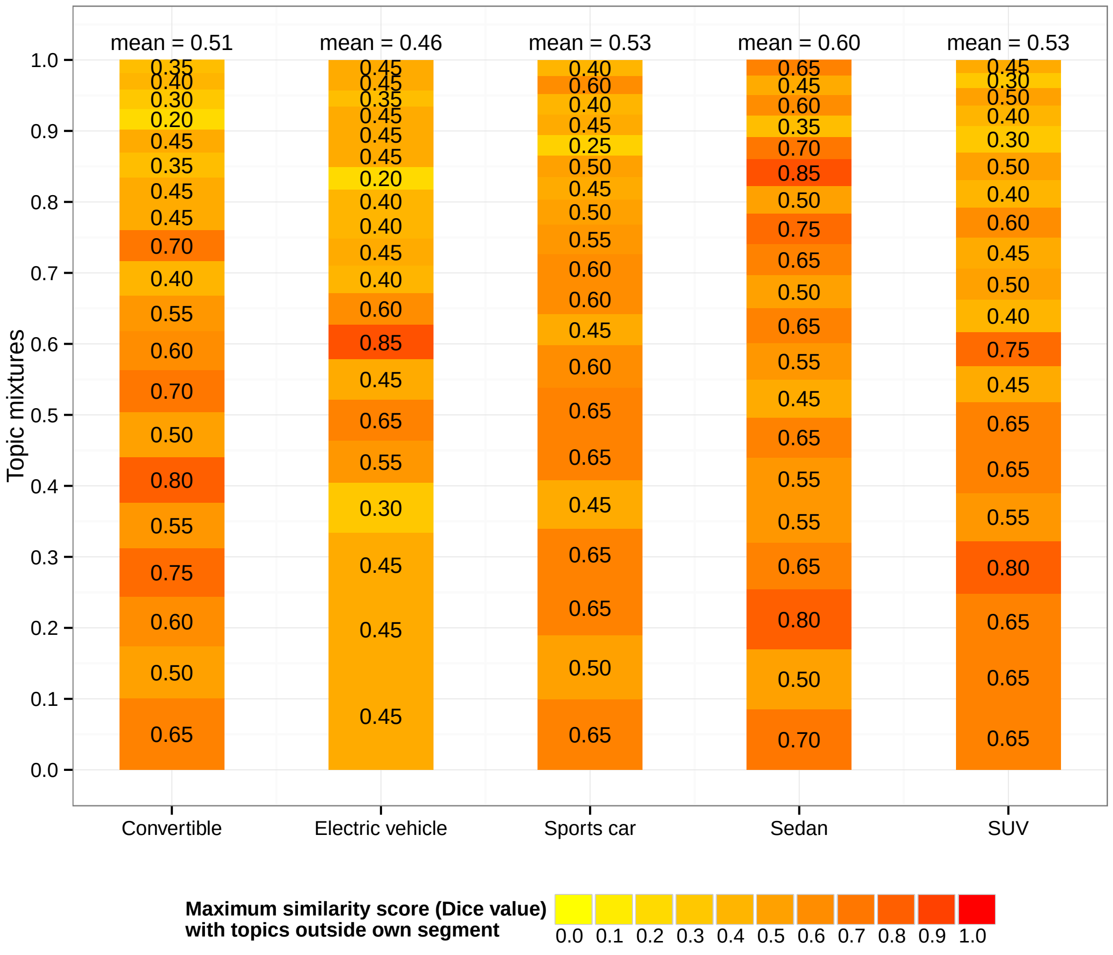

Figure 4.36: Bar chart with max Dice value

question which vehicle classes are most similar to each other based on their topic similarities.

Figure 4.37 shows the same stacked bar chart as displayed in Figure 4.36, with cell values indicating the topic with the largest Dice value in its segment, and arrows pointing to the corresponding topic to which a topic has been matched. These matches indicate that forum discussions around two separate vehicle classes have at least one topic in common. Several important insights emerge from these comparisons:

- The highest matching topic for convertibles is shared with SUVs. This match is reciprocal (indicated by the double arrow).
- Electric vehicles share their highest matching topic with sedans, which again is in both direc-

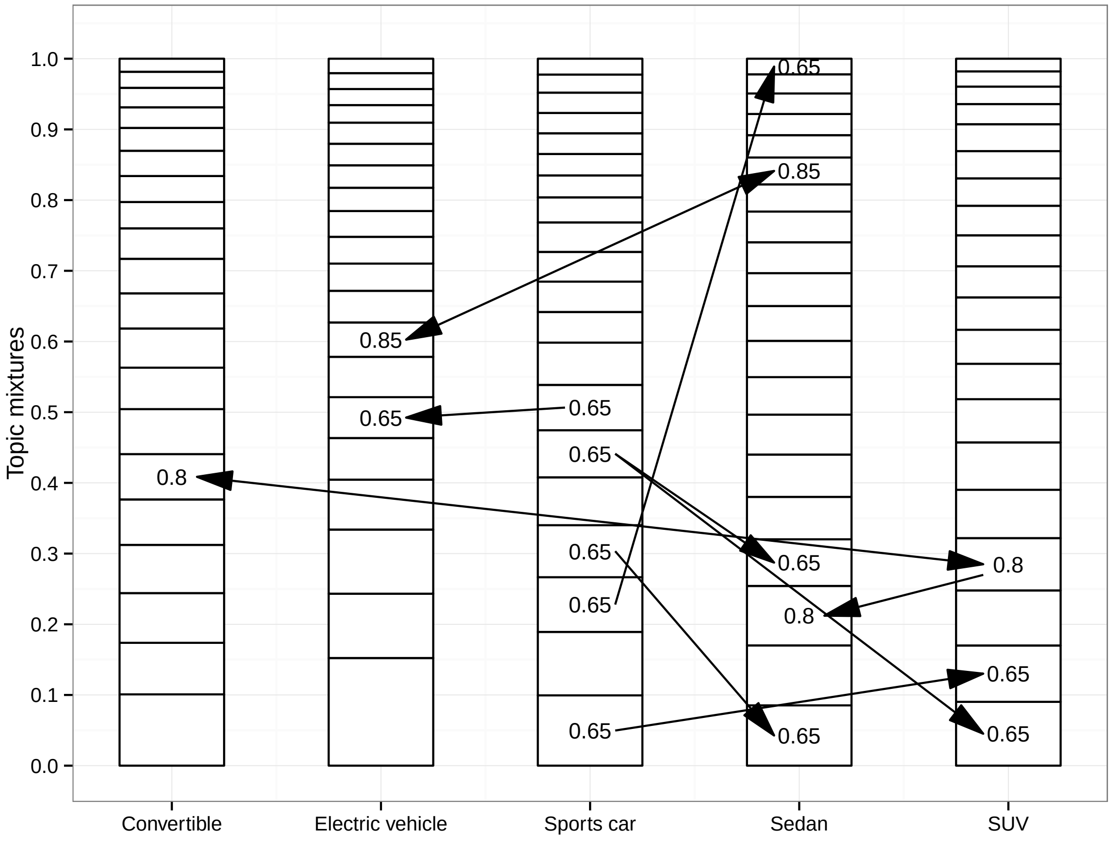

Figure 4.37: Bar chart with matches across segments

tions.

- The highest matching topics for sports cars have a relatively small value of 0.65, indicating that discussions around sport cars are fairly unique. Five topics have a Dice value of 0.65, three that match with sedans, and two matching electric vehicles and SUVs, respectively.
- SUVs have two top-matching topics with a value of 0.8, one that matches (in both directions) to convertibles, and one matching to sedans.

We did pair-wise comparisons of topics from one segment to topics from all other segments using the dice score as an indicator. The arrows are showing the most similar pair of topics from each segment to another. In each segment, there is one or several (if there is a tie) arrows pointing outwards, indicating the maximum-dice-value topic pairs when using topics in that segment for comparison. The start of an arrow shows the topic being compared in that segment, and the end of an arrow shows the topic being compared to in the other segment. We combined two arrows pointing to each other (reciprocal topic pairs) to one bi-direction arrow for the ease of drawing the graph.

From Figure 4.37, we find that the most similar topic pair exists between a topic in the Electric class and a topic in the Sedan class. The topic modeling results suggest that these two topics are actually two HTML noise topics. The second most similar topic pair exists between a topic in the Convertible class and a topic in the SUV class. These two topics has a dice value of 0.8. By looking at the topic modeling results, the following words are listed in both topics:

[‘great’, ‘love’, ‘look’, ‘would’, ‘color’, ‘car’, ‘looks’, ‘pics’, ‘black’, ‘thanks’, ‘nice’, ‘interior’, ‘white’, ‘really’, ‘wheels’, ‘like’].

This suggests that a common topic about color of wheels and interior are discussed in both SUV and Convertible cars. And the color of black or white receives the most attentions which are potentially positive. Besides, SUV also has a high match topic pair with Sedan, which shares the common words:

[‘great’, ‘good’, ‘love’, ‘look’, ‘would’, ‘color’, ‘car’, ‘one’, ‘black’, ‘looks’, ‘nice’, ‘interior’, ‘wheels’, ‘pics’, ‘really’, ‘like’].

This topic is also about the color of car interior and wheels. If we ignore the HTML noise topic pair in Sedan, the topic about color of car interior and wheels is actually the most similar common topic between the three classes of Sedan, Convertible and SUV.

The different words in Sedan topic and Convertible topic are:

[‘blue’, ‘see’, ‘think’, ‘congrats’], which suggests that the blue color is most discussed in Sedan but not in Convertible.

Based on the common topics and uncommon topics we discovered, there is clearly a color preference difference between users of Convertible and Sedan. Another interesting finding is that the Sports car has the most different topics than all other car classes. Although it has 5 equivalent maximum-dice-value topic pairs connected to other car classes, all of them has a relatively low dice value of 0.65. Similarly, the Electric car has a maximum matching value of 0.65 if the noise HTML topic is ignored. This suggests that the Sports car and Electric car classes are more one-of-a-kind than the other classes. Topics discussed in these classes’ segments are mostly unique and uncommon.

The arrows are connecting the maximum similarity topic pairs across segments. The arrows are showing the most similar topic of one segment to the matching topic of the other segment. In each segment, there is exactly one arrow pointing outwards.

The arrows show the largest topic match (based on the Dice coefficient) from one segment to another – indicated by an arrow. I've only included the top matching topic, and I din't break ties because multiple matches are interesting.

There are a couple of important insights in this chart:

- The highest match for convertibles is to SUVs, and this match is reciprocal (indicated by the double arrow). SUVs additionally have a high match to Sedans.
- The highest match for electric vehicles is to sedans, which is in both directions.
- Sports cars have five topic-matching topics, but with a relatively small Dice coefficient of 0.65. This means topics around sports cars are fairly unique because there is no high-matching topic. Three out of these five topics match to Sedans, indicating some overlap between the two segments.
- Figure 4.36 shows the maximum dice value of each top-20 topics when matched when topics from other segments. Topics in the middle has higher dice values than topics at the top or bottom part of each segment. The topics at the bottom and at the top often has less-than-maximum similarity scores, which means the most popular and least popular topics discussed in each segment are different from each other. Topics that has the highest matching scores are usually in the middle of a car class, which means the most common topics normally has medium popularity across segments.
- Figure 4.36 also reinforces our finding in Figure 4.37 about Sport car and Electric car. On average, the colors in Sports and Electric cars are lighter than the other classes, suggesting that they are distinctive among all car classes.
- Topics in the middle match better across segments than topics with either very high or very low proportions.
- The top matching topics are usually located somewhere in the middle, with the smaller topics at the top showing the smallest similarity scores.

- This chart also neatly confirms our finding from the first bar chart that sports cars have the most unique topics, with the largest similarity scores being 0.65 compared to values in the 0.7-0.85 range for the other car types.

4.4 Results of Automated Query Expansion of Streaming Social Media Data

Metrics. For each query associated with one LDA topic, we use the following metric to measure the quality of the tweets in the result. Instead of measuring the quality of each individual tweet as in [64], our indicators evaluate the aggregated tweets in the query result as a whole.

- **Tweet Count** counts the number of tweets matching the query condition starting from query time to end of the experiment.
- **Hash-tag Count** counts the number of hash-tags contained in the tweets matching the query condition starting from query time to end of experiment. We also use a weighted hash-tag count as indicator which weighs each hash-tag by their frequency in the corpus and their relevance to the event of interest.
- **Entropy** measures the entropy of the tweets matching the query condition starting from query time to end of experiment. The entropy is a measure of information contained in the query result and it is based on the frequency of words in the tweet corpus used in the experiment.
- **Convex Hull Area** measure the coverage area of the convex hull generated by placing all terms in the query result in the semantic vector space. This area indicates the semantic divergence and focus of each query and is used to compare the semantic differences between topics associated with queries.

Tweet Count Metric. We take the first window where our algorithm detects an emergent event (interval 16) and evaluates the quality of each of the 5 queries created using the aforementioned metrics (Table 4.5). Figures 4.38-4.42 shows the tweet count metric for the query result associated with each of the five topics generated using three of our proposed methods. The dotted lines are the smoothed regression lines that represent the level of emergence and noise for each query. The two query approaches are represented by blue lines (Query 2) and green lines (Query 3).

Table 4.5: Query result evaluation using LDA keywords

topic	**topic0**	**topic1**	**topic2**	**topic3**	**topic4**
tweet count	65841	307133	404814	774451	1383303
hash-tag count	37051	159220	225463	275028	632944

- **Variance on emergence and noise level between topics.** The first key observations we have is based on the tweet count metric of Query 3 (using only LDA keywords). Our goal is to generate queries that can catch the emerging topic but filter out unwanted noises. The shape of blues dotted lines in Figures 4.38-4.42 show the difference on emergence and noise level of each topic. For topic 0, 1 and 3, the blue lines show an impulse of high tweet count (≥ 100) around the interval when the query is triggered (interval 16). Then the tweet count decreases to 0 gradually at around 500 intervals for these topics. The shape of these curves resembles an exponential distribution, indicating a high level of emergence. On the contrary, for topics 1 and 4, the blue lines show an low tweet count around the triggering interval (≤ 50). The shape of the these curves follow a fluctuating and flat pattern that resemble an uniform distribution. The fluctuating pattern indicates high noise level of these topics. Since our goal is to suppress the noise in the query and catch the emergent information, we introduced Query 3 where the LDA words are combined with DEC keyword to form a new query. Figure 4.39 shows that for topic 1, the query result using approach Query 3 (green line) has much less noise compared to that of Query 2 (blue line), showing an impulse tweet count only at triggering interval 16. We further examine the quality of the queries using other metrics like hash-tag count and entropy later in the section.

Convex Hull Metric. To visualize the differences in semantic information contained in different query methods, we use convex hull method that outlines the query results as polygons in the vector space.

Figures 4.43-4.46 show the convex hull coverage results for 3 different query methods. Each polygon (in yellow) in Figure 4.43-4.45 is a convex hull in the vector space which represents the semantic coverage of an associated query. Figure 4.45's query is constructed using 3 LDA keywords. Since the keywords used to construct the Query of Figure 4.45 contains only LDA keywords, the words distribution inside the convex hull in Figure 4.45 is dense and concentrated on the right upper corner. The left bottom part of the convex hull contains less words and represents the noise in a

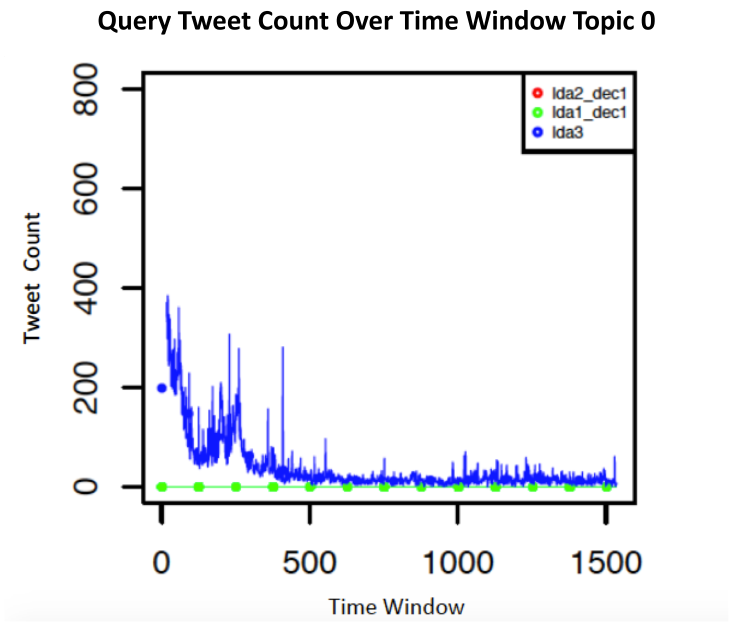

Figure 4.38: Tweet Count Over Time Window of Query Results with Topic 0

Table 4.6: Nearest Words for Selected Words in GDELT Feature Space

climate	**malaysia**	**curfew**	**protest**	**riot**
change	airlines	curfews	week	prison
economic	flight	demonstrators	members	activists
countries	china	gunmen	high	bodies
department	aircraft	demonstrations	group	parliament
global	countries	ceasefire	students	carolina
september	ukraine	bangkok	early	demonstration
secretary	plane	crackdown	against	opposed
human	agreement	cleared	april	armed
security	international	imposed	political	emergency
issue	money	checkpoint	held	matter

query. Figure 4.43's query is constructed using 1 DEC keyword and 1 LDA keyword; Figure 4.44's query is constructed using 1 DEC keyword and 2 LDA keywords. It is observed that both the coverage shape and words density distribution of Figure 4.43 and Figure 4.44 are similar to each other, indicating that adding an LDA keyword results in a good sample of the original vector space of a more noisy query. Figure 4.46 compares the convex hull coverage of the three queries directly in a shared space. The key observation is that by adding a DEC keyword to LDA keywords and

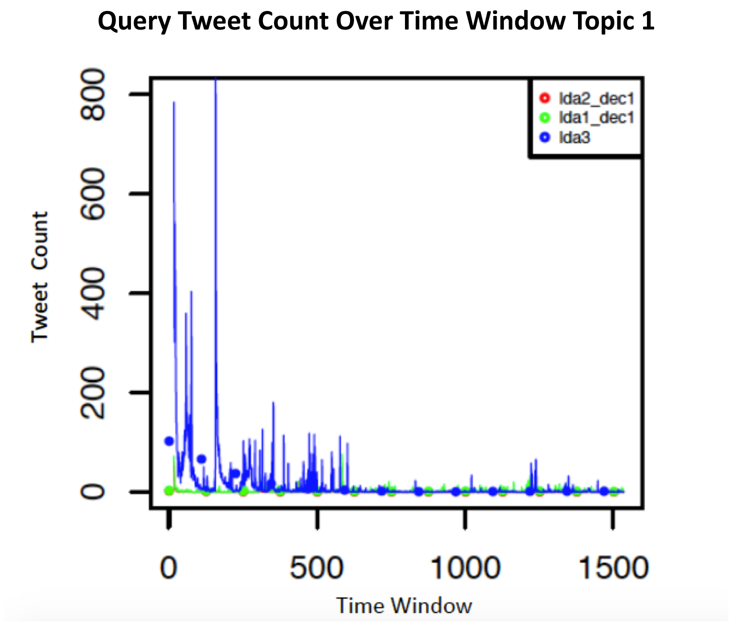

Figure 4.39: Tweet Count Over Time Window of Query Results with Topic 1

construct a new query (method dec_1_lda1 and method dec_1_lda2) effectively filtered out the noise located at the bottom left corner of the convex hull of Query lda3. This suggests that our proposed method effectively captures the key semantic information contained in topic modeling keywords while being able to filter out the unwanted noise.

GDELT Feature Space. In order to construct a feature space representing a historical stream that we could introduce novel keywords through, we turned to the Global Database of Events, Language, and Tone. From this repository of daily news articles, press briefings, and reports, we scraped 1,000 articles a day for the 2014-2015 year, totaling 3.6 million articles. A Ball Tree-based Nearest Neighbors algorithm was used to collect nearby words in the feature space. Selected words and their 10 nearest neighbors are summarized in Table 4.6.

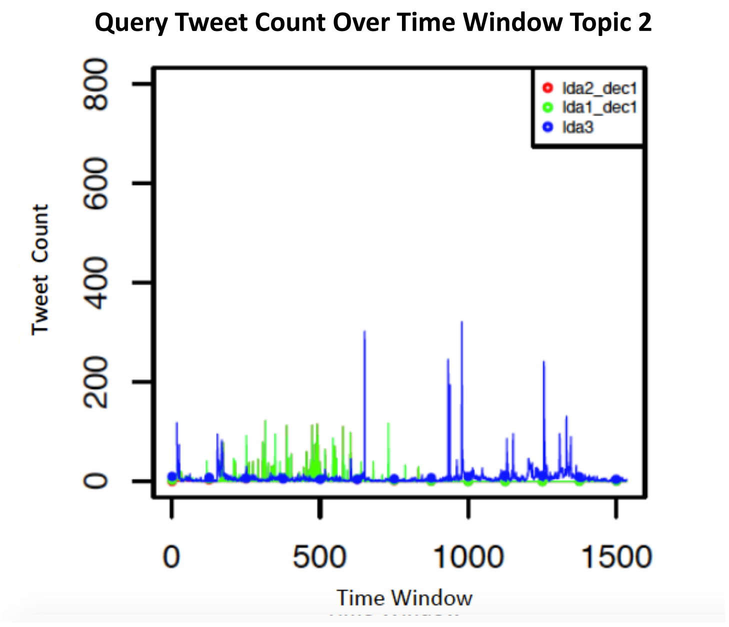

Figure 4.40: Tweet Count Over Time Window of Query Results with Topic 2

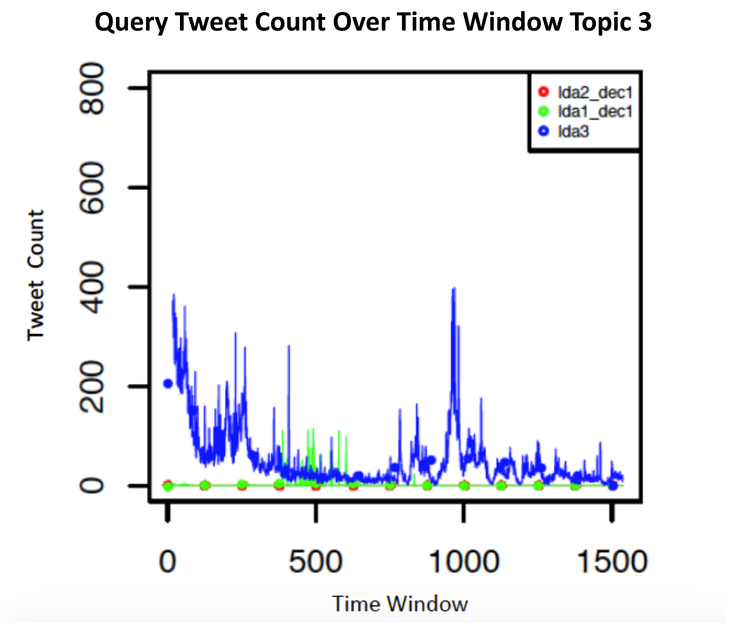

Figure 4.41: Tweet Count Over Time Window of Query Results with Topic 3

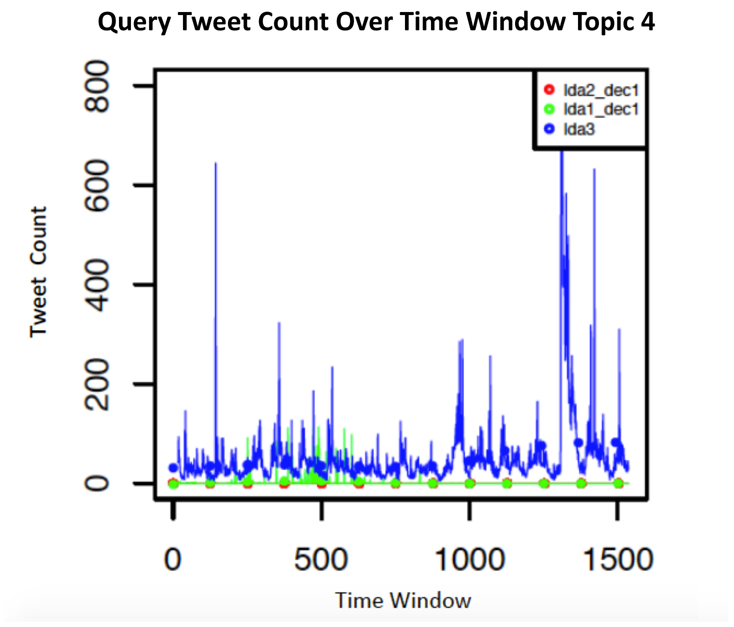

Figure 4.42: Tweet Count Over Time Window of Query Results with Topic 4

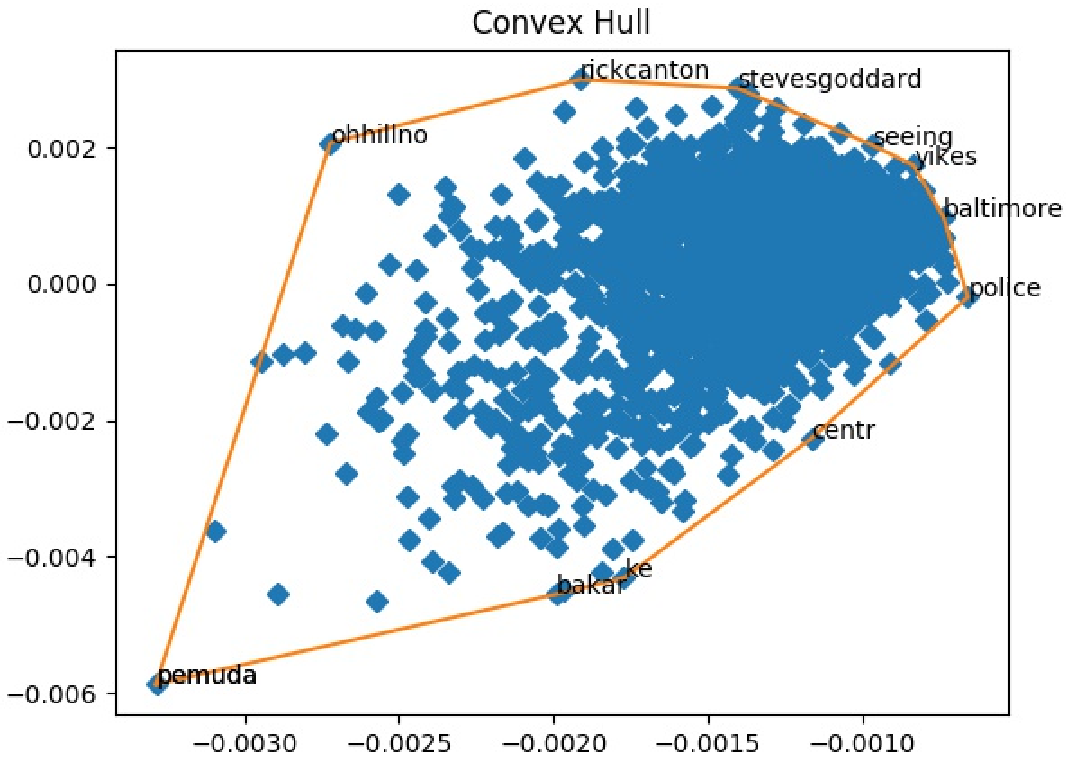

Figure 4.43: Convex Hull Coverage of Query using 1 DEC keyword and 1 LDA keyword

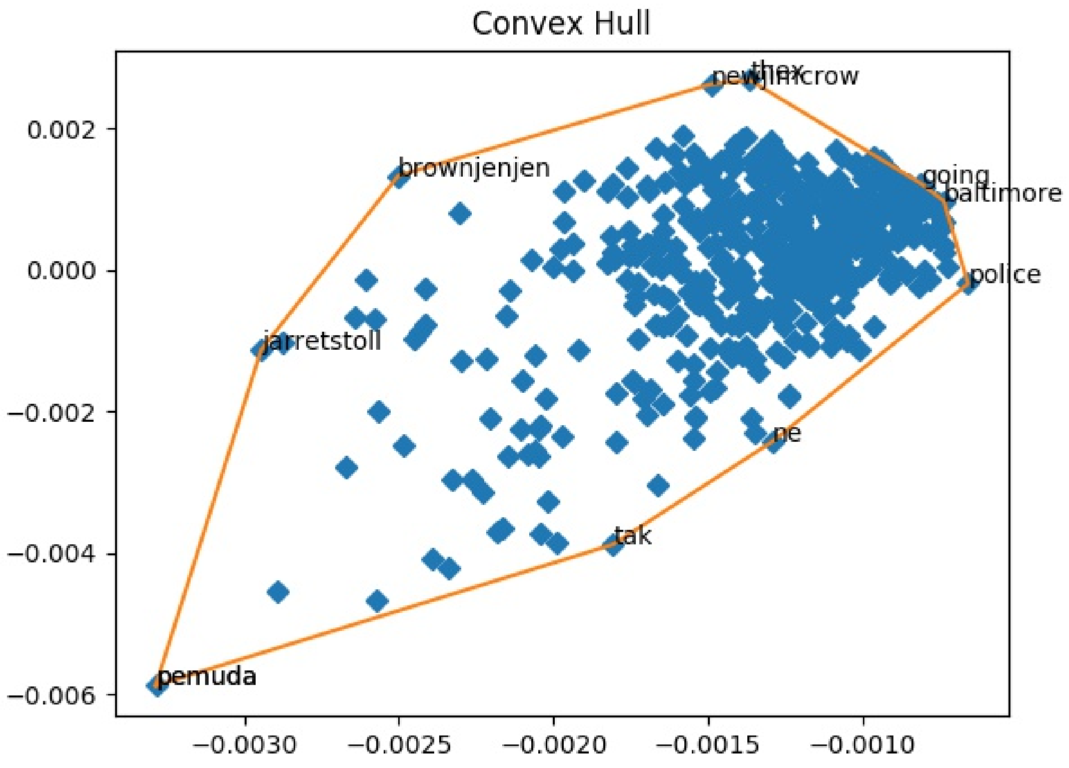

Figure 4.44: Convex Hull Coverage of Query using 1 DEC keyword and 2 LDA keywords

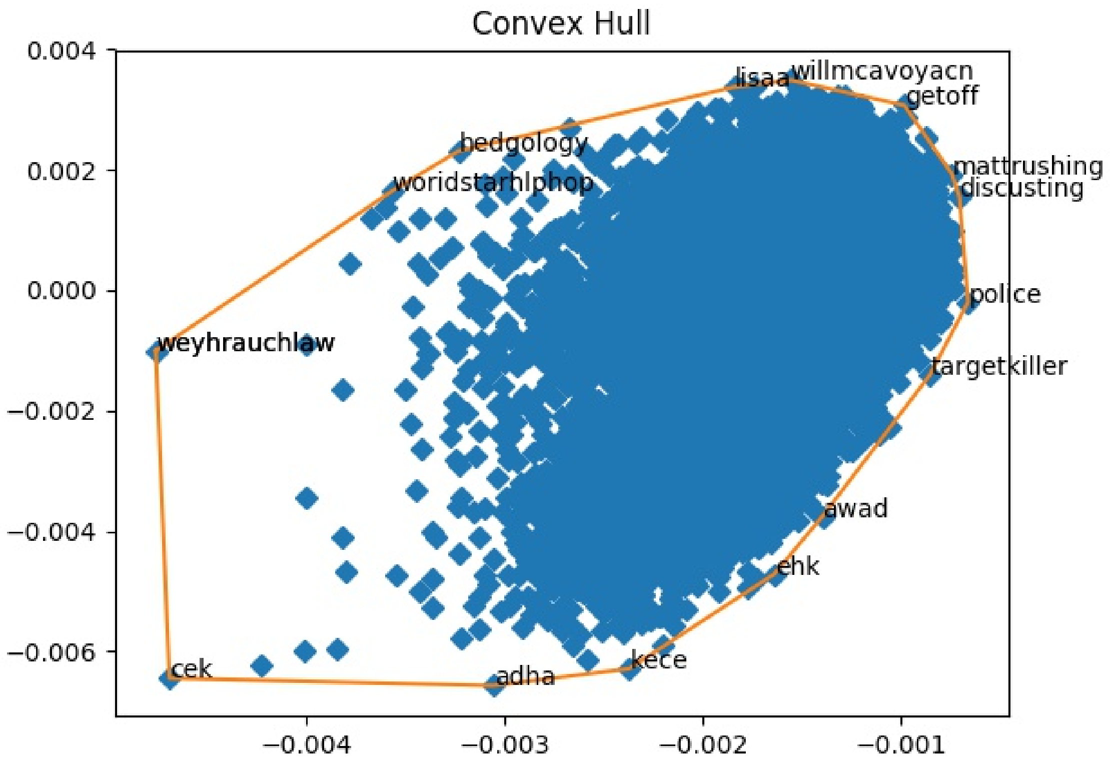

Figure 4.45: Convex Hull Coverage of Query using 3 LDA keywords

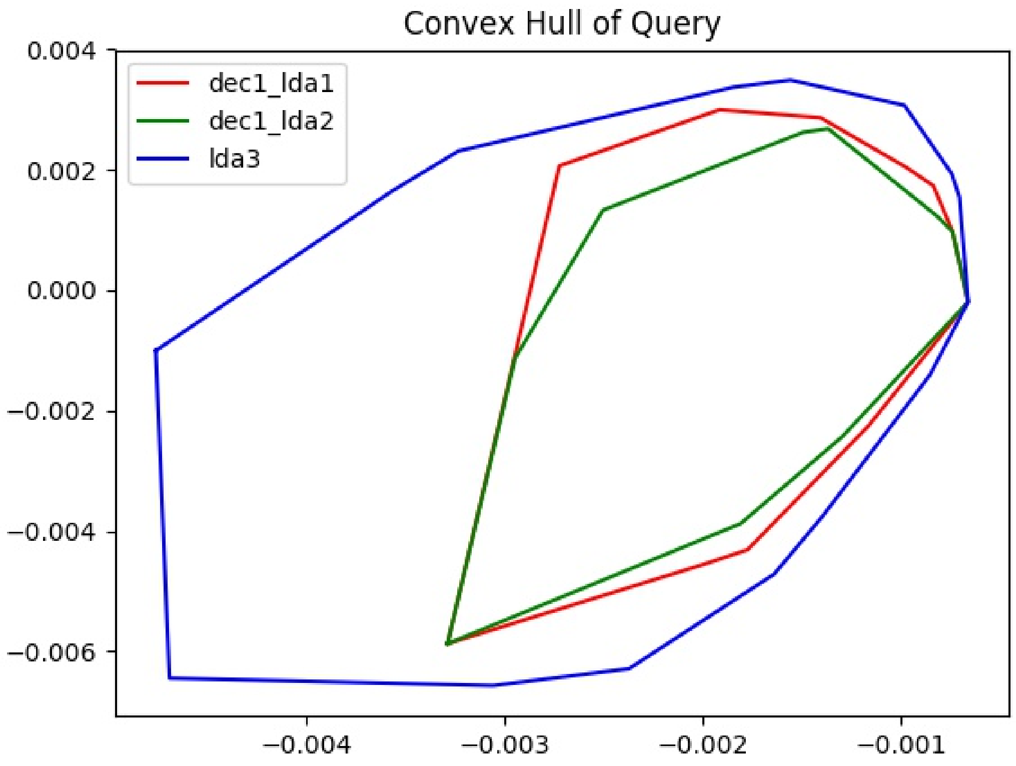

Figure 4.46: Comparison of Convex Hull Coverage Area of Queries

Chapter 5

Summary of Contributions

In this dissertation, we have developed an end-to-end solution to handle the problems of data collection, curation, and analysis in streaming data in automotive domain and social media domain. In this chapter, we provide a summary of our contributions in this dissertation.

5.1 Distributed Data Delivery System Emulations

First, we evaluate a distributed message delivery system for connected vehicle systems in which multiple CV applications ran simultaneously with data transfers between RSUs and different transportation centers. For this purpose, all information flows defined for diverse CV applications in CVRIA were characterized based on time and spatial contexts of data sent from various transportation centers to RSUs and data sent from RSUs to various transportation centers. Our analyses indicate that the message delivery system reduces message redundancies by identifying unique information flows in multiple CV applications. This efficient message delivery system provides a strategy that enables large-scale ingestion, curation, and transformation of unstructured data (including roadway traffic-related and roadway non-traffic-related data) into labeled and customized topics, which can be used by large numbers of subscribers or consumers for various CV applications.

We evaluate the distributed message delivery system by developing a prototype infrastructure using Kafka, which is an open source message broker platform. We then compare the performance of this system to the minimum latency requirement for CV applications. Experimental analyses were performed using two distributed computing testbeds, and the latencies and throughput of the message delivery system for CV applications were examined.

Different experimental scenarios with different numbers of Kafka brokers and consumers were executed to evaluate the message delivery systems performance. The evaluation of this distributed message delivery system shows that the highest total average latency was 7.95 ms, which was measured as the difference between the time when a message was generated by a RSU to the time of its reception by a Traffic Management Center. The highest total average latency, out of all the scenarios considered in our experiments, was found to comply with the recommended USDOT system requirements for CV pilot deployments. However, this latency includes only the delivery time of a message and not the processing times, such as the time required for aggregation and complex data transformation. The results of this study will help ITS professionals to develop a message delivery system to support CV applications and provide an efficient distributed message delivery system that provides many benefits over the current paradigm of centralized message delivery systems.

In reporting the latency measurements in the result section for this component, we use average latency per message. A calculation of a 95% confidence of latencies would be an improvement for the real-time system. In real-life CV applications, the number of consumers often goes beyond the maximum number of consumers we test in our experiments. An experiment scenario deployed in commercial cloud platform which allows a larger number of consumers would be an improvement.

5.2 Topic Modeling on Online Forums

Secondly, in this dissertation, we have investigated the use of an open source web crawl data repository, the Common Crawl, in LDA topic modeling for online forum data. To evaluate how representative Common Crawl is as a sample for extracting LDA topics, we collected both the full data and the Common Crawl sample for 14 subforums from a car user forum. We compared the LDA topics estimated from Common Crawl samples and on the full data both quantitatively and qualitatively. In both cases, the topics generated from Common Crawl and those drawn from the full data are not statistically different from each other. Through our experiments, we demonstrate that Common Crawl does not perform worse than randomly drawn samples from the full data in terms of topic similarity. We also demonstrate the usefulness of topic models for drawing business insights from an online forum through a discussion of the primary and secondary discussion themes estimated from four representative car classes from the Common Crawl data set.

Our results provide evidence that data collected from Common Crawl is a good candidate for LDA topic modeling on online forums. There are several problems associated with collecting data from online forums directly, including the need to develop a customized web crawler, the possibility of one's IP address becoming blacklisted, the size of the data, and the time required to download the full data. Using Common Crawl as a sample of the full data circumvents many of these problems, and our results show that topics estimated from Common Crawl are not significantly different from the full data in terms of topic proportions, and reasonably similar (and not worse) than under random sampling in terms of word rankings.

Our findings are based on the analysis of 14 subforums that represent different car models made by a specific vendor, but our research strategy provides a template that can be used in other domains to evaluate the representativeness of topic models. Future research will need to investigate the sensitivity of LDA topic modeling results on different online forums and product categories. It is also an open question whether results from extensions of LDA, such as dynamic topic models [28] (which account for topic evolution over time) or hierarchical topic models [27] (which allow for topic hierarchies), would exhibit the same sampling properties. Future work will also explore alternative methods for evaluating the similarity of two inferred topic models, such as by combining the mixture and alignment based metrics used here, or examination of document classification.

Moreover, NLP research is increasingly using deep learning systems [66], which are capable of extracting more semantic features from the data. Recurrent neural networks have been proven to capture contextual dependencies. For example, word vector models and deep learning is used to analyze and process textual data. Extensions of our work could investigate if our representativeness estimation can also be applied to these deep learning models.

The dataset we use in evaluating our data curation component contains only car user forums. It would be an improvement to extend the datasets being studied to include other forms of active online forums. The velocity of the streaming data we study in online forum format are much slower than the Twitter data we use in the third component of the dissertation. It would be an improvement to add an experiment which performs query expansion techniques on online forum data as well.

5.3 Business Insights Observed Using Topic Modeing on Online Forums

From a business perspective, topic models enable analysts to infer themes from documents in an unsupervised, automated way. By annotating documents with topics, navigation and processing of the text data is improved. Another important application is the combination of topic models and supervised classification approaches, e. g., to sort documents into a fixed set of categories (e. g., a model or defect category). Topic models provide a condensed document representation that is also well suited as input for classification algorithms. It would be an improvement if we use the topic modeling outputs as features for a classification model that classifies a post into a certain type of user complaint.

5.4 Automated Query Expansion on Streaming Social Media Data

Finally, our proposed Query Expansion method collects less Tweets than the original static stream from the raw stream, while maintaining a higher quality overall of messages collected. As for the expansion of the queries themselves, whoever chooses to implement the system has the ability to fine-tune their queries to require either more topic words, or more novel vector space words. This gives the user a unique ability to toggle the information they retrieve to be more sensitive either to the internal stream, or the external stream.

The intuition of fusing a historical stream's information with the data of our internal stream are intuitively compelling. In addition to providing a wide-scale acknowledgment of past and global events that might not be currently represented in the internal stream, it also allows us to aid against potential bias from a manual keyword selection approach. The system presented in this paper will be able to adjust its various tools, from triggering query expansion, to construction of the feature space, to the method of expanding the queries themselves, to capture a more robust, relevant, and useful set of data from information streams.

The dataset we use in the third component of this dissertation is 16 days of Twitter data which contains two emergent events during this time period. It would be an improvement if we can show the performance of our system on more emerging events in Twitter. Since the Twitter data has very high velocity and each Tweet is short in content, we use a small number of topics for the language modeling of Tweet streams. It would be an improvement to perform our query expansion method on slower velocity streaming data, e.g., online forum data which contains much longer documents than Tweet streams.

Appendices

Appendix A List of CV Data Types Collected from VISSIM Simulation

List of CV Data Types Collected from VISSIM Simulation

Time stamp; date; vehicle ID; vehicle length; vehicle type; vehicle weight; vehicle name; vehicle power; fuel consumption; speed; average speed; speed difference; desired speed; theoretical speed; acceleration; safety distance; headway; desired direction; desired lane; destination lane; lane change; lane number; target lane; link number; destination link; lateral position relative to middle of the lane; leading vehicle; preceding vehicle; number of stops; occupancy; world coordinate front x; world coordinate front y; world coordinate front z; world coordinate rear x; world coordinate rear y; world coordinate rear z; route number; routing sequence; trip chain: activity; trip chain: departure time; trip chain: destination zone; trip chain: minimum duration; trip chain: parking lot number; emissions (evaporation) Hydro Carbon (HC); emissions benzene; emissions CO; emissions CO2; emissions Hydro Carbon (HC); emissions Non-Methane Hydrocarbon (NMHC); emissions Non-methane Organic Gas (NMOG); emissions NOx; emissions particulates; traffic interaction state; total distance traveled; total time in a network; delay time with respect to optimal drive time; destination parking lot; following distance; gradient; total number of queue encounters; queue flag; and queue time.

Bibliography

[1] The Connected Vehicle Next Generation ITS [online]. Available: http://itsamerica.org/the-connected-vehicle-next-generation-its/,2017.

[2] Research Data Exchange (RDE), Federal Highway Administration, U.S. Department of Transportation [online]. Available: https://www.its-rde.net/index.php/data/searchdata,2017.

[3] Connected Vehicle Reference Implementation Architecture (CVRIA) [online]. Available: http://www.iteris.com/cvria/html/applications/applications.html,2017.

[4] Southeast Michigan Connected Vehicle Testbed Resource Guide. U.S. Department of transportation, Federal Highway Administration [online]. Available: http://www.its.dot.gov/testbed/PDF/SE-MI-Resource-Guide-9-3-1.pdf.

[5] Connected Vehicle Infrastructure University Transportation Center. Testbed development [online]. Available: http://cvi-utc.org/test-bed-development/.

[6] Apache ActiveMQ [online]. Available: http://activemq.apache.org/.

[7] IBM Websphere MQ [online]. Available: http://www-03.ibm.com/software/products/en/ibm-mq.

[8] Oracle Enterprise Messaging Service [online]. Available: http://www.oracle.com/technetwork/middleware/ias/index-093455.htm.

[9] TIBCO Enterprise Message Service [online]. Available: http://www.tibco.com/products/soa/messaging/.

[10] MQTT [online]. Available: http://mqtt.org/.

[11] Southeast Michigan Test Bed 2014 Concept of Operations [online]. Available: http://www.iteris.com/cvria/docs/SE_Michigan_Test_Bed_2014_ConOps_-_v1_-_2014-Dec_29.pdf.

[12] Connected Vehicle Applications and Supporting Documentation [online]. Available: http://www.its.dot.gov/pilots/pilots_mobility.htm.

[13] CO-PILOT Background, Assumptions, and User Instructions, Cost Overview for Planning Ideas & Logical Organization Tool (CO-PILOT) [online]. Available: https://co-pilot.noblis.org/CVP_CET/help.html.

[14] Palmetto Cluster, Clemson University [online]. Available: http://citi.clemson.edu/palmetto/.

[15] Report on Detailed Requirements for the INFLO Prototype, Final Report. FHWA-JPO-14-171, US Department of Transportation, December 27, 2013 [online]. Available: `http://ntl.bts.gov/lib/54000/54800/54832/FHWA-JPO-14-171.pdf`.

[16] The connected vehicle test bed: Available for device and application development [online]. available: `http://www.its.dot.gov/factsheets/connected_vehicle_testbed_factsheet.htm`.

[17] Gaia online. `http://www.gaiaonline.com/forum/`, 2017.

[18] Vissim. 5.10 user manual. ptv vision,, July 18, 2008.

[19] Saif Al-Sultan, Moath M Al-Doori, Ali H Al-Bayatti, and Hussien Zedan. A comprehensive survey on vehicular ad hoc network. *Journal of network and computer applications*, 37:380–392, 2014.

[20] Abdel Rahman Alkhawaja, Luis Lino Ferreira, and Michele Albano. Message oriented middleware with qos support for smart grids. In *INForum 2012-Conference on Embedded Systems and Real Time.*, 2012.

[21] Jason W Anderson, KE Kennedy, Linh B Ngo, Andre Luckow, and Amy W Apon. Synthetic data generation for the internet of things. In *Big Data (Big Data), 2014 IEEE International Conference on*, pages 171–176. IEEE, 2014.

[22] Neela Avudaiappan, Alexander Herzog, Sneha Kadam, Yuheng Du, Jason Thatche, and Ilya Safro. Detecting and summarizing emergent events in microblogs and social media streams by dynamic centralities. In *Big Data (Big Data), 2017 IEEE International Conference on*, pages 1627–1634. IEEE, 2017.

[23] P Barnett, D Stening, and P Collinson. Real life experiences during the design, realization and commissioning of m25 london orbital motorway cctv enhancement scheme using digital transmission technology. 2004.

[24] Hila Becker, Mor Naaman, and Luis Gravano. Beyond trending topics: Real-world event identification on twitter. *Icwsm*, 11(2011):438–441, 2011.

[25] Jonathan Bischof and Edoardo M Airoldi. Summarizing topical content with word frequency and exclusivity. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 201–208, 2012.

[26] David M. Blei. Probabilistic topic models. *Commun. ACM*, 55(4):77–84, April 2012.

[27] David M Blei, Thomas L Griffiths, and Michael I Jordan. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *Journal of the ACM (JACM)*, 57(2):7, 2010.

[28] David M Blei and John D Lafferty. Dynamic topic models. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 113–120. ACM, 2006.

[29] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[30] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

[31] JeanCarlo Bonilla and Bharat Rao. Decoding data analytics capabilities from topic modeling on press releases. In *2015 Portland International Conference on Management of Engineering and Technology (PICMET)*, pages 1959–1968. IEEE, 2015.

[32] Christian Buck, Kenneth Heafield, and Bas Van Ooyen. N-gram counts and language models from the common crawl. In *LREC*, volume 2, page 4, 2014.

[33] Paris Carbone, Asterios Katsifodimos, Stephan Ewen, Volker Markl, Seif Haridi, and Kostas Tzoumas. Apache flink: Stream and batch processing in a single engine. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 36(4), 2015.

[34] Feng Chen, Juan Du, Weining Qian, and Aoying Zhou. Topic detection over online forum. In *Web Information Systems and Applications Conference (WISA), 2012 Ninth*, pages 235–240. IEEE, 2012.

[35] Mashrur Chowdhury, Amy Apon, and Kakan Dey. *Data Analytics for Intelligent Transportation Systems*. Elsevier, 2017.

[36] Common Crawl, 2017. `https://commoncrawl.org/`, last accessed August 17, 2017.

[37] Common Crawl website, 2017. Frequently Asked Questions, `https://commoncrawl.org/big-picture/frequently-asked-questions/`, last accessed August 2017.

[38] Francisco Cribari-Neto and Achim Zeileis. Beta regression in R. *Journal of Statistical Software*, 34(2):1–24, 2010.

[39] David Blei, 2017. `https://github.com/blei-lab/lda-c`, last accessed August 19, 2017.

[40] Francois Dion, Ralph Robinson, and Jun-Seok Oh. Evaluation of usability of intellidrive probe vehicle data for transportation systems performance analysis. *Journal of Transportation Engineering*, 137(3):174–183, 2010.

[41] Yuheng Du, Mashrur Chowdhury, Mizanur Rahman, Kakan Dey, Amy Apon, Andre Luckow, and Linh Bao Ngo. A distributed message delivery infrastructure for connected vehicle technology applications. *IEEE Transactions on Intelligent Transportation Systems*, 19(3):787–801, 2018.

[42] Yuheng Du, Alexander Herzog, Andre Luckow, Ramu Nerella, Christopher Gropp, and Amy Apon. Representativeness of latent dirichlet allocation topics estimated from data samples with application to common crawl. In *Big Data (Big Data), 2017 IEEE International Conference on*, pages 1418–1427. IEEE, 2017.

[43] Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

[44] Chris Gropp, Alexander Herzog, Ilya Safro, Paul W Wilson, and Amy W Apon. Scalable dynamic topic modeling with clustered latent Dirichlet allocation (CLDA). *arXiv preprint arXiv:1610.07703*, 2016.

[45] V Gudivada, Amy Apon, and Junhua Ding. Data quality considerations for big data and machine learning: Going beyond data cleaning and transformations. *Int. J. Adv. Softw*, 10(1):1–20, 2017.

[46] Booz Allen Hamilton. Vehicle infrastructure integration (vii) proof of concept (poc) test executive summaryinfrastructure. *Research and Innovative Technology Administration, US Department of Transportation, Washington DC*, 2009.

[47] Yiming He, Mashrur Chowdhury, Yongchang Ma, and Pierluigi Pisu. Merging mobility and energy vision with hybrid electric vehicles and vehicle infrastructure integration. *Energy Policy*, 41:599–609, 2012.

[48] Urs Hunkeler, Hong Linh Truong, and Andy Stanford-Clark. Mqtt-sa publish/subscribe protocol for wireless sensor networks. In *Communication systems software and middleware and workshops, 2008. comsware 2008. 3rd international conference on*, pages 791–798. IEEE, 2008.

[49] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the Association for Computational Linguistics*, 2015.

[50] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 1–12. ACM, 2009.

[51] Yufeng Jing and W Bruce Croft. An association thesaurus for information retrieval. In *Intelligent Multimedia Information Retrieval Systems and Management-Volume 1*, pages 146–160, 1994.

[52] Karen Sprck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21, 1972.

[53] Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*, 2016.

[54] Jay Kreps, Neha Narkhede, Jun Rao, et al. Kafka: A distributed messaging system for log processing. In *Proceedings of the NetDB*, pages 1–7, 2011.

[55] Jess Kropczynski, Guoray Cai, and John M. Carroll. Investigating incidence of common ground and alternative courses of action in an online forum. In *Proceedings of the 15th Annual International Conference on Digital Government Research*, dg.o '14, pages 24–33, New York, NY, USA, 2014. ACM.

[56] Kelsey Lantz, Sakib Khan, Linh B Ngo, Mashrur Chowdhury, Sarah Donaher, and Amy Apon. Potentials of online media and location-based big data for urban transit networks in developing countries. *Transportation Research Record: Journal of the Transportation Research Board*, (2537):52–61, 2015.

[57] Kelsey Lantz, Sakib Khan, Linh B Ngo, Mashrur Chowdhury, Sarah Donaher, and Amy Apon. Potentials of online media and location-based big data for urban transit networks in developing countries. *Transportation Research Record: Journal of the Transportation Research Board*, (2537):52–61, 2015.

[58] Freddy Lécué, Simone Tallevi-Diotallevi, Jer Hayes, Robert Tucker, Veli Bicer, Marco Luca Sbodio, and Pierpaolo Tommasi. Star-city: semantic traffic analytics and reasoning for city. In *Proceedings of the 19th international conference on Intelligent User Interfaces*, pages 179–188. ACM, 2014.

[59] Xiaodong Lin, Rongxing Lu, Chenxi Zhang, Haojin Zhu, Pin-Han Ho, and Xuemin Shen. Security in vehicular ad hoc networks. *IEEE communications magazine*, 46(4), 2008.

[60] Zhiyuan Liu, Yuzhou Zhang, Edward Y Chang, and Maosong Sun. Plda+: Parallel latent dirichlet allocation with data placement and pipeline processing. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):26, 2011.

[61] Yongchang Ma, Mashrur Chowdhury, Adel Sadek, and Mansoureh Jeihani. Real-time highway traffic condition assessment framework using vehicle–infrastructure integration (vii) with artificial intelligence (ai). *IEEE Transactions on Intelligent Transportation Systems*, 10(4):615–627, 2009.

[62] Yongchang Ma, Mashrur Chowdhury, Adel Sadek, and Mansoureh Jeihani. Integrated traffic and communication performance evaluation of an intelligent vehicle infrastructure integration (vii) system for online travel-time prediction. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1369–1382, 2012.

[63] L Magnoni. Modern messaging for distributed sytems. In *Journal of Physics: Conference Series*, volume 608, page 012038. IOP Publishing, 2015.

[64] Kamran Massoudi, Manos Tsagkias, Maarten De Rijke, and Wouter Weerkamp. Incorporating query expansion and quality indicators in searching microblog posts. In *European Conference on Information Retrieval*, pages 362–367. Springer, 2011.

[65] Robert Meusel, Sebastiano Vigna, Oliver Lehmberg, and Christian Bizer. Graph structure in the web—revisited: a trick of the heavy tail. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 427–432. ACM, 2014.

[66] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[67] David Mimno, Hanna M Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 262–272. Association for Computational Linguistics, 2011.

[68] Yi Mou, David Atkin, Hanlong Fu, Carolyn A Lin, and TY Lau. The influence of online forum and SNS use on online political discussion in China: Assessing "spirals of trust". *Telematics and Informatics*, 30(4):359–369, 2013.

[69] Nicolas Nannoni. Message-oriented middleware for scalable data analytics architectures, 2015.

[70] Oded Netzer, Ronen Feldman, Jacob Goldenberg, and Moshe Fresko. Mine your own business: Market-structure surveillance through text mining. *Marketing Science*, 31(3):521–543, 2012.

[71] Dung Nguyen, Andre Luckow, Edward Duffy, Ken Kennedy, and Amy Apon. Evaluation of highly available cloud streaming systems for performance and price. In *2018 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, pages 360–363. IEEE, 2018.

[72] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[73] Brandon Posey, Linh Bao Ngo, Mashrur Chowdhury, and Amy Apon. Infrastructure for transportation cyber-physical systems. In *Transportation Cyber-Physical Systems*, pages 153–171. Elsevier, 2018.

[74] Mizanur Rahman, Yucheng An, Mashrur Chowdhury, Kakan Dey, Yuheng Du, and Parth Bhavsar. Framework for design and evaluation of a distributed data exchange infrastructure for dynamic transit operations. *96th Transportation Research Board Annual Meeting compendium of papers.*

[75] Arti Ramesh, Dan Goldwasser, Bert Huang, Hal Daume Iii, and Lise Getoor. Understanding MOOC discussion forums using seeded lda. *Proceedings of the Ninth Workshop on Innovative Use of NLP for Building Educational Applications*, 2014.

[76] Rajiv Ranjan. Streaming big data processing in datacenter clouds. *IEEE Cloud Computing*, 1(1):78–83, 2014.

[77] Xin Rong. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014.

[78] Maciej Rostanski, Krzysztof Grochla, and Aleksander Seman. Evaluation of highly available and fault-tolerant middleware clustered architectures using rabbitmq. In *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on*, pages 879–884. IEEE, 2014.

[79] Julian Seitner, Christian Bizer, Kai Eckert, Stefano Faralli, Robert Meusel, Heiko Paulheim, and Simone Ponzetto. A large database of hypernymy relations extracted from the web. In *Proceedings of the 10th edition of the Language Resources and Evaluation Conference, Portoroz, Slovenia*, 2016.

[80] Guodong Shao, Seung-Jun Shin, and Sanjay Jain. Data analytics using simulation for smart manufacturing. In *Simulation Conference (WSC), 2014 Winter*, pages 2192–2203. IEEE, 2014.

[81] Xiaolin Shi, Jun Zhu, Rui Cai, and Lei Zhang. User grouping behavior in online forums. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 777–786. ACM, 2009.

[82] Eric Simmon, Kyoung-Sook Kim, Eswaran Subrahmanian, Ryong Lee, Frederic De Vaulx, Yohei Murakami, Koji Zettsu, and Ram D Sriram. *A vision of cyber-physical cloud computing for smart networked systems.* US Department of Commerce, National Institute of Standards and Technology, 2013.

[83] Jason R Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch, and Adam Lopez. Dirt cheap web-scale parallel text from the common crawl. In *ACL (1)*, pages 1374–1383, 2013.

[84] Eduardo Souto, Germano Guimarães, Glauco Vasconcelos, Mardoqueu Vieira, Nelson Rosa, Carlos Ferraz, and Judith Kelner. Mires: a publish/subscribe middleware for sensor networks. *Personal and Ubiquitous Computing*, 10(1):37–44, 2006.

[85] Alex Stolz and Martin Hepp. Towards crawling the web for structured data: Pitfalls of common crawl for e-commerce. In *Proceedings of the 6th International Workshop on Consuming Linked Data*, 2015.

[86] Robert Underwood, Jason Anderson, and Amy Apon. Measuring network latency variation impacts to high performance computing application performance. In *Proceedings of the 2018 ACM/SPEC International Conference on Performance Engineering*, pages 68–79. ACM, 2018.

[87] Olga Vechtomova and Ying Wang. A study of the effect of term proximity on query expansion. *Journal of Information Science*, 32(4):324–333, 2006.

[88] Athulan Vijayaraghavan and David Dornfeld. Automated energy monitoring of machine tools. *CIRP annals*, 59(1):21–24, 2010.

[89] Hanna M Wallach. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 977–984. ACM, 2006.

[90] Hanna M Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. Evaluation methods for topic models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1105–1112. ACM, 2009.

[91] Jiangzhe Wang, Ryuji Wakikawa, and Lixia Zhang. Dmnd: Collecting data from mobiles using named data. In *Vehicular networking conference (VNC), 2010 IEEE*, pages 49–56. IEEE, 2010.

[92] Lucas Wang, Alexander Afanasyev, Romain Kuntz, Rama Vuyyuru, Ryuji Wakikawa, and Lixia Zhang. Rapid traffic information dissemination using named data. In *Proceedings of the 1st ACM workshop on emerging name-oriented mobile networking design-architecture, algorithms, and applications*, pages 7–12. ACM, 2012.

[93] Zhenghe Wang, Wei Dai, Feng Wang, Hui Deng, Shoulin Wei, Xiaoli Zhang, and Bo Liang. Kafka and its using in high-throughput and reliable message distribution. In *Intelligent Networks and Intelligent Systems (ICINIS), 2015 8th International Conference on*, pages 117–120. IEEE, 2015.

[94] James Wright, J Kyle Garrett, Christopher J Hill, Gregory D Krueger, Julie H Evans, Scott Andrews, Christopher K Wilson, Rajat Rajbhandari, Brian Burkhard, et al. National connected vehicle field infrastructure footprint analysis. Technical report, United States. Dept. of Transportation. ITS Joint Program Office, 2014.

[95] Chao Wu, Chunlin Li, Wei Yan, Youlong Luo, Xijun Mao, Shumeng Du, and Mingming Li. Identifying opinion leader in the internet forum. *International Journal of Hybrid Information Technology*, 8(11):423–434, 2015.

[96] Hao Wu, Jiajun Bu, Chun Chen, Can Wang, Guang Qiu, Lijun Zhang, and Jianfeng Shen. Modeling dynamic multi-topic discussions in online forums. In *AAAI*, 2010.

[97] Jinxi Xu and W Bruce Croft. Quary expansion using local and global document analysis. In *Acm sigir forum*, volume 51, pages 168–175. ACM, 2017.

[98] Fangjin Yang, Gian Merlino, Nelson Ray, Xavier Léauté, Himanshu Gupta, and Eric Tschetter. The radstack: Open source lambda architecture for interactive analytics. 2017.

[99] Jijun Yin, Tamer ElBatt, Gavin Yeung, Bo Ryu, Stephen Habermas, Hariharan Krishnan, and Timothy Talty. Performance evaluation of safety applications over dsrc vehicular ad hoc networks. In *Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, pages 1–9. ACM, 2004.

[100] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J Franklin, Scott Shenker, and Ion Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, pages 2–2. USENIX Association, 2012.

[101] Xin Zhao and Jing Jiang. An empirical comparison of topics in twitter and traditional media. *Singapore Management University School of Information Systems Technical paper series. Retrieved November*, 10:2011, 2011.

[102] Tom Chao Zhou, Chin-Yew Lin, Irwin King, Michael R Lyu, Young-In Song, and Yunbo Cao. Learning to suggest questions in online forums. In *AAAI*, 2011.