ON CROSS-DOMAIN SOCIAL SEMANTIC LEARNING

A Dissertation

presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

by

SUMAN DEB ROY

Dr. Wenjun Zeng, Dissertation Supervisor

DECEMBER 2013

The undersigned, appointed by the dean of the Graduate School, have examined the thesis entitled

ON CROSS-DOMAIN SOCIAL SEMANTIC LEARNING

presented by Suman Deb Roy,

a candidate for the degree of doctor of philosophy,

and hereby certify that, in their opinion, it is worthy of acceptance.

Professor Wenjun Zeng

Professor Dong Xu

Professor Yi Shang

Professor Sanda Erdelez

Dedication

This dissertation is dedicated to my family,
whose courage, support and love allowed me
to be a dreamer, to find my way by moonlight
and see the dawn before the rest of the world.

Acknowledgements

Firstly, thanks to my advisor Dr. Wenjun Zeng for his counsel, patience and support. This thesis would not have been possible without his capable guidance, the freedom and encouragement he has given me over the past five years, including the summers when I interned with various companies. I was also fortunate to have an amazing group of mentors from the industry, who strongly influenced not only my thinking but also my efforts and dedication throughout the PhD program, as I strived to connect elements of industrial research to my thesis. Tao Mei -my mentor at Microsoft, Jun Tian - my internship supervisor at Huawei and Gilad Lotan - the chief scientist at Betaworks and a good friend; all of whom I have learned a lot and am still learning from.

Thanks to my thesis committee, Dr. Dong Xu, Dr. Yi Shang and Dr. Sanda Erdelez for being patient and kind enough to entertain my ideas and their advice, guidance and comments. All associated with the Department of Computer Science at the University of Missouri have been very kind to me. Thanks to Jodie Lenser and Sandra Moore for helping me with academic, official and instructor jobs I had. Five years in a college town would not have been possible without strong friendships, thanks to Alex, Abhishek, Simit, Sreenu, Qia, Gary and Tatiana. Their words helped me persevere with my research on several occasions when things seemed down.

Finally, my family was compassionate enough to never ask: 'are you done yet?'. A heartfelt thanks to my mother, who silently supported my ambitious pursuits after dad passed away, my grandfather for helping me think big and my sisters for looking out for my best interests. This thesis is a product of all your faith in me. Thank you.

Table of Contents

List of Tables

List of Figures

List of Equations

"Do not try and bend the spoon. That's impossible. Instead... only try to realize the truth. There is no spoon. Then you'll see, that it is not the spoon that bends, it is only yourself."

- Spoon Boy, The Matrix

On Cross-Domain Social Semantic Learning

Suman Deb Roy

The University of Missouri, 2013

Supervisor: Wenjun Zeng

Abstract

Approximately 2.4 billion people are now connected to the Internet, generating massive amounts of data through laptops, mobile phones, sensors and other electronic devices or gadgets. Not surprisingly then, ninety percent of the world's digital data was created in the last two years. This massive explosion of data provides tremendous opportunity to study, model and improve conceptual and physical systems from which the data is produced. It also permits scientists to test pre-existing hypotheses in various fields with large scale experimental evidence. Thus, developing computational algorithms that automatically explores this data is the holy grail of the current generation of computer scientists.

Making sense of this data algorithmically can be a complex process, specifically due to two reasons. Firstly, the data is generated by different devices, capturing different aspects of information and resides in different web resources/ platforms on the Internet. Therefore, even if two pieces of data bear singular conceptual similarity, their generation, format and domain of existence on the web can make them seem considerably dissimilar. Secondly, since humans are social creatures, the data often possesses inherent but murky correlations, primarily caused by the causal nature of direct or indirect social interactions.

This drastically alters what algorithms must now achieve, necessitating intelligent comprehension of the underlying social nature and semantic contexts within the disparate domain data and a quantifiable way of transferring knowledge gained from one domain to another. Finally, the data is often encountered as a stream and not as static pages on the Internet. Therefore, we must learn, and re-learn as the stream propagates.

The main objective of this dissertation is to develop learning algorithms that can identify specific patterns in one domain of data which can consequently augment predictive performance in another domain. The research explores existence of specific data domains which can function in synergy with another and more importantly, proposes models to quantify the synergetic information transfer among such domains. We include large-scale data from various domains in our study: social media data from Twitter, multimedia video data from YouTube, video search query data from Bing Videos, Natural Language search queries from the web, Internet resources in form of web logs (blogs) and spatio-temporal social trends from Twitter.

Our work presents a series of solutions to address the key challenges in cross-domain learning, particularly in the field of social and semantic data. We propose the concept of bridging media from disparate sources by building a common latent topic space, which represents one of the first attempts toward answering sociological problems using cross-domain (social) media. This allows information transfer between social and non-social domains, fostering real-time socially relevant applications. We also engineer a concept network from the semantic web, called semNet, that can assist in identifying concept relations and modeling information granularity for robust natural language search. Further, by studying spatio-temporal patterns in this data, we can discover categorical concepts that stimulate collective attention within user groups.

Using these various disparate data from different domains, my dissertation aims to assert that intelligent learning is a mixture of two parts: combinatorial knowledge representation from diverse data, and transferring the gained knowledge appropriately to tackle a new task which could not be solved elegantly without the synergy. In summary, this work demonstrates that traditional learning models for classification, prediction and recommendation (such as Support Vector Machines, Latent Dirichlet Allocation, Genetic Algorithms, Conditional Random Fields, Decision Trees, Path Analysis, Probabilistic Automata) can be boosted by algorithmically transferring related social and semantic data from cross-domains.

CHAPTER 1: INTRODUCTION

Over the last decade, two ideas have fundamentally disrupted how humans attain information. The first involved engineering powerful search algorithms, which can quickly parse through the plethora of online resources for contextual facts. Online search allowed users to gain instant, accessible and often expert information regarding topics that they or their real world social circle often lacked. Although automated search technology might have first seemed to distance humanity from the direct need of social/inter-personal advise, soon another technology was born that gave ordinary users the power to not only to publish and share information online, but to become content creators themselves. Online Social Networks and Social Media revolutionized information diffusion in societies, compelling traditional media, advertising and technology companies to honor the wisdom of the crowds.

The idea of online search and online social networks is erected on two separate factors. Search technology is driven by an algorithm's understanding of a user's query intent. The user intent is indicated by the meaning of the search query, also known as its *semantics*. On the other hand, social media is built on users sharing information with each other where millions of micro-level user interactions give rise to macro level social media trends. The shared data inherently bears a *social* footprint by means of the network motif where it was shared or edges through which it spread to new users. This chapter begins with the proposition that intelligence includes processing data from different domains and understanding cross-domain associations among data.

1.1 Data and Intelligence

The fundamental trait of what we perceive is that it somehow generates data. We see objects because it reflects light, millions of photons as data. We hear music because of audio signal data. As we perceive more data, our mind starts detecting patterns in the data. Thus, humans gain experience, knowledge, wisdom, insights, ability at problem-solving, drawing analogies and much more. All these abilities make us intelligent. Current computational methods can replicate some of these abilities in isolation, including audio/image/video data analysis, text mining etc etc.

Intelligent algorithms of the future must understand two attributes of data: its semantics and its social nature. The task is challenging, since the audio/video/textual data has associated graphs - social and semantic. Also, the data exists in various domains of the Internet (e.g., social streams, video sharing platform etc.). The different domains give rise to various non-related attributes to the data, e.g., making it real-time, noisy in grammatical construction, too huge in size to process in one machine, or having variable interpretation based on context. Therefore, a bunch of traditional algorithms fail to scale to these new properties of the data, which has been popularly termed as, Big Data.

Human intelligence is a product of evolution. From Darwin's theory of evolution, we know that survival is directly correlated with adaptation to change. Any intelligent agent's adaptation to change is dependent on how quickly it can modify its action strategy in a new environment. The choice of strategy in the new environment further depends on how quickly it can learn about the new environment itself. Thus, the key to intelligent adaptation is learning, and transferring the learned information into successful actions required to accomplish a new task. In a similar fashion to most adaptable intelligent organisms, *machine learning algorithms of the future must adapt to the features of this new social and semantic data existing in cross-domains over the Internet*. In the next three sections, we explore three issues key to this dissertation, (a) the advent of big data, (b) their feature disparity due to the existence of this data across domains, which essentially complicates designing of a combined learning strategy, and (c) how new algorithms that transfer information from one domain to another can be built beyond existing single-domain machine learning.

1.1.1 Big Data

The advent of the Internet, faster processors, cheaper tablets and powerful mobile technology has enabled humanity to interact with each other and the surrounding environment with unprecedented elasticity. Our online activity, collected through ubiquitous information-sensing digital devices, creates a digital world around us that is getting progressively more local. Through these networked devices, we communicate with human and artificial intelligence in various ways on a daily basis, ranging from our social network activities to every web search we query. In this information ecosystem, there is also the pervasive presence of systems that record each of our digital correspondences. Such correspondence could include social network status updates, surveillance camera recordings, uploaded videos in YouTube, searches on mobile phones, the GPS tag in an uploaded photo etc. As of 2011, there 2.4 billion online users. Each individual is generating data every time he/she interacts with the networked digital world, resulting in massive amounts of data being generated. This has created an explosion in the amount of digital data available, so much so that 90% of the world's current digital data was created in the last two years!

Thus was born Big Data - a compilation of large complex data sets collected from various sources and information-sensing domains. Examples of Big Data include, but are not limited to, web logs, sensor network data, RFIDs, social network and social media data, organic Internet data (web documents), atmospheric science data, genomic databases, surveillance data, healthcare and medical records, video archives and e-commerce data.

Big Data has unique characteristics which make search, analysis, interpretation and visualization of such data considerably challenging using traditional database tools. The first challenge is to store and analyze the large volume of data generated. Consider the social micro-blog Twitter, where users generate almost 600 GB of tweets per day. Secondly, the Big Data is often in motion, having velocity or stream inflow. For example, Twitter generates around 300 tweets/second under normal operations. In peak circumstances (e.g., the Euro 2012 soccer finals), Twitter has been known to generate almost 15000 tweets/second. This means analysis and prediction models needs to be latency-sensitive so that the data change rate can be balanced against the decision window. Thirdly, the breadth of interpretation of such data varies largely with context. This has profound implications on predictions involving product strategy, brand sentiment etc. Finally, the data has significant variety depending of source of creation, arriving in different formats including unstructured data. In other words, each data set originates in some domain (e.g., social streams, video archives, semantic web) contributing to domain-specific features. Fig. 1 illustrates these key properties of Big Data.

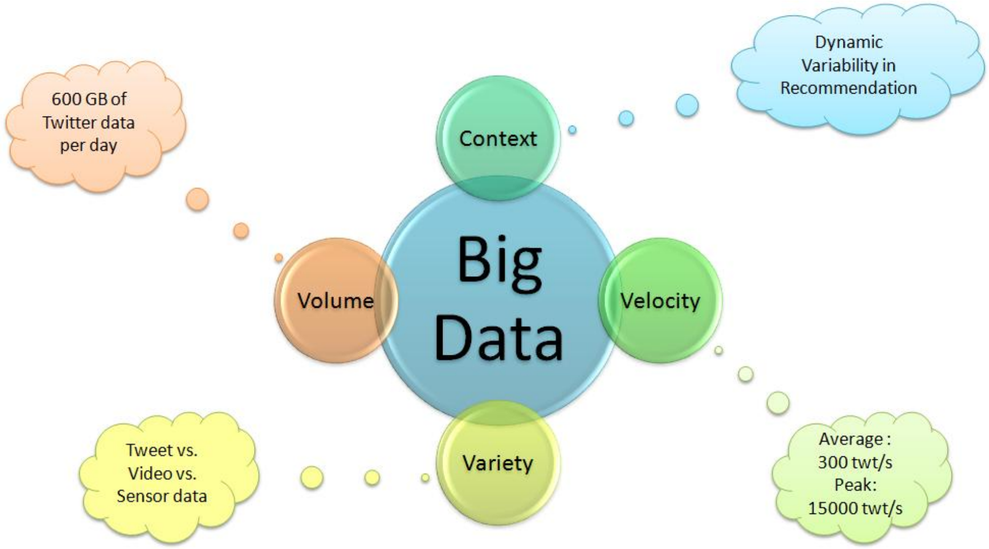

600 GB of Twitter data per day
Context
Dynamic Variability in Recommendation
Volume
Big Data
Velocity
Variety
Tweet vs. Video vs. Sensor data
Average : 300 twt/s Peak: 15000 twt/s

Figure 1: Properties of Big Data.

In spite of all the challenges involving intelligent analysis of Big Data, the promise it holds is immense. Big Data could be leveraged to develop and improve applications ranging from high frequency trading, real-time fraud detection, social media based recommendation, network traffic shaping and popularity based content caching, activity based advertisement, transportation and social gaming to name a few. Moreover, there is a non-deterministic angle to Big Data: it has the potential to facilitate exploratory search, model based analytics and support expert systems like Watson. Essentially, unlike usual data, Big Data allows a researcher to explore what questions to ask.

Let us consider a few of such questions, e.g., is it beneficial to connect Big Data existing in different domains? Can data from one domain explain the observed behavior of data in another domain? If yes, then how can we go about in building frameworks that allow connections between Big Data from different sources for seamless information transfer[1]? Will a synergy between data from cross domains help us in better prediction

[1] Seamless information transfer occurs when each of the four challenges (volume, velocity, variability and context) in handling Big Data is tackled in every domain involved in cross-domain learning

than existing models based on data from a single domain? How can we make such cross-domain learning techniques scalable? Will cross-domain learning allow us to build ingenious new applications that could not be supported by traditional single domain approaches?

Questions such as the ones stated above are not being tackled currently in the Big Data community, since researchers are predominantly occupied with improving application performance in a single domain. However, as we will show in this work, the key to improving prediction-based performance in one domain is by understanding why the domain data is behaving the way. The crucial factor in understanding data behavior is realizing that these domains are not independent, but strongly causal. We will call two domains causal if the generated data in one domain directly or indirectly affects the generated data in another domain. Understanding causality is one of the vital ingredients in envisioning a cross-domain learning systems, since the main aim of learning from one domain is so that we can make intelligent predictions in another domain. Causality analysis of domains can help learning cross-domain models that depict true data generating mechanisms and improve predictions that account for changes in the conditional distribution of the target variable.

Therefore, the real objectives that need fulfillment to fabricate cross-domain learning approaches are: (1) Detect causal domains, (2) Develop a framework that allows for seamless information transfer between these domains, and (3) Discover novel applications supported by the synergy between the two cross domains, which are generally realizable exclusively by cross-domain learning approaches.

1.1.2 Cross-Domain Data

Media on the Internet is unevenly distributed depending on platforms, popularity and bias. Its power is limited by the domain where it originates. For example, video popularity is usually judged by view count [25], but not by how trending the video topic is. We observed that viral videos, which spread by sharing, do not usually contain any common topics with the trending topics in social media. Another example is that Twitter users can only see related media shared in Twitter, but not from external sources. This compels users to perform unguided search in external resources manually. Such video sites are more often than not filled with an explosion of video/image information. Thus, we feel the need for better cross domain media recommendation systems to be a key constituent to *social search* and empower online media. Such media are collected from cross domain resources, and are not constrained by the bias of the social site or the analytics of the video publishing site. Thus, incorporating social knowledge into traditional media applications requires cross-domain information transfer, which contains the *wisdom of the crowds*. It is therefore important to develop a cross-domain knowledge transfer mechanism from the crowd-sourced social domain to traditional media (video) domain.

There are various kinds of media on the Internet - some publish interest specific information, some share in real time and some provide crowd sourcing options. Although multimedia has become a primal entity on the Internet beating text-only content (like XML), it is essentially distributed disparately, e.g., tweets about Haiti Earthquake in Twitter and videos about the same event in separate video publishing sites like YouTube are potentially disconnected, unless users explicitly link them. The socialized power that each kind of media can enhance others has not been fully realized. For example, do trends detected in social streams have latent relations with user search patterns in video publishing sites? If such similar associations can be drawn and analyzed, user experience in one media domain (e.g. social stream) can be enriched by virtue of information in another media domain (e.g. video publishing). This can help solve some problems that purely multimedia techniques cannot accomplish elegantly [22], such as better modeling of video popularity using socially trending topics/events.

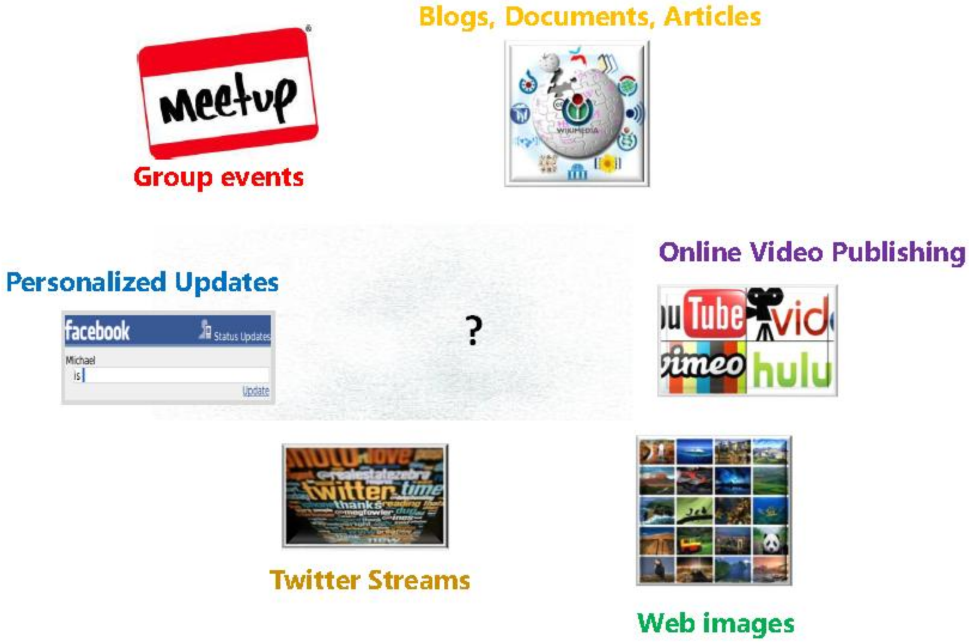


Figure 2: Existence of media resources in disparate domains on the Internet.

Social Media

Social Media gives ordinary people the power to be content creators and information disseminators. This information is embedded in multimedia shared across social networks, containing valuable indications about various facets of human life - what captures our attention, our sharing biases and digital traces we abdicate.

Social media has become a disruptive platform for addressing many multimedia problems elegantly[4]. It has penetrated every realm of business and academia (marketing, advertising, journalism, broadcast, stock markets etc.) and its existence is ubiquitous. Moreover, remarkable insights can be extracted from social media. For example, real-time social data is being utilized in a number of scenarios - from visualizing political activity and flu outbreaks [80, 98], forecast and prediction to sentiment detection [99] and emergency advisory systems [97].

Social media has also largely affected existing models of communication and information retrieval. Akamai, a content-distribution company, recently reported that traffic from social sites has multiplied by five times in 2012, capping at 1 million requests per second. This has strong implications on traffic shaping for computer networks. Audiences are turning to social sites to ingest traditional news, e.g., 78% of web traffic to the New York Times website comes from just Facebook and Twitter combined. The rest 22% arrives from the organic web. Existing political and non-profit campaign prediction models, search tools and media recommendation has also changed to incorporate the massive amounts of social data generated every day.

One aspect of social micro-blogs like Twitter [80] is its short text format, which is fast and real time. Thus, social media data hits the web faster than articles, images, or videos on the same topic. In the chain of *digitization of a real-world event* (Fig. 3), social stream data like tweets from Twitter are often the source of breaking news. In fact some famous breaking news in the last year has been captured first as tweets, including the death of Osama Bin Laden, the Hudson plane crash, announcement of the royal wedding etc. This property can be leveraged to resolve interesting real time applications, e.g. semantic video indexing [19] and topic evolution and topic tracking [83].

It remains challenging to extract relevant and valuable information from social streams (e.g., Twitter) and correlate social media across different domains. One reason is due to the noisy nature of social streams. For example, each tweet in Twitter is limited to 140 characters. This severely hinders techniques based on 'bag-of-words'. The tweets are often noisy and improperly structured in grammar/syntax, which makes them difficult to process using standard Natural Language Processing tools. An additional concern is that the incoming data of tweets typically arrives in high-volume streams (bursty traffic) and thus, algorithms mining them must scale in learning (for decomposition methods based on Normalized Cut are too slow to scale). Efforts such as Social multimedia signal processing aims to transform the noise-like phenomena in social media into signals useful for building novel socially-aware multimedia applications and targeted advertising techniques, and exploring new marketing methods and a fresh way to look at the existence of multimedia in online social networks.

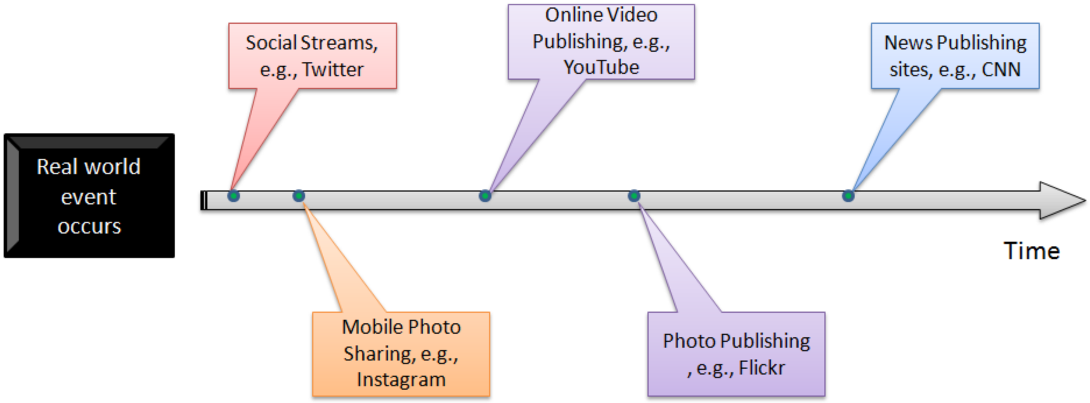

Social Streams, e.g., Twitter
Online Video Publishing, e.g., YouTube
News Publishing sites, e.g., CNN
Real world event occurs
Time
Mobile Photo Sharing, e.g., Instagram
Photo Publishing , e.g., Flickr

Figure 3: Chain of Digitization of real-world event.

Semantic Web Data

The semantic web is the next stage of evolution of the world wide web (WWW), where computers will not only be able to exchange data based on standard formats and protocols (like HTML), but also interpret contextual information in the data in an automated fashion, allowing machine readable assistance to users in making sense of the huge amount of information on the web [40]. The idea has been popularized as Linked

Data. Unlike the WWW where computers are connected, the Semantic Web is built on top of WWW where data is connected, or linked - hence the name Linked Data.

The unit of the Semantic Web is a data model called the Resource Description Framework (RDF), which is similar to the classic entity-relationship conceptual model of organizing data. Each RDF entry is composed of three parts in the form of a triple *<resource><property><value>*, where the *<value>* is the Universal Resource Identifier (URI) of the resource, the describes an attribute of the resource and the represents the specific object value of the attribute. An example RDF is

<http://dbpedia.org/resource/Abraham_Lincoln>

<http://dbpedia.org/ontology/birthPlace> <http://dbpedia.org/resource/Kentucky>

where the resource is 'Abraham_Lincoln', the property is 'birthPlace' and the value of that property is 'Kentucky'. Thus, RDFs represent a subject-predicate-object expression for some resource on the Web. The general query language for RDF datasets is called SPARQL [40], which is a SQL like language to traverse through RDF resources.

Built on top of RDFs, is a family of formal languages called Simple Knowledge Organization System or SKOS. It represents higher concepts than mere entity resources, specifically thesauri, classification schemes, taxonomies etc. The system has one core, called the SKOS core and many SKOS extension based on the field of classification. The core represents common concepts found in most fields. Concepts are organized in hierarchies.

Natural Language Data

Although understanding the rules of natural language is predominantly a branch of linguistics, artificial intelligence plays a big role in this task by extracting similar patterns in sentences, revealing the rules of the grammar itself. Irrespective of the content of the sentence, a certain set of grammatical rules must be followed in constructing the sentence, without which, the semantic information in the sentence cannot be interpreted by the reader [47]. Fundamentally, every language has three key parts: (a) a lexicon which is similar to the vocabulary, (b) a parser than can show dependency of words and (c) a grammar, which upholds the lexical relationship in the sentence and necessary to understand the internal representation.

Natural language texts occur almost all over the Internet. In fact, apart from tweets, videos, music or animation, most other written content on the web is in natural language. Thus, the importance of extracting semantic meaning from natural language is of immense importance. Unfortunately, humans often tend not to follow standard rules of sentence constructions [57]. This will often confuse an automated algorithm trying to extract semantics of the natural language sentence [52]. Therefore, it is key to find robust algorithms which can understand that diverse constructions of natural language sentences might still bear the same user intent, as is often encountered in web search queries [53].

Multimedia Video Data

A video is a sequence of images played at a particular frame rate, creating an impression of continuous moving image. Video data has several attributes, including aspect ratio that describes the dimensions of video screen and video picture elements, compression scheme which balances the quality vs. increased data rate, quality which can be measured with metrics like PSNR etc.

With the advent of video publishing sites like YouTube [25], users can upload videos captured with the cameras onto the Internet. Such videos usually also come with meta data, such as the title, tags, description etc. This metadata is critical for video search engines to retrieve relevant videos based on user queries [26]. Extracting content information from the video signal itself (also called visual words) is another way to estimate the context of the video. However, slight variations in object recognition can mislead the visual word extractor, thus, many video sites utilize only tags as keywords for video recommendation.

Spatio-Temporal Data

Finally, a new type of data is emerging in research nowadays called Spatio-temporal social data. This data is essentially a time series of some signal generated by some entity or groups of entity, which has a spatial component to it [83]. Thus, the signal spreads out not only in space but also in time. The data could just be in the form of a series of time stamped locations.

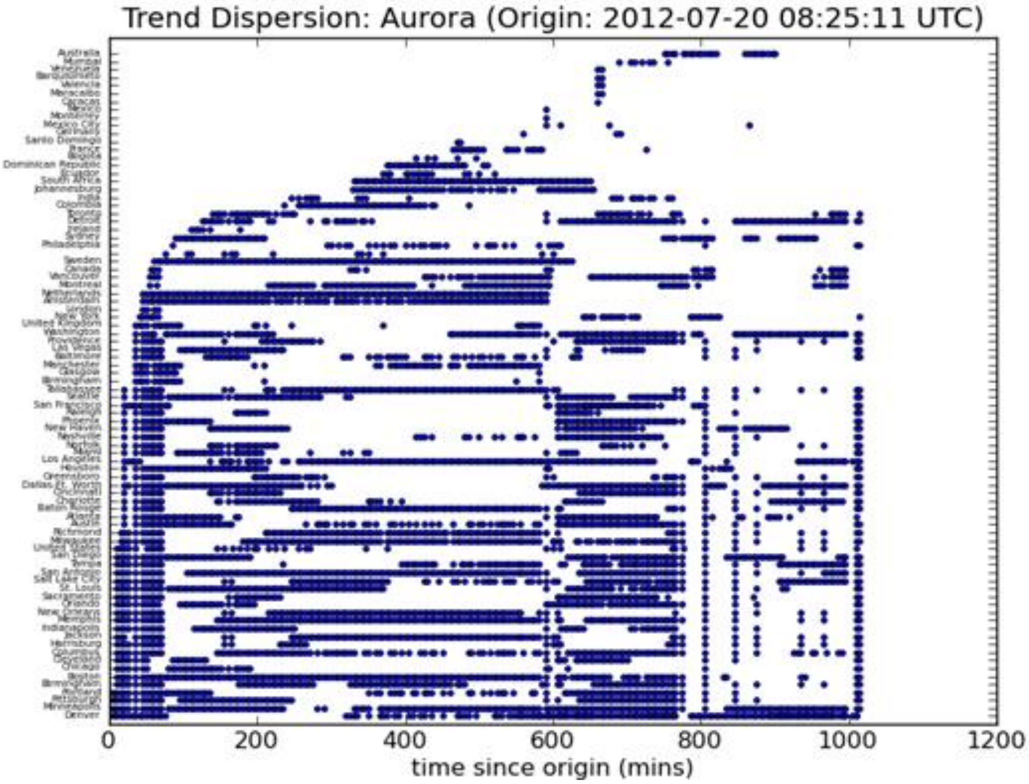


Figure 4: Dispersion plot showing spread of the geographical Aurora trend.

A simple example of spatio-temporal trends is the geographical spread of a Twitter trend. When people in particular geographical locations talk increasingly about some topic on Twitter, it is captured as a trend for that location [81]. This trend has an origin in a particular location, and then spreads to other locations as more and more people begin to talk about it in Twitter. Shown in Fig. 4, is the spread of the Twitter trend #Aurora, which reflects the discussion about the tragic theatre shooting in Aurora, Colorado.

1.1.3 Learning Algorithms

A learning algorithm discovers patterns from data, and uses it to make predictions or classifications on new data. The data must be independent and identically distributed. It must also maintain the same distribution throughout its period of generation, barring which, the learning algorithm must adapt to the new distribution [7]. The learning algorithm is embodied by the classifier - a program that classifies data based on previously seen patterns. Two key tasks that any learning algorithm must accomplish are representation of the data using features and generalization. The latter is the ability to accurately predict class or label of unseen data.

Learning algorithms belong to the field of artificial intelligence called Machine learning [7]. It deals with algorithms that learn from experience in discovering conjectures and knowledge from specific data, rooted in statistical and computational principles. Given the algorithm has seen an instance of some data, with certain features and a known class, it can make intelligent prediction about the class of a new instance by reading the features of the latter.

The taxonomy of learning algorithms include (a) supervised learning, where the learning function maps inputs to desired outputs (also called labels), (b) Unsupervised learning, where the learning algorithm clusters similar data into groups since labels are not known in advance, and (c) Semi-supervised learning, where both labeled and unlabeled data is utilized to build a classifier.

Many learning algorithms are known to researchers, including Decision Trees [44], Conditional Random Fields [62], Artificial Neural Networks, Support Vector Machines, Clustering, Genetic Algorithms etc. [100]. Applications of such algorithms have found wide adoption in academia and industry for tasks such as computer vision, natural language processing, stock market analysis, computational advertising, information retrieval, sentiment analysis and recommender systems.

Some major drawbacks of current learning algorithms include their requirement to have identical distribution of features in both the training and test data, non-portability across multiple domains, too much reliance on statistics only causing over-fitting etc. (ANN) , inability to be implemented efficiently over a cluster (topic models). In this thesis, we shall augment these algorithms to transfer information between cross-domain media.

1.2 Motivation

There are several challenges in building learning algorithms that can scale big data constraints, learn from disparate feature sets of cross-domain data and transfer information among domains. Below are mentioned some key challenges addressed in this research.

Challenges in Learning from Social Streams

1. Dealing with the noisy, incomplete, ambiguous, and short form nature of social stream data. Each tweet is limited to 140 characters and often

improperly structured in grammar/ syntax. Traditional language model (e.g., Bag-of-Words) would fail to scale up with such kind of data..

2. Social streams are real-time, trends appear and disappear within minutes. Twitter data is generated often at an average rate of 5000 tweets/ second, requiring the learning algorithm to scale (learn topics from one chunk before next chunk appears) with the incoming burst of data.

Challenges in Transferring information from Social to Video Domain

3. Developing an unified framework to combine the social and multimedia feature information which has different domain-specific properties.
4. We need to align combinatorial features across two (cross) domains of data. For example, tweets from Twitter has a different feature set compared to videos from YouTube. Thus, we need to detect common feature that can describe both data.
5. Formulating a transfer learning algorithm that can seamlessly propagate the knowledge (i.e. social topics) mined from the crowd sourced social streams to the video domain in real-time.
6. The scaling up and adaptation of the transfer learning algorithm to the ever bursty real-time nature of the social streams.

Challenges in Learning from the Semantic Web

7. Building a graph database (Concept Graph) from Semantic RDF data that can relate concepts, and not just resources.
8. Developing metrics that utilize the Semantic Web to quantify the semantic coherency in a topic or collection of entities.
9. Using the network properties of the Concept Graph to explore various aspects of information, such as granularity or communities.

Challenges in Transferring Semantic Information from Concept Graph to Natural Language Search Queries

10. Designing a canonical form for every natural language query, that can directly interface with the graphical structure of the Concept Graph.
11. Performing query expansion (new words related to the query words) using the links in the Concept Graph.

Challenges in Learning from spatio-temporal social trends

12. Designing a criterion that models attention in social network user communities.
13. Developing metrics that characterizes various attributes and the spread of spatio-temporal social trends.

1.3 Contributions

This dissertation makes contributions to areas of Computer Science that deal with learning from cross-domain data, aiming to string together an several approaches of simultaneously learning from data which are generated in disparate Internet domains. The key idea is that data in various domains can be beneficial to other domains, and if we can learn intelligently from that data, align combinatorial features across domains then efficient information transfer can be realized (Fig. 5). We evaluate our theory on social media data, multimedia data, natural language search data, spatio-temporal social trend data and semantic web data.

The main contributions of this dissertation are as follows:

(1) We propose the concept of bridging social media and traditional media from disparate sources by building a common latent topic space, which represents one of the first attempts toward answering sociological problems using cross-domain social media.

(2) We propose *SocialTransfer*, a novel transfer learning framework based on efficient graph spectra analysis by seamlessly integrating the topic space learned from social stream in real time.

(3) We develop several socially aware media applications based on *SocialTransfer*, which could otherwise hardly be realized in conventional approaches, and evaluate through large scale real-world social media data.

(4) We construct a graph database from the Semantic Web called the Concept Graph, that can be used to categorize and extract concepts from texts from various data domains on the Internet. It can further be used to model the semantic coherence within a group of entities (text or media).

(5) We show how the semantic concept graph can enable us to accomplish cross domain tasks such as natural language search, predict movie profitability and extract cultural patterns from journalistic publishing.

(5) We propose a cognitive model to tackle the noise in natural language query constructions. We computational develop this model and present results proving that the computational cognitive model is closer to human intent in query constructions.

(6) We devise metrics that characterize the various aspects of spatio-temporal social trends. Using these metrics, we build an automaton that can predict the attention of various user communities on Twitter with respect to topics of interest.

1.4 Organization of the Dissertation

Chapter 1 introduces readers to the three topics this dissertation will focus on time and time again: big data, cross-domain data and learning algorithms. It also provides motivation for the importance of this research and the considerable challenges we need to overcome to build scalable solutions.

In Chapter 2, we discuss the technical background required to comprehend this research, focusing specially on topic modeling algorithms, transfer learning, natural language processing and search, semantic networks and various multimedia applications that could be improved with a social flavor. In this chapter, we want to clearly point out the existing and state of the art research in these fields, so that our contribution in the future chapters is clear.

In chapter 3, we encounter our first real world dataset - Twitter tweets, and focus on a scalable way to learn topics from this social stream data. In doing so, we portray the novel idea of an intermediate topic space between various media domains. We also mention our proposed Online Streaming LDA algorithm for real time topic learning from social streams.

In Chapter 4, we present our novel transfer learning algorithm, called *SocialTransfer*. We argue the social signal penetration theory on which our transfer learning scheme is based. Lastly, we demonstrate three novel socially-aware multimedia applications built on top of the *SocialTransfer* framework. Here, we deal with both social and video data.

In Chapter 5, we dive into semantic data for the first time. Our goal in this chapter is to build a semantic network from existing RDF data. We further show how this semantic network is useful in various categorical classification tasks on real world data.

Moreover, the semantic network can be leveraged to measure the semantic coherence of a group of words.

In Chapter 6, we aim to find semantics in natural language search queries. In order to do so, we leverage a cognitive model of semantic information understanding. Using the concept network and a learning technique called Conditional Random Fields, we recreate the cognitive model and use it to better understand natural language queries.

In Chapter 7, we utilize spatio-temporal social trends to model the attention of a group of users in the social network. The learning model which predicts the attention of an user group with respect to some trend is called the Attention Automaton.

Finally, in Chapter 8, we conclude the dissertation, discussing the how our work tackles the cross domain and the information transfer issue in the current state of the Internet. We also suggest applications of our work and future research in this field.

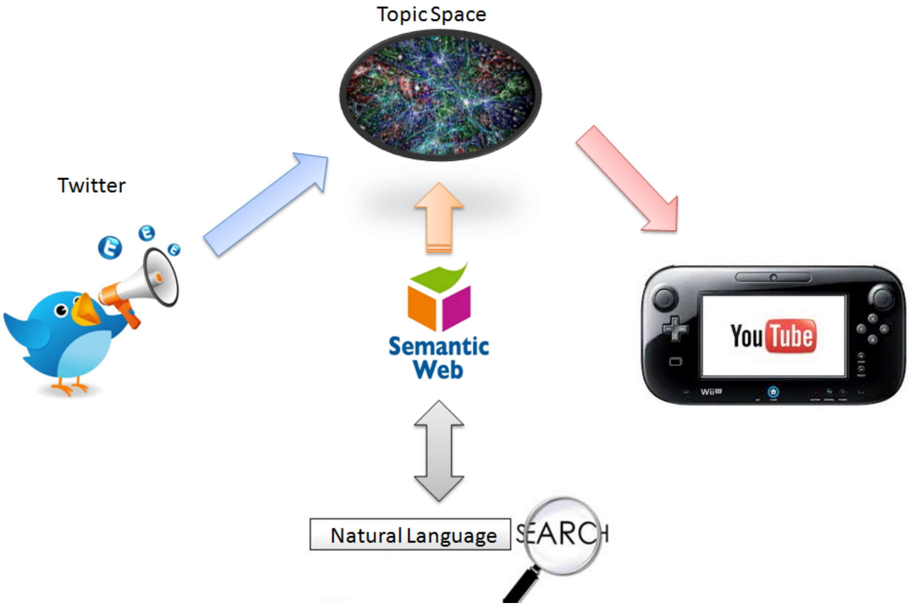


Figure 5: Information Transfer across Cross-Domain Data using an intermediate topic space.

CHAPTER 2: BACKGROUND AND RELATED WORK

In this chapter, we focus on the technical background required prior to diving deeper into this dissertation. We shall also mention key state-of-the-art research results so that it is easy to distinguish our contributions from existing previous work. Since we deal with different kinds of data, we shall focus on technologies that have been studied in relation to some of this data. A major portion of these technologies belong to the field of intelligent information understanding, machine learning and recommendation systems.

We begin with topic modeling, the art of extracting topics from a collection of documents. Then we discuss a specialized branch of machine learning called transfer learning, which is applicable when the task to be solved/automated belongs to a domain where the labeled (training) data is not. However, some other domain has labeled data and thus, we must transfer information between domains. Following this, we discuss the growing area of natural language search which focuses on understanding user search intent from natural language queries, specifically beyond keyword oriented techniques. A significant portion of this dissertation deals with finding semantics in information. Therefore, we discuss semantic networks, semantic web and linked data research. Finally, we mention several multimedia applications that are gaining popularity and can be better realized using social and semantic signals. Often, these signals do not originate in the multimedia domain, but in some cross domain. Therefore, we must detect and estimate these social and semantic signals, transfer them across domains, and improve recommendations in the multimedia domain.

2.1 Topic Modeling

Recently, topic modeling has gained a lot of popularity in analyzing semantic context in textual data [95]. Topic modeling originates from the impression that the construction of any sentence entails a mixture of topics [90]. Each word that the writer chooses to be part of the sentence is drawn from a mixture of topics in his head. Consequently each sentence, composed of these words, will also develop a membership towards some topics and not so much towards others. Thus, we can consider the mixture of topics to be the cause behind the generation of the entire document. Each document is a distribution over topics [89]. Topic modeling aims to uncover this inherent distribution of topics that guide the creation of the document.

A topic is an abstract concept. It is a collection of words, which when grouped together make some semantic sense. Another word for 'showing semantic sense' is to exhibit 'semantic coherence'. There are several popular methods to uncover the underlying topic distribution given a set of documents, such as Latent Dirichlet Allocation [101], Probabilistic Latent Semantic Analysis [89], Hierarchical topic models[88], Latent Semantic Analysis [90] etc. As mentioned earlier, the goal of topic modeling is to generate several clusters of words. Each cluster represents a particular topic. The result of topic modeling is to generate two distributions, namely the topic word distribution P(w|z) and the document-topic distribution p(z|d). Before we dive into LDA, we will first briefly discuss its predecessors, PLSA and LSA and the general vector space model.

The vector space model is a technique of representing documents as vectors of terms, usually the words in the documents. Each dimension corresponds to a separate term and when a term occurs in a document, its value in the vector is non-zero. Then, the cosine of the angle between two vectors indicates the similarity between documents represented by the corresponding vectors.

LSA falls into the category of vectorial semantics, where the features in a natural language sentence is represented by its words. Again, each document can be represented by a vector of words. The goal of LSA is to detect words that are semantically close. Given a collection of documents, each document can be represented as a column and the each word can represent the row. This generates a term-document matrix containing where each cell contains the number of times the word occurred in the document. Following this, a mathematical technique called Singular Value Decomposition is used to reduce the number of columns while preserving the similarity structure among rows [102]. Then, the cosine similarity between two rows represents how similar the two words are. Values close to 1 indicate very similar words while values close to 0 indicate dissimilar words.

The disadvantage of LSA is its inability to detect polysemy. It also assumes that words and document hold a joint Gaussian probability model, however research has shown this distribution is often Poisson [91]. The alternative to this is using a multinomial model, which is the basis of PLSA.

Probabilistic Latent Semantic Analysis is also statistical technique to understand co-occurrence of words in textual data. Unlike LSA which reduces a term-document matrix using linear algebra, PLSA uses a mixture decomposition from latent class models. As with LSA, let us assume there is a co-occurrence *(w, d)* of words and documents. PLSA models the probability that each co-occurrence was a mixture of conditionally independent multinomial distributions, i.e.

$$P(w,d) = \sum_{c} P(c)P(d|c)P(w|c) = P(d)\sum_{c} P(c|d)P(w|c)$$

where the latent class is *c*. More popularly, PLSA and LDA is often represented by the plate notation, shown in Fig. 6, where *M* is a set of documents, *d* is the document index, *c* is the word's topic drawn from the document's topic distribution *P(c|d)* and *w* is the word drawn from the word -topic distribution *P(c|z).* The shaded circles (*w* and *d*) are observable whereas the unshaded topic (*c*) is the latent variable. Of course, the number of parameters to learn equals *cd* + *wc*. These parameters can be learned using the Expectation Maximization algorithm [92].

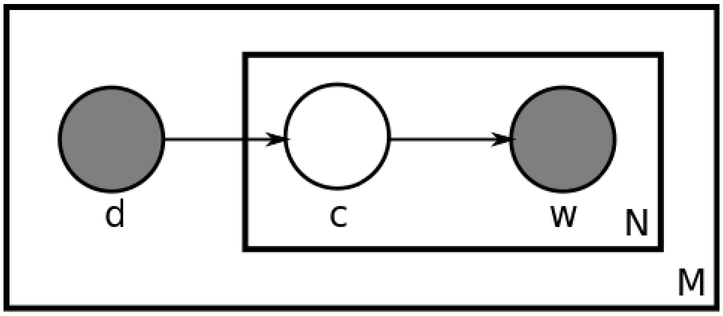


Figure 6: The plate representation of Latent Dirichlet Allocation.

In essence, LDA is very close to PLSA in terms of how terms and documents are treated. The major difference is that LDA is completely generative model overlapping a Hierarchical Bayesian model. In other words, PLSA does not maintain a prior probability on the parameters to be learned. But LDA assumes this parameters are itself variables and thus can be treated as hyper parameters with prior probabilities. *This prior is drawn from a Dirichlet distribution*, owing LDA its name.

LDA introduces two prior probabilities alpha and beta which affect how the per-document topic distribution and the per-document word distribution respectively behaves. As shown in Fig. 6, the outer plate represents a set of documents *M* while the inner plate represents the inner represents the topics and words within one document *N*.

The limitations of current topic modeling algorithms include the scalability with streaming or bursty set of documents, interpretability of the topics themselves and ..

Social Stream Topic Mining

Social data from Twitter streams can be mined to build a relevant topic space using topic modeling [17, 21]. Such topic space can act as a bridge between the social and the traditional media domain, supporting multimedia applications like social video recommendation and social video popularity. Topic modeling aims to extract topics from large corpus of unlabeled document by using generative models like Latent Dirichlet Allocation (LDA) [12]. There have been previous efforts to incorporate social data for recommendation [18, 23], but they do not use social streams specifically [21]. Social streams are more challenging to extract topics from; due to their dynamic, noisy, short and real-time nature [17]. Thus, large scale matrix decomposition is infeasible for social streams [18].

Previous research on mining social stream data assumes that the entire tweet stream is available to the algorithm at the beginning of the run. This assumption is only applicable in ideal case; it does not hold in real life situations. In our paper, we simulate the tweet stream in pseudo real-time, where the SocialTransfer algorithm has not seen the entire tweet stream in advance. Instead, the complete timeline is divided into time slots, and a certain number of tweets occupy each time slot as they are generated in real life, similar to the technique in [1]. Tweet chunks are fed to the SocialTransfer algorithm in time-sequential batches based on the time slots in which they are generated (pseudo real-time). We show later how Social- Transfer is a unique method to combine scalable social stream topic modeling and transfer learning; providing a natural interface for topic modeling to fit into the process of transfer learning and seamlessly integrate topic model and transfer learning.

2.2 Transfer Learning

Common machine learning techniques traditionally address isolated tasks. In contrast, transfer learning aims to transfer knowledge learned in one source domain and use it to improve learning in a related target domain. Fig. 7 shows the basic concept of transfer learning. The source domain data Zsrc contains the auxiliary data, while target domain Ztar contains the training and test data. A comprehensive survey of transfer learning techniques is provided in [114]. A unified framework for transfer learning in scenarios ranging from cross-domain, cross-category and self-taught learning is described in [8]. Transfer learning has been previously used in various cases including classification, image clustering, collaborative filtering, and sensor based location prediction [8, 20, 27].

Domain-independent feature representation in transfer learning can also have significant effects on performance (e.g., to avoid negative transfer) [8]. Spectral techniques have been used to address the problem of combined feature representation [11]. However, such spectral techniques (e.g., eigenvector extraction [18]) should scale to dynamic social stream traffic, which is addressed in this paper. Although [22] attempts to use transfer learning for social recommendation, their model is not real time and limited to non-streaming data only. Instead, we show how to model transfer learning from streaming social data in real time, which is a significantly challenging problem not yet resolved.

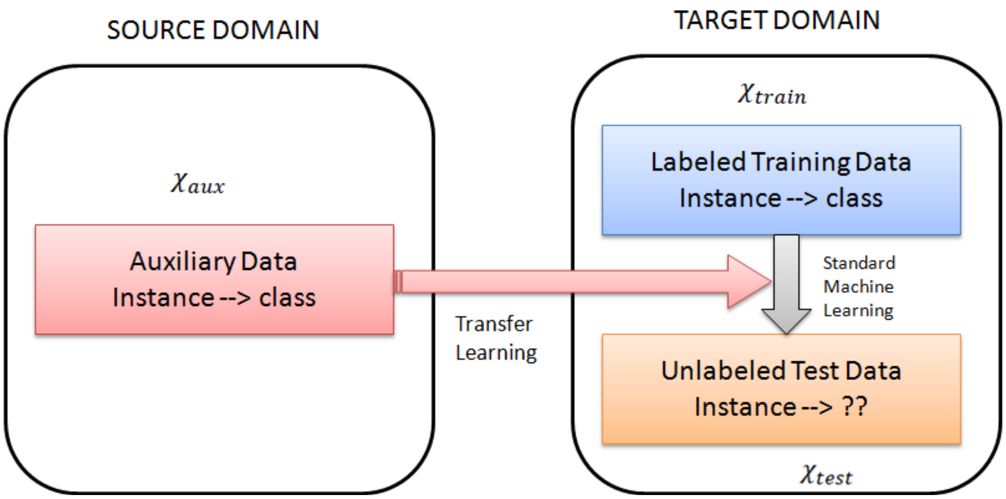


Figure 7: Transfer Learning compared to standard machine learning.

Our *SocialTransfer* framework is inspired by the work in [8]. However, we distinguish ourselves from [8] in scaling transfer learning to specifically incorporate social stream data as source domain and show how topic learning can be smoothly combined with transfer learning in real-time. To *the best of our knowledge, a framework that can handle social stream topics distinctively as source domain for cross-domain transfer learning has not been proposed before.* This is challenging due to the unique characteristics of social stream data [16].

2.3 Natural Language Search

The collaboration between researchers in information retrieval and linguists is helping us to transcend into an era of natural language search, where search engines can comprehend user intent or meaning from queries written in natural language (NL). This essentially requires algorithms that can not only retrieve results based on keywords, but more importantly, understand semantics, discourse and pragmatics in a NL sentence.

Understanding semantics of the query involves finding meaningful relations among its words, which can be represented as a network of words, called the semantic subnet of the query [46]. It has been found that semantic subnets enable identification of event structures within sentences [51] and assist higher-level NLP tasks, like Question Answering (QA) [52].

Although huge progress has been made in the field of computational linguistics, there still appears to be enough diversity in NL constructions that hinder sufficient information extraction purely from the NL query for improved search results. An alternative, suggested by numerous researchers, is to search the document space with more words than those contained in the original query [50]. This technique, called Query Expansion , relies on finding words that are semantically similar to query words but not in the query (called expanded words). There are two popular ways to find such expanded words. One way is to analyze the user search logs and discover which words occurred in the same query [63]. For example, the chances of 'flu' and 'medicine' occurring in the same query will be much larger than 'flu' and 'guitar', allowing the algorithm to realize that 'flu' and 'medicine' has stronger semantic similarity. However, this technique suffers from the problem of the long tail [93]. The second way is to use an ontology or a semantic network, where expanded words can be detected as multi-hop neighbors of the query word represented as a vertex. In other words, query words can represent concept nodes in the semantic network, and the expansion of the query can be realized using the linkage of the network. Unfortunately, a breath first search from each vertex node of the graph database that matches a query word is not computationally feasible.

A common way to extract word connectedness from NL sentences is using parse trees, which depend on lexical structure of the NL sentence [51]. Further, functional keywords in NL can be detected using methods like Named Entity Recognition (NER) or

Semantic Role Labeling (SRL) [53]. Both these techniques provide a higher level of abstraction than the basic syntax/parse tree.

Due to immense diversity in human query constructions, the lexical patterns too have a great variety. This noise in sentence structure will often mislead algorithms. Imprecision of NL usage is a major obstacle to computation with NL. Therefore, it is necessary to develop a technique that partially relaxes the rigid grammar of the language. While imprecise or varied grammatical constructions are difficult to capture using POS or predicate logic, note that the human cognition can often eliminate such noise to interpret meaning. At first this sounds like a baffling fact; but everyday experiences reveal that human cognition is significantly more robust in extracting meaning from poorly constructed sentences compared to state-of-the-art techniques for NL understanding [47].

Several problems like word-sense disambiguation, specificity of grammar and keyword (not semantic) based approaches inhibit portability of several existing NLP techniques across systems and domains [49, 63]. The closest work to our research is [46], which uses a POS-based approach in extracting subnets from queries. The accuracy of query subnet extraction compared to a human standard can be evaluated using metrics such as Consistency Index [68]. The results stated in [46] are tested on a very limited number of queries (approx. 12), which does not come close to capturing the diversity in human query constructions or web scale. In contrast, we provide empirical results on 5000 queries from three query datasets with different noise levels.

2.4 Semantic Networks

A semantic network is a directed or undirected graph where nodes are concepts and edges represent semantic relations between two concepts. Such graphs are the widely used for knowledge representation. Popular semantic networks (and semantic databases) include WordNet [42], DBpedia [37], Freebase [94] etc. Freebase is one of the key components of Google Knowledge Graph [43].

Building large semantic networks starts with the construction of a Simple Knowledge Organization System (SKOS), which was recommended by the World Wide Web Consortium to be part of the Semantic Web [37]. It is a family of formal languages used to represent thesauri, classification schemes, taxonomies, subject-heading systems, or any other type of structured controlled vocabulary. The SKOS represents the core of the Semantic Web.

As explained later (Chapter 5), when we query an SKOS-based semantic network with a concept or entity, it returns the possible categories to which the entity belongs to. Around the SKOS, various sets of concepts can be added (usually using other datasets), which can tackle more complicated tasks like semantic parsing, semantic role labeling and word sense disambiguation. The core graph is often quite sparse, especially when viewed with the force-atlas spread visualization as shown in Fig. 8.

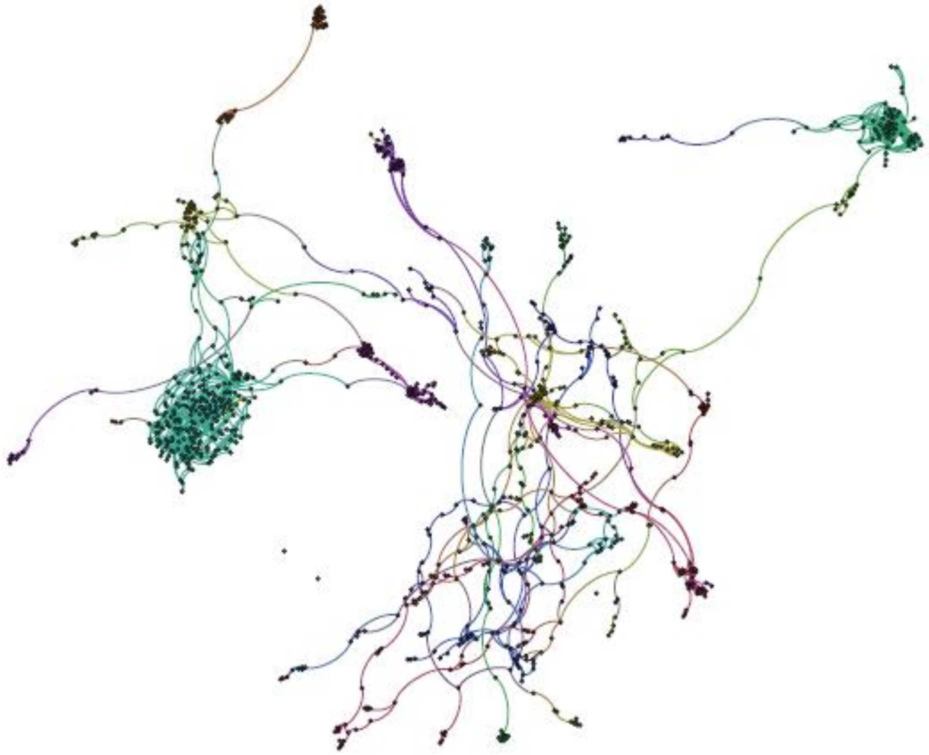


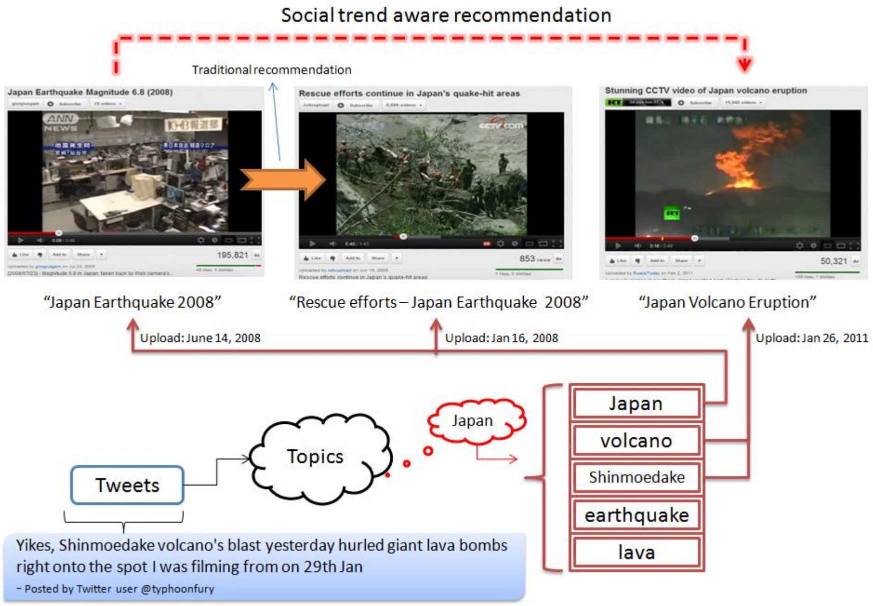
Figure 8: Part of core SKOS network of DBpedia.

It is important to note that the term Semantic Web is often confused with 'Semantic Networks'. The former is a standards movement, which involves designing web pages in formats that can be easily machine-readable. It uses Resource Description Formats (RDF) as units to describe data. Semantic networks on the other hand, are generic graph that describe concept relations. The latter can be engineered by using data from the semantic web.

Network science is the study of relational data in physical, biological and social systems leading to predictive modeling of related phenomena. In general, there are several metrics that can indicate the importance of a node in the network, its relation with other nodes and the properties of the network as a whole. Popular network attributes include its average degree, clustering coefficient and centrality-based measures [2]. When the network nodes can be naturally grouped into overlapping cluster of nodes such that nodes within a cluster are densely connected, it is said to exhibit community structure [6]. The greater the number of communities in the network, the more is its modularity [2].

2.5 Multimedia Applications

Multimedia is media that consists of many content formats, such as text, images, videos, micro-texts, interactive visualizations etc. Most multimedia applications are either linear or non-linear. Linear multimedia does not allow user interaction, meaning it is usually un-altered. Images, videos or audios are example of such multimedia. On the other hand, video games, social micro-texts etc. allow for users changing the content as they interact, meaning they fall into the non-linear category.


Social trend aware recommendation
Traditional recommendation
Japan Earthquake Magnitude 6.8 (2008)
195,821
Rescue efforts continue in Japan's quake-hit areas
853
Stunning CCTV video of Japan volcano eruption
50,321
"Japan Earthquake 2008"
"Rescue efforts – Japan Earthquake 2008"
"Japan Volcano Eruption"
Upload: June 14, 2008
Upload: Jan 16, 2008
Upload: Jan 26, 2011
Tweets
Topics
Japan
Japan
volcano
Shinmoedake
earthquake
lava

Yikes, Shinmoedake volcano's blast yesterday hurled giant lava bombs right onto the spot I was filming from on 29th Jan
– Posted by Twitter user @typhoonfury

Figure 9: Transforming traditional multimedia application to social-aware.

In today's digital world and online communities, the usage of multimedia applications are ubiquitous. From image viewers on Facebook (social network) to filters on Instagram (social photos), from screen-casting on Twitch (video games) to 3D modeling in Maya (motion picture 3D), from music players like Spotify to video publishing sites like YouTube and from e-book applications on Amazon's Kindle to gif videos like Vine - multimedia applications govern human interaction with machines.

Several of these applications consists of challenging artificial intelligence problems. For example, in video publishing, the site needs to recommend relevant video to user based on what he/she is currently watching - a classic example of recommendation systems [26]. In image search, the web site must parse a natural language text, extract semantics and retrieve related images. Moreover, query suggestion helps users to restructure their query based on available media content on the website and what other users' have searched for [22]. Brands use social media to quantify audience engagement, which requires intelligent analysis of user-generated media content to detect user profiles. These are all scenarios we shall discuss in this thesis, and describe how cross-domain data can help in improving individual multimedia applications.

CHAPTER 3: LEARNING FROM SOCIAL DATA

As mentioned previously, two ideas within the last decade have fundamentally disrupted how humans obtain information. The first involved engineering powerful search techniques which could quickly retrieve relevant web documents. The second is to allow sharing of user-generated content by means of social media and social networks. Web search aims to retrieve relevant documents across many domains on the Internet. Thus, there is media data in different domains that can be searched. If one of those domains is social, it also enables you to access real-time data about users and connect it to relevant media.

Realizing that there exists cross-correlation between media data in different domains often generated in response to the same events in the physical world, we aim to build a common topic space between two domains to enable cross-domain learning and recommendation (Fig. 13). As a proof of concept, in this research we show it is possible to sustain such a topic space between the domains of social stream and online video. In particular, we take the social stream of Twitter and the videos collected from a commercial video search engine as examples in this work. The principal reason behind building a topic space is to construct a base context platform upon which multiple media applications can be forged [4]. It acts like a *bidirectional* bridge between tweets and videos.

We tweak the online LDA model [103] to learn topics in real time from social stream and adapt it to scale with the bursty nature of social streams. The proposed topic model, which we call Online Streaming LDA (OSLDA) is utilized to extract, learn, populate, update and curate the topic space in real time, scaling with streaming tweets

[4]. The learned topics can then be used as a supervised bias when transferring information from the social domain to the video domain, which will be discussed in the next chapter.

3.1 Social Stream Mining

It is challenging to extract and mine relevant and valuable information from social streams (e.g., Twitter) and correlate social media across different domains. This is because of the noisy nature of social streams. For example, each tweet in Twitter is limited to 140 characters. This severely hinders techniques based on "bag-of-words." [95] Second, tweets are usually noisy and improperly structured in grammar/syntax, leading to the difficulty to process via standard Natural Language Processing (NLP) tools. Third, the input data typically arrives in high-volume streams (bursty traffic), and thus, algorithms mining them must scale in learning.

We use Online Learning LDA (explained in the next page) to extract topics ($z \in Z$) from a stream of tweets ($d \in D$) [101]. LDA generates two distributions: a topical word–topic distribution $P(w\,|z)$ and topics-tweets distribution $P(z|d)$. The vocabulary consists of words $w \in W$. Parameters α and β are Dirichlet priors to the topic-tweet and the word-topic distributions respectively. A tweet is a sequence of words, where w_n is the n^th^ word in the sequence.

Consider a k-dimensional Dirichlet random variable θ that can takes values in (k-1) simplex. LDA assumes the following generative process for each tweet d in the corpus D: (i) Choose N from a Poisson distribution. (ii) Choose $\theta \sim Dirichlet(\alpha)$. (iii) For each of the N words w_n: (a) Choose a topic $z_n \sim Multinomial(\theta)$ (b) Choose a word w_n from $p(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic z_n. The dimensionality k

of the Dirichlet distribution is assumed known and fixed. Please refer to [101, 103] for further details.

Therefore, the joint distribution of the topic mixture θ, the set of *N* topics Z and a set of *N* words in the vocabulary W is given by:

$$p(\theta, Z, W|\alpha, \beta) = p(\theta|\alpha) \prod_{n=1}^{N} p(z_n|\theta) p(w_n|z_n, \beta) \qquad (1)$$

LDA represents every tweet as a random mixture over latent topics whereas every topic has a distribution over the words. A topic is comprised of a set of *topical words*. For example, one topic generated by LDA is: {*egypt, mubarak, tahrir, army, revolution, ...*} , which clearly is related to the concept of the Egyptian revolution in Feb, 2011.

3.2 Online Streaming LDA

Our system learns in real time by updating the topic space with every incoming stream of tweets in a time slot (Fig. 10). We call it Online Stream LDA (OSLDA), since it leverages online LDA [103] and also scales across streams of incoming tweets, updating tweet-topic and topic-video connections at the same time. Unlike [103], which updates the word-topic prior distribution β with time, our method updates the topic space with time, using an active time decay function. Thus, OSLDA assumes the word-topic distribution can change significantly due to the dynamic nature of tweets [4]. This makes our model robust to streaming nature.

With each time slot, OSLDA models incoming bursts of tweets (Fig. 10)) and updates the topic space. Empirical studies showed that fixing number of topics to 30 was enough for 60K tweets per time slot. Intuitively, processing more tweets should take more time, but the number of topics needed to be extracted from a sudden burst (say 120K tweets) is usually less, since the burst is typically caused by a single event (single topic). So, the number of topics to be extracted does not double if the tweet burst doubles.

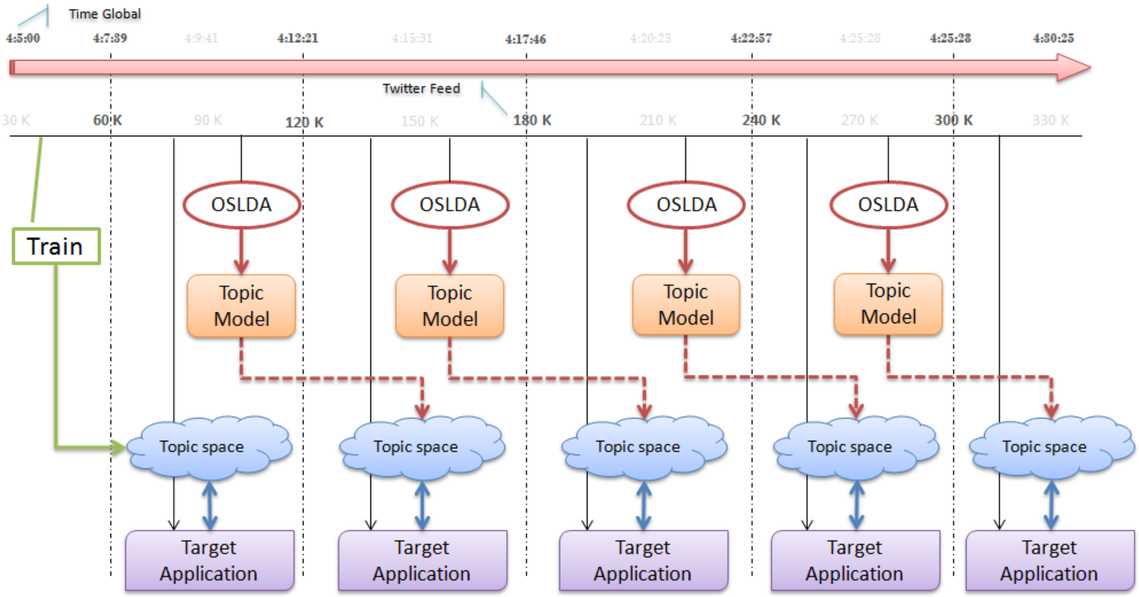


Figure 10: OSLDA updating topics space with incoming stream over time.

A principal difference of OSLDA from previous topic modeling algorithms is that OSLDA is capable of scaling with bursts of tweets. It is important to remember that the stream size per chunk/time slot is not constant, and therefore any social stream topic mining learner must deal with different document sizes at different times. In traditional LDA, the number of topics (a prefixed parameter) to be extracted depends on the diversity and the number of documents [115]. If the number of tweets in the stream doubles, it would appear that the stream would certainly become more diverse. Thus normally, more topics should be extracted, which would take more time computationally.

Interestingly however, the reverse phenomenon is observed for social stream data. A burst of tweets usually indicates 1 or 2 big events, which causes the stream diversity to drastically reduce. Thus, the necessary adaptation to bursts of tweets is not to increase the number of topics to be extracted, but reduce it. This single observation allows us to extract reasonable topics even when the stream size doubles. It allows OSLDA to scale with variable and bursty nature of social streams.

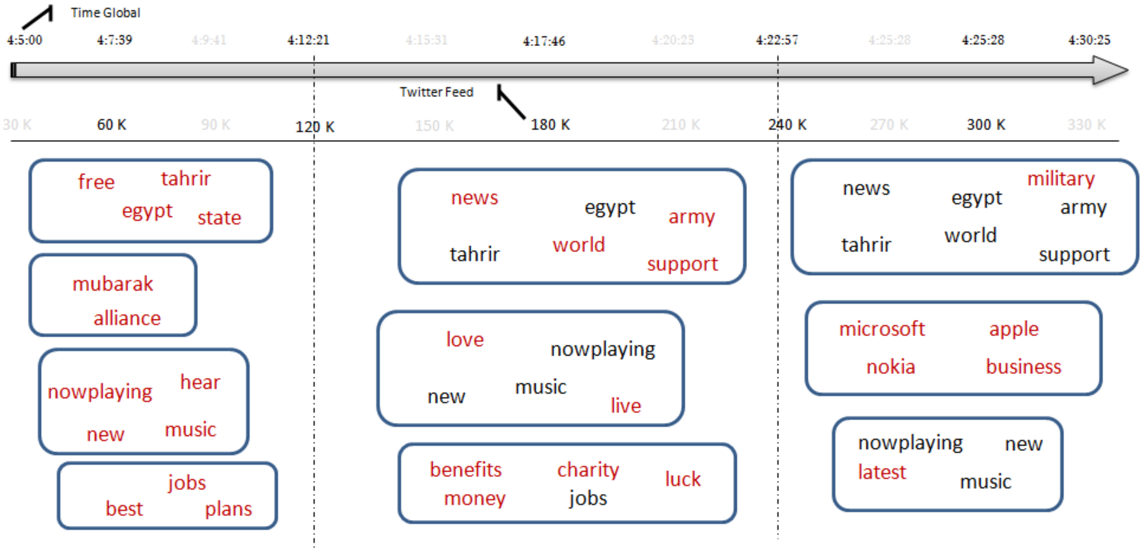

Time Global
4:5:00
4:7:39
4:9:41
4:12:21
4:15:31
4:17:46
4:20:23
4:22:57
4:25:28
4:25:28
4:30:25
Twitter Feed
30 K
60 K
90 K
120 K
150 K
180 K
210 K
240 K
270 K
300 K
330 K
free tahrir egypt state
mubarak alliance
nowplaying hear new music
jobs best plans
news egypt army tahrir world support
love nowplaying new music live
benefits charity luck money jobs
news egypt military army tahrir world support
microsoft apple nokia business
nowplaying new latest music

Figure 11: Topics (trending) detected by OSLDA from Twitter stream over time.

An example of topics extracted by OSLDA on real-world data (half an hour of Twitter stream) over time is shown in Fig. 11. Each chunk of tweets is shown by the dotted vertical line, during which OSLDA runs once. Every block resembles a topic, consisting of topical words. Newly detected words in a topic are colored red.

Are the extracted topics relevant ?

Fig. 12 shows the distribution of search queries with time in video query logs for the topic '*Egypt'* with real-time trend variation on Twitter as detected by OSLDA. We clearly notice that there is few minutes time lag between a trend topic appearing on Twitter, and the same topical words being searched on the commercial video search engine. This means as trends rise and fall in Twitter, the volume of queries on the same topic rises and falls for video search. patterns for web and image search on Feb 11, 2011.

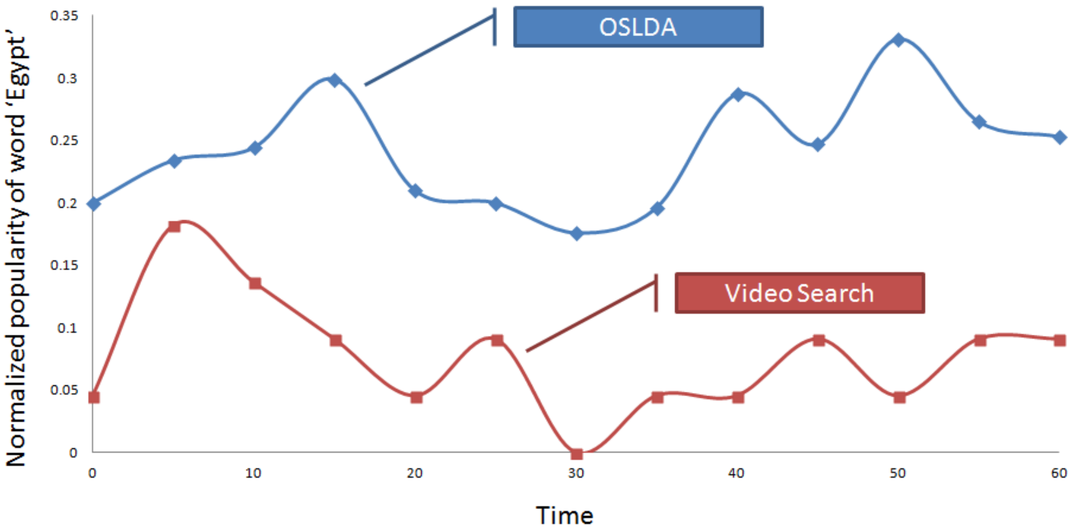


Figure 12: Trending score of topical word 'Egypt' (detected by OSLDA) compared to real-world video search trending keywords. It illustrates the periodic lag that video search sustains when compared to OSLDA topic detection.

Topical Words	**Assigned Topic**	**YouTube Category**
dance, adventure, photography, visit	events	Travel & Events
anime, hero, online, celebrity, diva	films	Films & Animation
iphone, games, showcase	electronics	Sci. & Tech
war, economy, army, revolution, blog, egypt	politics	News & Politics
trailer, show, live, watch	entertainment	Entertainment
wow, rap, jam, gaga	music	Music

Table 1: Topical Words detected by OSLDA belonging to certain topics. Column 3 represents relevant YouTube categories for these topics.

It was not surprising that '*Egypt*' was the hottest search topic that day. In fact, Google Web Insights (www.google.com/ insights/search/) provided us with the top 10 web search keywords related to '*Egypt*'; seven of which had already been detected by OSLDA earlier. For Google Image search, 6 of the top 10 search keywords were detected by OSLDA.

3.3 Topic Space

Remember our main focus is to transfer the information among domains. Thus, it is necessary to store the learned topics somewhere and update it with time as new topics come in. This abstract space is called the topic space. The topic space is a matrix, where each row represents one topic and each column represents a feature word. The entry in a cell represents the probability that the word belongs to the topic, as given by OSLDA $P(word|topic)$. We maintain a list of 75 top topics at a certain time in the system, which means the number of rows in the matrix is 75. The feature word size varies depending on the type of topics, but on average it can be as large as 38,000. An easy way to detect the 75 top topics is by adding each row and consequently sorting.

As shown in Fig. 13, this topic space servers as the *bidirectional connection* between tweet and video domain. Once such a bidirectional connection is established, information can flow in either direction, consequently supporting applications such as social video recommendation or tweet enrichment by video. In the next sub-section, we describe how this is achieved mathematically.

Topic Space as the bridge

Using the topic space, we can connect a set of videos for any tweet. On the *vidSide,* we have a set of videos (*V*) with related video identifiers. Our goal is to find the membership strength each video possesses with the set of topics in the topic space.

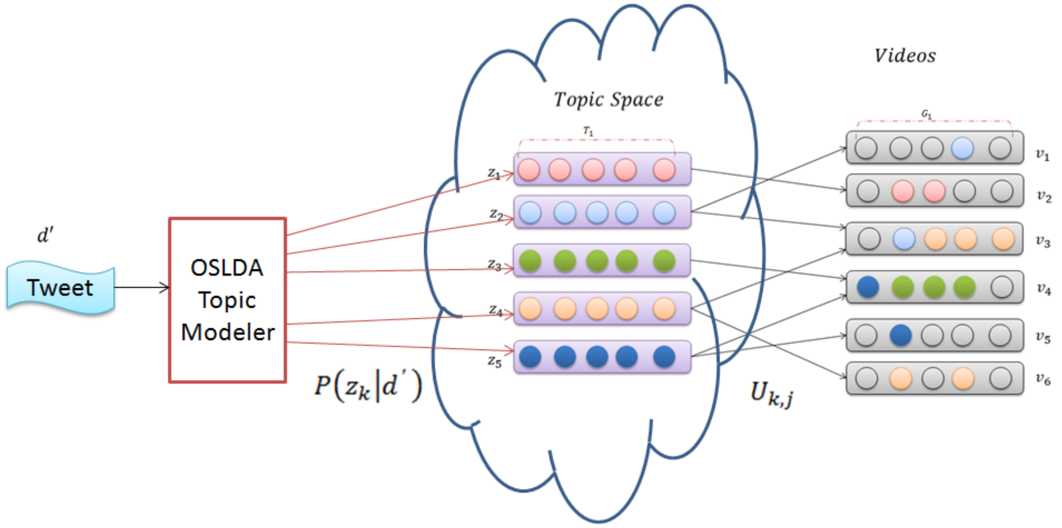


Figure 13: The topic space, as a bridge between cross media domains.

Please note that a video tag is a video identifier. For the j^{th} video, the set of tags is represented by G_j. We also have a set of topical words (which were already extracted from tweets). Let the topical words in the k^{th} topic be represented by the set T_k. Then, treating the set of topics and videos as a bipartite graph, we can define a link weighting function *U* such that:

$$U_{k,j} = \frac{T_k \cap G_j}{T_k}, \quad 0 \leq k < |Z|, 0 \leq j < |V| \qquad (2)$$

Thus, the more the common tags a video has with the words of a topic, the higher the weight $U_{k,j}$; and thus the higher the membership of the video towards this topic.

Tweets are often noisy and difficult to understand for users. We can improve user experience of tweets by recommending related and relevant media. From the user perspective, this should enrich the information surrounding the tweet (in terms of the topic of the tweet), since media (image/ video) is probably easier to comprehend for the user. Once the LDA topic modeler is trained on a stream of tweets, we can use it to connect any tweet to a topic, and eventually the selected topic to a set of videos as described in the previous subsection. The idea is clarified using Fig. 13.

Given a tweet *d'*, we can find the probability distribution of topics for that tweet using the LDA topic modeler. Subsequently, videos to be recommended are selected based on the optimization:

$$v^* = \arg max_{0\le j<|V|} \sum_{0\le k<|Z|} P(z_k|d').\, U_{k,j} \qquad (3)$$

Thus, the tweet connects to those videos for which it has the strongest links through the topic space. Think of $P(z_k|d')$ signifying the tweet-topic link weight and $U_{k,j}$ representing the topic-video link weight.

Runtimes of OSLDA

Our Twitter dataset consists of 3.6 million tweets generated on February 11th, 2011. We fixed each time slot period to five min. We noticed that approximately 60K tweets were generated every five min. The first 50K tweets were used as training data. The rest of the 3.1 million tweets were used for test. Tests were run on a system having AMD Opteron 2.09 GHz and 64 GB RAM. Performance of OSLDA is summarized in Table 2.

# Topics Extracted / # Tweets	5 topics	10 topics	30 topics	60 topics
30 K Tweets	0.66	0.72	1.09	1.67
60 K Tweets	1.41	1.83	2.11	3.31
90 K Tweets	1.75	1.93	3.18	5.19
120 K Tweets	2.53	3.1	4.42	6.81

Table 2: Time taken (in minutes) to extract a certain number of topics from a tweet stream of size from 30,000 to 120,000 .

In summary, we empower tweets with related videos from cross domain. On a related theme, we should note that Twitter social trends are also a distribution over topics in the topic space. We know that such trends are a measure of real-time social popularity. Thus, if we leverage this observation, we could augment video popularity based on socially trending topics. This is the theme of the next chapter.

CHAPTER 4: TRANSFERRING INFORMATION FROM SOCIAL TO VIDEO DOMAIN

The task of information transfer asks two fundamental questions (1) what information is transferable (2) how to transfer this information in real-time. In the scenario of social stream data, both these questions are considerably complex to answer. The first question is difficult to solve since streams are noisy, consisting of several non-natural language user-generated textual data. Moreover, tweets are generated at a very fast rate, thus the speed of information transfer or update is not trivial either.

4.1 Social Signal Penetration Hypothesis

The social signal penetration hypothesis states that a social trend (which is associated to Twitter) behaves as a spatio-temporal signal which penetrates into other domains (like YouTube), i.e. data in YouTube is affected by the trend in Twitter after some time delay. We claim that the topic space allows for the signal to be carried over to the other domain [105]. In this part, we explain the engine that lets this penetration possible. Remember our constrains in designing the engine: (1) Real-time. (2) Progressively updating the recommendations as the topic space changes.

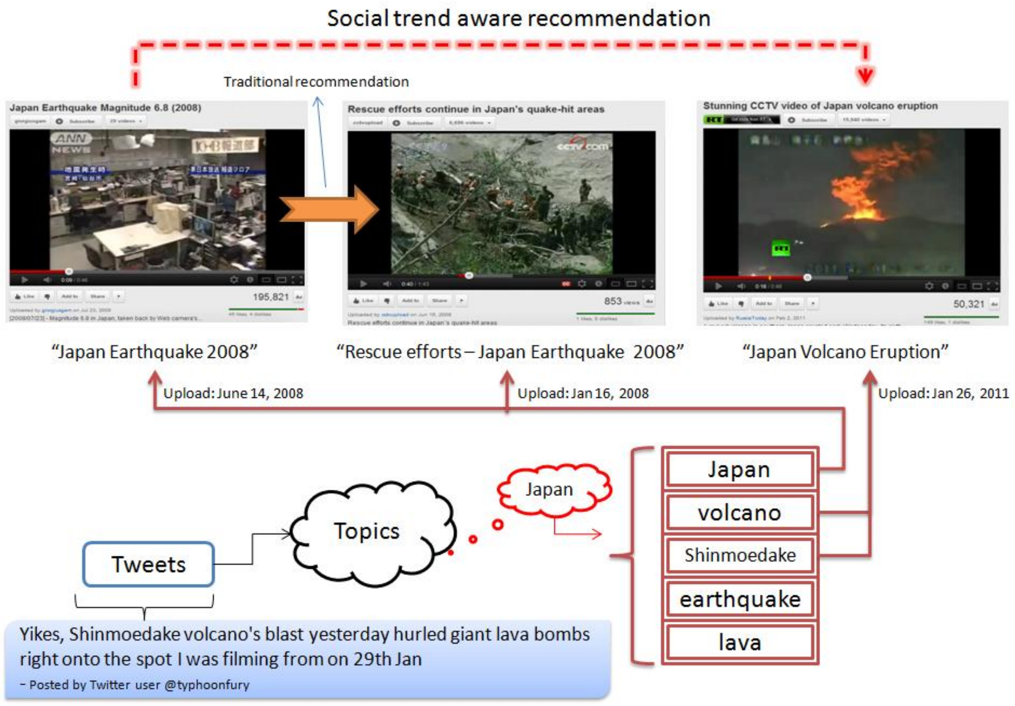
Problem Definition

We have two datasets in the target domain; the target training data $\chi_{train} = \{x_{tr}^{m}\}_{m=1}^{M}$ with labels and the target test data $\chi_{test} = \{x_{ts}^{n}\}_{n=1}^{N}$ without labels. The training data contains *M* instances whereas the test data contains *N* instances. Unlike traditional machine learning, we also have an auxiliary data set $\chi_{aux} = \{x_{ax}^{k}\}_{k=1}^{D}$, consisting of D tweets instances. We assume that the target data and the auxiliary data share the same categories (e.g., both a tweet and a video can be regarding music), but exist in different domains (e.g., tweet is social text-based micro-blogging while RVGs consist of videos).

Consider a set of B videos in the target domain. For a video $v_i, 1 \leq i \leq B$, we can represent the set of tags of v_i as $\{tags(v_i)\}$. Each tag in the set $\{tags(v_i)\}$ is a word, represented as $w_j^i, 1 \leq j \leq |tags(v_i)|$. Now consider a stream of D tweets picked from the source domain to be used for modeling the social topic space. For a tweet $t_k, 1 \leq k \leq D$, let $tpw(t_k)$ represent the topical words in the topic of t_k (we consider only the principal topic, i.e. topic for which the conditional probability of topic given tweet is maximum). Then each instance/label of the twitter stream data can be represented as $t_k \rightarrow tpw(t_k)$. These instances can be combined into the auxiliary data set $\chi_{aux} = \{x_{ax}^{k}\}_{k=1}^{D}$.

All the instances $x \in \chi_{train} \cup \chi_{test} \cup \chi_{aux}$ are represented by the features in the feature space $\mathcal{F} = \{f_s^{(i)}\}_{i=1}^{s}$. Our goal is to learn an accurate classifier $f'(.)$ from $\chi_{train}$and χ_{aux} that can predict the testing data with minimum classification error. We call this classifier $f'(\chi_{test})$. Thus, the goal of transfer learning is to minimize the prediction error on χ_{test} by leveraging the auxiliary data from χ_{aux}.

In the next section, we present *SocialTransfer* – a scalable technique for real-time transfer learning between the domains of social streams and traditional media (like video). *SocialTransfer* utilizes topics extracted from social streams to build an intermediate topic space in between the social and video domains. The topic space is an abstract space containing several clusters of words belonging to various topics that reflect world events in real time, including current and past trends. We use the Online Stream LDA model (OSLDA) to learn topics from social streams [4]. *SocialTransfer* uses a graph based framework to model the transfer learning problem (what feature information is transferable and how) between the social and the video domains. Spectral analysis of this graph fetches the eigenvectors, using which we can represent both the social and the video feature information as a combined feature representation [24]. Since the stream is temporal nature, *SocialTransfer* also allows progressively updating the topic space and seamlessly incorporating newer trends into the transfer learning framework for socially aware media recommendations.


Social trend aware recommendation
Traditional recommendation
Japan Earthquake Magnitude 6.8 (2008)
195,821
Rescue efforts continue in Japan's quake-hit areas
853
Stunning CCTV video of Japan volcano eruption
50,321
"Japan Earthquake 2008"
"Rescue efforts – Japan Earthquake 2008"
"Japan Volcano Eruption"
Upload: June 14, 2008
Upload: Jan 16, 2008
Upload: Jan 26, 2011
Japan
volcano
Shinmoedake
earthquake
lava
Japan
Topics
Tweets

Yikes, Shinmoedake volcano's blast yesterday hurled giant lava bombs right onto the spot I was filming from on 29th Jan

– Posted by Twitter user @typhoonfury

Figure 14: Example of using social topics in building social trend aware multimedia applications. In this example, we show that related video (i.e. video-video) recommendation can be enriched by using topics learned from the domain of social streams. This cross-domain transfer of knowledge is accomplished through a mutual topic space (e.g., the space includes the topics like "Japan" containing words like "volcano," "earthquake," and so on).

Fig. 14 shows an example of this kind, for social video recommendations. The framework we develop can be reused for several multimedia applications where social influence is capable of improving performance. Our results show that *SocialTransfer* considerably outperforms traditional learners without transfer learning.

4.2 SocialTransfer

Our goal is to combine the training, test and auxiliary data into a single transfer framework for prediction. There are two problems we particularly need to solve in this framework: (1) we must learn the interconnected pattern of shared features between the source and the target data, and (2) since the topics modeled from social stream (auxiliary data) changes with the real world trends, we need a transfer framework that can allow *progressive inclusion of topics in pseudo real-time*.

Let us focus on the first problem and understand how to learn the interconnected structure of shared features across the domains. The single transfer framework we use for this purpose is represented as a graph called the transfer graph *G* (see Fig. 16), which contains the videos, tweets, feature words and category information. To learn the interconnected pattern of shared features between the source and the target data, we perform spectral analysis [24] of the transfer graph. As shown in Fig. 15, spectral learning uses a technique called Power Iteration [106] to extract the eigenvectors from the Laplacian representation of the transfer graph. Spectral analysis of the transfer graph gives us the combined feature representation of the auxiliary and the training data using eigenvectors. This eigen feature representation reflects the intrinsic structure in terms of the principal components of the combined source and training data. Traditional learners (like Support Vector Machines/SVM [27]) can then use the combined features for prediction rather than using only the training features.

Now, let us focus on the second problem of how to progressively include social topics. Since the tweet stream is incrementally witnessed by the algorithm, the transfer graph needs to be updated in order to progressively include the twitter topics in pseudo-real time. Said alternately, in order to include topics as they are generated in real time, we must update the transfer graph and recalculate the graph spectra. This is achieved by treating the topics as input supervision before spectral learning (as shown in Fig. 15). In particular, to incorporate the new topical information to the existing transfer graph, we utilize selected topics from the topic space created from the tweets. We can treat the topical words of tweets and the corresponding topics as labeled instances, and then incorporate the new tweet information as a semi-supervised rank update (a rank update refers to cases where a matrix is updated using outer product (as opposed to dot product) on the existing Laplacian matrix as shown in the flow diagram Fig. 15. In other words, selected topics act as input supervision for the Laplacian matrix which allows for smooth incorporation of social topics into the transfer learning framework.

We use the Online Streaming LDA (OSLDA) model for real-time topic learning from Twitter stream [4], described in Chapter 3. Each topic is comprised of a group of related words called *topical* words. Topic learning treats each tweet as a document and builds a generative model to connect the tweet to one or more topics. Thus, the topic of a tweet contains words (topical words) that are related to the tweet words but might not be explicitly present in the tweet itself. More precisely, the topic modeling generates two distributions, a tweet-topic distribution and a topic-word distribution.

As mentioned earlier, extracting topics from social streams is non-trivial, due to the unique characteristics of social stream data [80]. Previous work has however shown that significantly popular topics (e.g. trending topics) can be extracted from social streams with reasonable accuracy [107]. Since every topical word in the topic space has an assigned topic label as shown in Table 1, the entire topic space can be treated as some sort of social bias for any semi-supervised learning task that requires social influence. Again, *devising a natural way to incorporate this social bias into transfer learning is not trivial, which is one of the important issues addressed in this dissertation.* Note that each assigned topic consists of a cluster of topical words. Similarly, each topic can be considered a cluster in the topic space. We can limit ourselves to incorporating only selected topics from the topic space as input supervision (an additional set of labeled instances) for the transfer learning task.

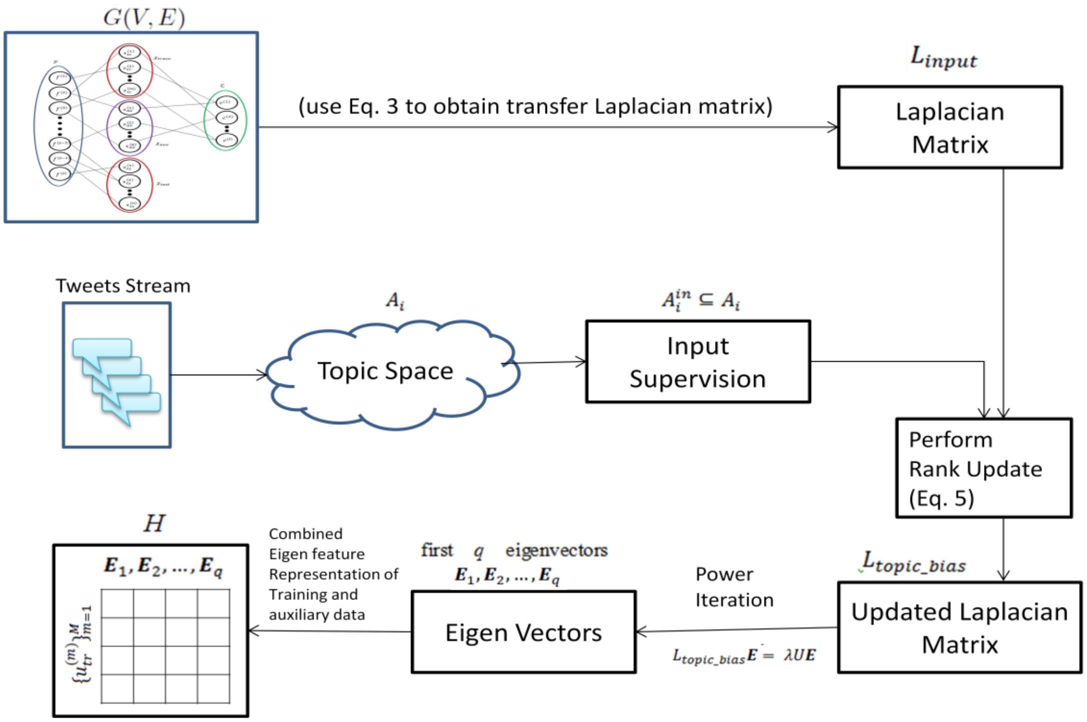


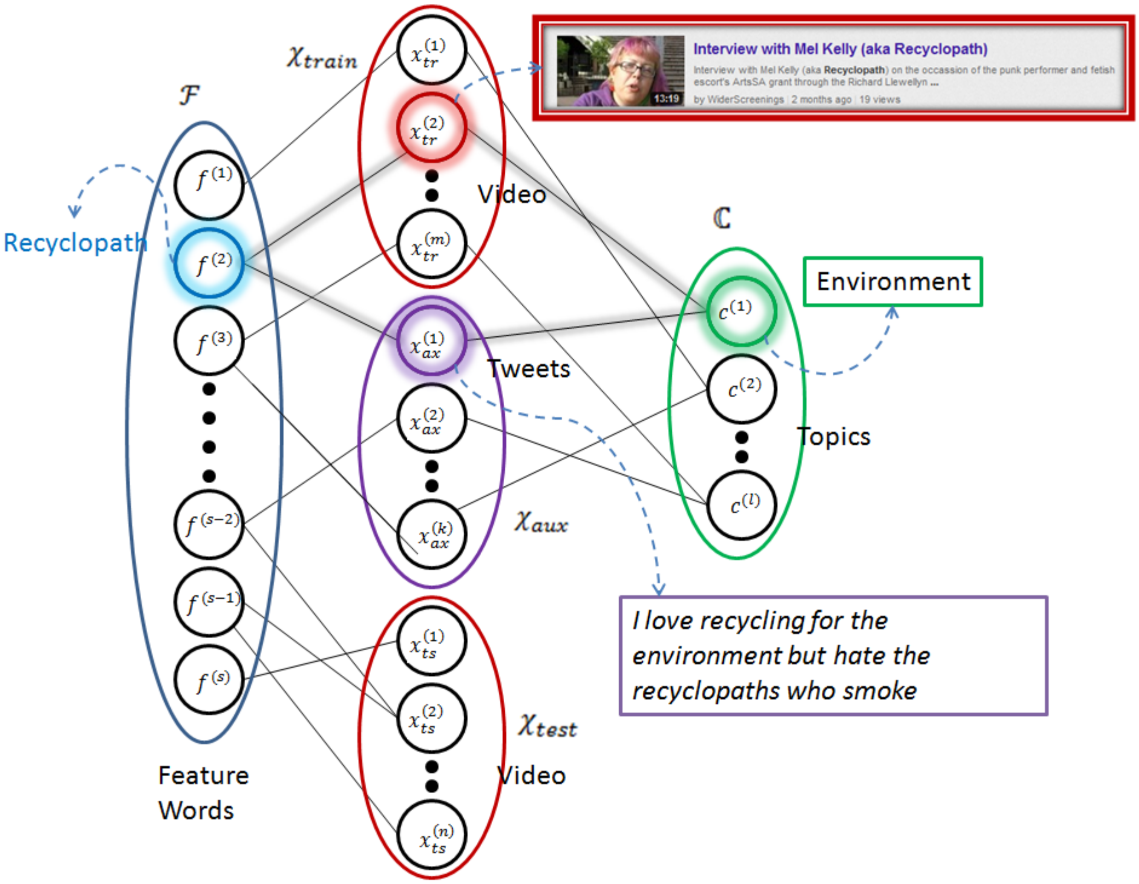
Figure 15: The flow diagram addresses the overall approach in solving the two key problems of SocialTransfer: (1) learning the shared feature representation across domains in terms of eigenvectors using Spectral Learning (Power Iteration), and (2) reflecting the progressive inclusion of topics by updating the transfer Laplacian matrix.

This choice will depend on factors such as whether we want to model only fresh (trending) topics or only video category specific topics. Thus for K topics in the global topic space, we can choose a particular set of topical words $A_i^{in} \subseteq A_i$, for i=1, 2,…, K to act as the bias or input supervision to update the transfer graph before spectral learning. This sort of input topic supervision is fed into the transfer graph progressively, as is depicted in Fig. 15, where topics modeled in real-time from the social stream using OSLDA is used to update the transfer graph by means of a ranked update (Eq. 7) on the transfer Laplacian matrix representation of the transfer graph. This allows progressive and seamless inclusion of topics into the transfer graph as shown in Fig. 4, facilitating the social influence in transfer learning.

Transfer Graph

A general graph based framework for cross-domain transfer learning was proposed in [23], which includes the target and the auxiliary data with some common relations and attributes between them. We adapt that framework in our scenario. However, the graph in [23] cannot update itself to incorporate streaming tweets topic information in scalable fashion. Instead, the transfer graph in *SocialTransfer* is capable of updating itself with new tweets stream topics in real-time. The transfer graph's main purpose is to capture the cross-domain attributes of social streams and videos for using in the transfer learning task and model the relation between the auxiliary data from Twitter and the target video data. This 'transfer graph' (Fig. 16) contains the instances, features and class labels of the target data and the observed auxiliary data as vertices. The edges are set up based on the relations between the auxiliary and the target data nodes. The transfer graph presents a unified graph structure to represent the task of transfer learning from social domain to video domain.

Before diving into the details of the transfer graph, it is important we mention that the novelty of our approach lies in how we incorporate the learned social topics into this transfer graph. We incorporate the learned topic model into the transfer graph by means of a ranked update on the Laplacian matrix representation of the transfer graph. *SocialTransfer is a unique method to combine topic modeling and transfer learning; providing a natural interface for topic modeling to seamlessly fit into the process of transfer learning.*



Interview with Mel Kelly (aka Recyclopath)

Interview with Mel Kelly (aka **Recyclopath**) on the occassion of the punk performer and fetish escort's ArtsSA grant through the Richard Llewellyn ...

by WiderScreenings | 2 months ago | 19 views

I love recycling for the environment but hate the recyclopaths who smoke

Figure 16: Transfer graph for SocialTransfer with connections among auxiliary and target data including features and class labels.

Let us focus on the example in the transfer graph illustrated in Fig. 16. The feature word 'recyclopath'[2] occurs in the training video instance 'Interview with Mel Kelly (aka Recyclopath)' shown in the top right. Since the video lacks any tags related to 'Environment', a traditional learner will find it difficult to extract the topic of this video to be related to 'Environment'. However, the auxiliary data has a tweet instance belonging to the 'Environment' topic having the word 'recyclopath'. Thus, the transfer learner can label this video as 'Environment'-related and associate this video to another 'Environment'-related video. This is an example of discovery of video associations by understanding video topics with the help of social topics.

As shown in Fig. 16, the transfer graph *G(V,E)* consists of vertices representing instances, features or class labels, and edges *E* denoting co-occurrences between end nodes in the target and the auxiliary data i.e.:

$$V = \chi_{train} \cup \chi_{test} \cup \chi_{aux} \cup \mathcal{F} \cup \mathbb{C} \qquad (4)$$

The weight of each edge where one of the end nodes belongs to $\mathbb{C}$ indicates the number of such co-occurrences. Let $\omega_{x,f}$ represent the importance of the feature $f \in \mathcal{F}$ that appears in instance $x \in \chi_{train} \cup \chi_{test} \cup \chi_{aux}$. Then, the weight of an edge where one of the end nodes belongs to $\mathcal{F}$ is indicated by $\omega_{x,f}$. The importance of a feature word $\omega_{x,f}$ can be calculated using the topic-word probability distribution matrix obtained from OSLDA. The total number of features and class label nodes remains fixed in the transfer graph. Let $T(x)$ represent the true label of the instance. If e_{ij} denotes the the weight of an edge between two nodes $\vartheta_i$and ϑ_j in the transfer graph, then edge weights can be assigned as:

[2] `Recyclopath` means a person who is almost paranoid about recycling and is an extreme environmentalist.

$$e_{ij} = \begin{cases} \omega_{\vartheta_i,\vartheta_j} & \vartheta_i \in \chi_{train} \cup \chi_{test} \cup \chi_{aux} \wedge \vartheta_j \in \mathcal{F} \\ \omega_{\vartheta_j,\vartheta_i} & \vartheta_i \in \mathcal{F} \wedge \vartheta_j \in \chi_{train} \cup \chi_{test} \cup \chi_{aux} \\ 1 & \vartheta_i \in \chi_{train} \wedge \vartheta_j \in \mathbb{C} \wedge T(\vartheta_i) = \vartheta_j \\ 1 & \vartheta_i \in \chi_{aux} \wedge \vartheta_j \in \mathbb{C} \wedge T(\vartheta_i) = \vartheta_j \\ 1 & \vartheta_i \in \mathbb{C} \wedge \vartheta_j \in \chi_{train} \wedge T(\vartheta_j) = \vartheta_i \\ 1 & \vartheta_i \in \mathbb{C} \wedge \vartheta_j \in \chi_{aux} \wedge T(\vartheta_j) = \vartheta_i \end{cases} \qquad (5)$$

For all other cases except the ones mentioned in Eq. (5), we set $e_{ij} = 0$. The edge weights thus represent the occurrence/importance of a category or feature present in the auxiliary/target data, which will be eventually utilized as a distance metric during spectral clustering. Some nodes in the graph may be isolated with no edge connections. The matrix updating process adds new edges to the isolated nodes. The transfer graph *G* is usually sparse, symmetric, real and positive semi-definite, which allows the possibility of calculating its spectra efficiently [21]. The graph spectrum in terms of eigenvectors is the impression of the structure of relations among the source and target data. This structural relation between the cross domain data is the essence of transfer learning [23]. Thus, it is necessary to represent the source and target data as a transfer graph and then analyze their structural relation by learning the graph spectrum.

Learning Transfer Graph Spectra

The highlight of *SocialTransfer* is how it learns transfer graph spectra and incorporates new social topics into the transfer graph in real-time. This task is non-trivial, since if not properly done, it may incur substantial costs in terms of scalability (e.g., in eigen-feature extraction) and interoperability (in integration of topics) between topic modeling and transfer learning. In this section, we demonstrate how we achieve both these goals efficiently.

Once the transfer graph *G=(V,E)* is built, we can use graph spectra analysis to form an eigen feature representation, which combines the principal component features from the training and the auxiliary data. In order to extract the top-q eigenvectors of the transfer graph *G=(V,E)*, we first need to convert the graph into a Laplacian matrix. Let $\deg(\vartheta_i)$ denote the degree of the i-th vertex in G. Then the transfer graph Laplacian $L_{input} \coloneqq (l'_{i,j})_{|V|\times|V|}$, can be obtained as:

$$l'_{i,j} \coloneqq \begin{cases} \deg(\vartheta_i) & if\ i = j \\ -1 & if\ i \neq j \wedge e_{ij} = 1 \\ 0 & otherwise \end{cases} \tag{6}$$

If the Laplacian eigen values are represented as:

$$\lambda_0 = 1 \geq \lambda_1 \geq \cdots \geq \lambda_p,$$

then the eigen gap can be defined as: $eigengap = \frac{\lambda_q}{\lambda_q - 1}$.

Since the Twitter stream is extremely dynamic, topics and trends change overtime. This requires a feature extraction scheme that can reflect and scale with the social stream. Previous approaches for spectral feature representation in transfer learning have suggested the use of the normalized cut (Ncut) technique for eigenvector extraction [23]. However, our experiments (Fig. 12 in Section 6.3) showed that the normalized cut technique is incapable of scaling with the twitter stream.

Therefore, we use a Power Iteration technique for computing the q largest eigenvectors of L_{input} [106]. The method begins with a random $|V| \times q$ eigenvector matrix and iteratively performs matrix multiplication and ortho-normalization until convergence [24]. The speed of convergence of this method depends on the eigen gap, i.e. the difference between successive eigen values. In fact, Bach *et.al.* mention that the number of steps required for the orthogonal convergence in the Power Iteration method is $O(\frac{1}{1-eigengap})$ [24].

Since topics are updated in the topic space with time, we need to devise a way to progressively incorporate these new topics into the transfer graph. These topics could be incorporated by picturing them to be a time-dependent labeled bias (like a semi-supervised bias) which is an additional set of labeled instances acting as input supervision. One option for incorporating the semi-supervised topic bias as input supervision into the Laplacian representation of the transfer graph (L_{input}) is by producing a ranked update on L_{input} (see Eq. 5). The update in effect recalculates the weights of edge/path between the features and the corresponding labels within the transfer graph, thus updating the characteristic of the Laplacian (Eq. 2, 3). Essentially, the ranked update on the Laplacian using the topic bias adds positive weights between feature words that share the same topic and adds negative weights between feature words that belong to different topics. Thus, the target and the auxiliary data instances act as sort of virtual nodes enabling this re-weighing of the feature edges.

An additional reason for using the ranked update technique is that previous work [21] has also rigorously demonstrated that when Laplacians such as L_{input} is positive semi-definite, a ranked update can improve eigenvector extraction speed by spreading the eigen gap. The next subsection elaborates on how we use ranked updates to incorporate semi-supervised topic bias and update the transfer Laplacian.

Incorporating Social Topics

We know from topic modeling that the words in tweets can be clustered into topics. Let us consider there are *K* such topic clusters. The semi-supervised topic bias is implemented by assuming we know the correct topic labels for a subset of the feature words. This input is learned by topic modeling using OSLDA, which was described in the previous chapter.

The semi supervised bias consists of a set of topical words for each topic $A_i^{in} \subseteq A_i,$ for i=1,2,…*K* that act as input supervision. Let us consider the simple case of two topic clusters A_1^{in} and A_2^{in}, such that $A^{in} = A_1^{in} \cup A_2^{in}$ denotes the set of labeled bias instances. Also, consider $d_i = \sum_j e_{ij}$ and $vol(A_k) = \sum_{i \in A_k} d_i$. We can then define a regularization vector δ_1 as:

$$\delta_1(i) = \begin{cases} \sqrt{\dfrac{d_i}{vol(A^{in})}}\, f(i) \ , & i \in A^{in} \\ 0 & , \ i \notin A^{in} \end{cases} \tag{7}$$

where, $f(i) = \sqrt{\frac{vol(A_2^{in})}{vol(A_1^{in})}}$ if $i \in A_1^{in}$ and $f(i) = -\sqrt{\frac{vol(A_1^{in})}{vol(A_2^{in})}}$ if $i \in A_2^{in}$.

The effect of the above Eq. 7 is to introduce a quadratic penalty if there is a violation in the topic bias label constraints. Said otherwise, this will cause vertices of features that belong to the same topic to cluster together while vertices of different topics will be assigned to separate clusters (due to the penalty). A rank-1 update on the original Laplacian can be made as:

$$L_{topic\_bias} = L_{input} + \gamma . \delta_1 \delta_1^T \tag{8}$$

Similarly, if there are *K* topics, we can modify the original matrix L_{input} with a rank-k update [21] instead of a rank-1 update. This supervised ranked update firstly allows us to seamlessly incorporate streaming data progressively. Secondly, it aims at tuning certain algebraic properties of the input Laplacian matrix which are related to the convergence rate of the Power Iteration method, eventually speeding the eigen decomposition.

In summary, the input supervision using topics learned from the social stream allows us to implement rank-*k* updates on the transfer-Laplacian matrix as a similarity learning mechanism, where vertex similarities are adjusted on the basis of the topic bias. Note that the number of nodes in the graph is not changed during updating (dimension |V| is fixed); instead the updates only introduce new edges or re-weights existing edges in the graph as it iteratively reuses the eigenvectors from previous update. Due to lack of space, we refrain from describing in detail how the rank-k update improves the speed of eigenvector extraction. In fact, the ranked update increases the eigen gap, which accelerates the convergence of the Power Iteration method. For a detailed explanation of how a supervised bias using rank-k update accelerates the eigenvector extraction process, please refer to [21].

Algorithm for SocialTransfer

Once the first q eigenvectors $\boldsymbol{E}_1, \boldsymbol{E}_2, \ldots, \boldsymbol{E}_q$ have been found by iteratively using the Power Iteration method with the topic-based input supervision, we can form a combined feature representation that depends on both the training and the auxiliary data. Traditional learners like SVMs can use the combined features that include the transfer task to train a classifier *f'*(χ_{test}). Described below, is the algorithm for *SocialTransfer* for classification in the target domain based on auxiliary social stream data.

Algorithm 1: *SocialTransfer* – Transfer Learning from Social

Input: A target classification task which includes the target training data set χ_{train}, the source auxiliary data set χ_{aux} and the target test data set χ_{test}.

Output: Classification result on χ_{test}

1. Construct the initial transfer graph *G(V,E)* based on the social transfer clustering task (c.f. Section IV.D).
2. Calculate transfer Laplacian matrix: L_{input} from *G* using Eq. (3).
3. **for** each chunk of tweets entering the system **do**
4. Calculate the regularization vector δ_1 using the input supervision of social topics A^{in} as shown in Eq. (4).

```
5. Perform semi-supervised topic bias update on transfer Laplacian: $L_{topic\_bias} = L_{input} + \gamma . \delta_1 \delta_1^T$ as shown in Eq. (5).
6. Use Power Iteration to calculate the first $q$ eigenvectors of $L_{topic\_bias}$: $\boldsymbol{E}_1, \boldsymbol{E}_2, \dots, \boldsymbol{E}_q$ which satisfy the generalized eigenproblem: $L_{topic\_bias}\boldsymbol{E} = \lambda U \boldsymbol{E}$. The resulting eigenvectors will be used as initial eigenvectors for the next updated Laplacian matrix.
7. **end for**
8. Construct matrix *H* with $\boldsymbol{E}_1, \boldsymbol{E}_2, \dots, \boldsymbol{E}_q$ as columns.
9. **for** each $x_{ts}^{(m)}$ in $\chi_{train}$ **do**
10.     Let $u_{tr}^{(m)}$be the corresponding row in *H* w.r.t $x_{tr}^{(m)}$.
11. **end for**
12. Use a traditional classification algorithm (we use SVM) to train the classifier *f'*($\chi_{test}$) based on $\mho_{tr} = \{u_{tr}^{(m)}\}_{m=1}^{M}$ instead of the original training set $\chi_{train} = \{x_{tr}^{(m)}\}_{m=1}^{M}$ and then classify $\chi_{test} = \{x_{ts}^{(n)}\}_{n=1}^{N}$ in the eigen feature space.
```

4.3 Applications

We present three applications based on the *SocialTransfer* algorithm and the OSLDA topic modeling.

4.3.1 Socially Relevant Video Recommendation

Modern video publishing sites like YouTube use related video recommendation techniques [8, 9] based on RVGs to recommend a video to the user. The recommended video is related to the seed video which the user is currently watching by co-clicks or co-views, i.e. some signed-in user has clicked on both the seed and the related video in sequence. In contrary, Socialized Video Recommendation recommends videos which bear similar topics to a seed video the user is watching. Thus, proposed Social Video Recommendation is independent of the click through nature among videos, and has several advantages over traditional related video recommendation [25, 26], such as: (1) it considers video content/context (as in topics) while recommending videos, (2) its performance does not decrease as click data gets sparse, (3) it can recommend fresher videos that do not have significant user activity but are extremely relevant to the seed video and (4) it does not require signed-in user activity to learn and build RVGs.

For related video recommendation, the system must be able to predict which videos are 'related' to a seed video and are good candidates for recommendation. The first step in solving this is to detect the topic of the seed video. Thus, the job of the classifier is to classify the topic of a test seed video. Once we detect the topic of the test seed video, we can assume all the videos belonging to that topic are candidates for related video recommendation of the seed. We then recommend only those videos from the candidate pool whose tags match the seed. A socialized video recommender can be developed by creating a learner that uses auxiliary tweet data by means of transfer learning. Given a set of RVG videos in the target domain, a traditional Non-Transfer learner like SVM [27] will aim to predict the related videos of a given seed video in the test data set by building a classifier only from the training data. Instead, SocialTransfer builds a classifier using both training video data and auxiliary tweet data.

Data Description: Our study is based on a 5.7 million videos crawled from YouTube and 10.2 million tweets obtained from the NIST Twitter dataset [18]. The source domain is Twitter and the target domain is YouTube. The varying bursty nature of tweets can be observed from Table 3. We use a preliminary list of YouTube related video ids collected for experiments in [8]. Video meta-data includes values for entities such as video id, title, tags, view count, age (in days since uploading), category, related video ids (which comprises related videos at depth 1 of RVG) etc. As mentioned earlier, the videos related to a given seed video is captured using a directed graph, which is known as Related Video Graph (RVG) [26, 28]. Thus, if a video y is in the related video list of a seed video x, then there is a directed edge/path $x \rightarrow y$ in the RVG [25]. Moreover, for an edge $x \rightarrow y$ in the RVG, its tags can be represented as the instance: $\{tags(x)\} \rightarrow \{tags(y)\} - \{tags(x)\}$, where $\{tags(x)\}$ represents the set of tags of video x and '$-$' means set difference. All such instance/label combinations of the B videos have to be randomly divided into two sets for training and testing, called χ_{train} and χ_{test} respectively; where $|\chi_{train}| = M$ and $|\chi_{test}| = N$ represented as $\chi_{train} = \{x_{tr}^{m}\}_{m=1}^{M}$, $\chi_{test} = \{x_{ts}^{n}\}_{n=1}^{N}$ where $M + N = B$ and $\frac{M}{N}$~1.5.

We have collected related video information up to five depths from an initial seed video ranging across the 14 main YouTube categories: Comedy, Entertainment ('Enter'), Education ('Edu'), Music, Film & Animation('F&A'), Non-Profits & Activism ('NonProf'), Science & Technology ('S&T'), Travel & Events ('T&E'), Pets & Animals (P&A), HowTo & Style ('H&S'), Autos & Vehicles ('A&V'), News & Politics ('N&P'), Sports, and People & Blogs ('P&B'). Some videos are categorized unavailable, and in such cases we use the category of its parent video. Apart from these main categories, [29] has suggested around 75 sub-categories to the main YouTube categories. We include all of these as the pool of categories from which class labels can be drawn. Therefore, we tune the OSLDA to detect tweets where the tweet words fall into the tag space belonging to any of these category videos.

Since RVGs are essentially related recommendation networks, distribution of categories over videos changes as we move one depth to the next. This introduces some degree of intended diversity in the next video recommended [25], since it might be of a different category compared to the seed but somehow related. On average, we found that the next recommended video has 25% chance of being in the same category as the seed video. Fig. 17 shows the category distribution of related videos at depth 1 and 2 from the seed video being watched.

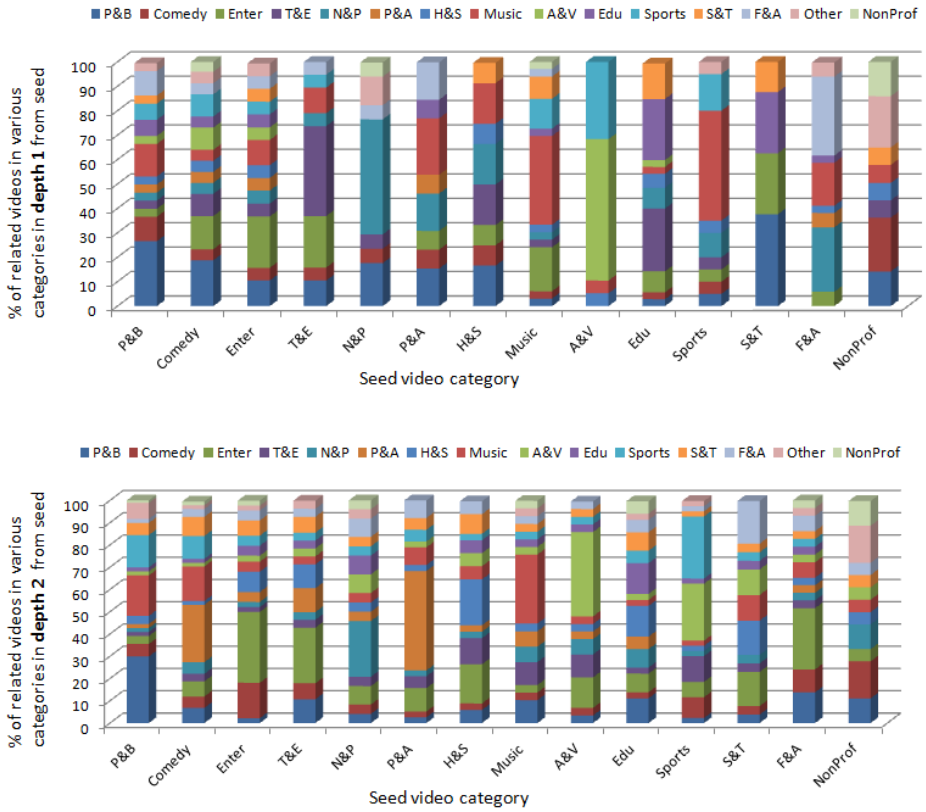


Figure 17: Distribution of video categories of recommended videos by RVG at depth 1 and 2 from the seed video showing the diverse nature of video recommendation in YouTube.

The Twitter dataset consists of 10.2 million tweets generated in the US and collected between Jan 26th 2011 and Feb 11th 2011. We simulate the twitter data as a stream, with each batch of tweets representing approximately 5 minutes. The resulting rate at which tweets streams over the last week of Jan, 2011 is shown in Fig. 18, where the 5 min batch time slots account for a total of 288 slots spanning 24 hours in the horizontal axis. We show the temporal stream volume (in no. of tweets generated) distribution only across seven days in order to avoid cluttering in Fig. 18.

From Fig. 18 we can conclude that under normal circumstances, the tweet rate distribution has a general pattern over 24 hours: there is a minima around 8:15 AM, followed by a gradual rise until 3 PM in the afternoon, where a local maxima is achieved. Interestingly, another spike is usually noticed in tweets around 2:30 AM in the morning. The drops to almost zero on Jan 29th can be accounted for by Twitter downtimes and the Blackberry outage in USA. The high spike around 5:15 PM on Jan 29th is caused due to a high volume of tweets during the onset of the Egyptian revolutions.

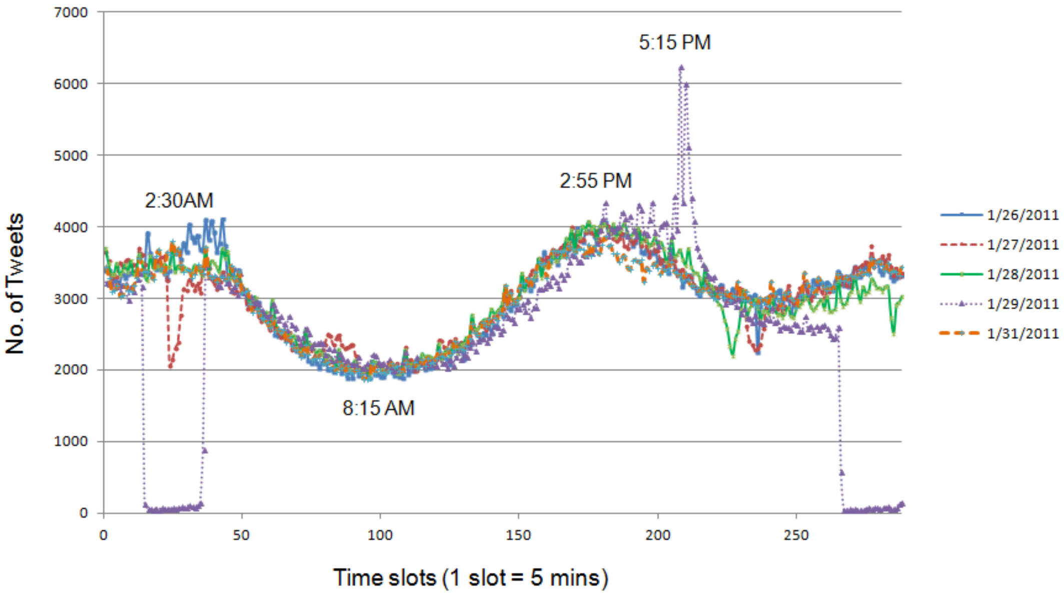


Figure 18: Daily tweet stream from 26th – 31st Jan, 2011.

Experimental Results: For socialized video recommendation, we test *SocialTransfer* against a traditional learner like SVM [27], where *SocialTransfer* uses auxiliary social data in combination with training data, whereas a traditional learner uses only the training data for prediction (called Non-Transfer) and serves as our benchmark.

Here, the classification task is simple: given a seed test video, classify whether another video is a related video of the seed or not. This in aggregation is same as the case: given a seed test video, predict the list of related videos for the seed test video.

For the experiments, we set γ=1.25, limit the power method to extracting top-25 eigenvectors and include 60% of the topic space for input supervision. The reasoning of these choices is explained over the following sections. We have three datasets for transfer learning - the target training data, the target test data and the source auxiliary data. The target dataset consists of 5.7 million videos in total along with their RVGs (contains a list of related videos for each seed video). However, some videos do not have categories or are removed from YouTube, and therefore we experiment on a reduced set of 4.8 million videos. Our training data consists of 60% videos randomly picked from the 4.8 million YouTube videos. The rest 40% videos (~2 million) are used for testing. As auxiliary data, we use the 10.2 million tweets from the Twitter stream.

# Topics Extracted / **# Tweets	**5 topics**	**10 topics**	**30 topics**	**60 topics**
30 K Tweets	0.66	0.72	1.09	1.67
60 K Tweets	1.41	1.83	2.11	3.31
90 K Tweets	1.75	1.93	3.18	5.19
120 K Tweets	2.53	3.1	4.42	6.81

Table 3: Number of video instances in popular categories.

We ensure to extract topics from tweets based on approximately 90 categories (16 main + 75 other) so that the source and target domains share same categories. Additionally, we also evaluate category-specific predictions based on six popular categories (Comedy, Film & Entertainment, Sports, People & Blogs, Music). Table 3 shows the number of video instances used for evaluation in some of the popular categories.

In Table 4, we report the average error in prediction for the Non-Transfer cases (SVM on training only) vs. *SocialTransfer*. Non-Transfer refers to application of the traditional SVM learner to the original target dataset with no social influence (only training features are used); *SocialTransfer* means to apply SVM on the combined feature representation learned using transfer learning from social data (training + auxiliary).

$\boldsymbol{Category} \rightarrow$ $\boldsymbol{Approach} \downarrow$	**Overall**	**Comedy**	**Film & Animation**	**Entertainment**	**Sports**	**People & Blogs**	**Music**
Non-Transfer	$\mathbf{0.357 \pm 0.049}$	$\mathbf{0.429 \pm 0.059}$	$\mathbf{0.334 \pm 0.063}$	$\mathbf{0.386 \pm 0.023}$	$\mathbf{0.394 \pm 0.072}$	$\mathbf{0.247 \pm 0.066}$	$\mathbf{0.356 \pm 0.036}$
SocialTransfer	$\mathbf{0.232 \pm 0.043}$	$\mathbf{0.397 \pm 0.065}$	$\mathbf{0.242 \pm 0.051}$	$\mathbf{0.219 \pm 0.015}$	$\mathbf{0.282 \pm 0.082}$	$\mathbf{0.112 \pm 0.032}$	$\mathbf{0.230 \pm 0.029}$

Table 4: Number of video instances in popular categories

The performance in Table 4 is measured in error rate by averaging 10 random repeats on each dataset by the two evaluation methods. For each repeat, we randomly select 5000 instances per category as target training data. We report the prediction error rate in each of the main categories, along with the overall error for the entire data set. We also report the standard deviation of the repeats in Table 4. The two methods are well-tuned using 10-fold cross validation. The overall gain using *SocialTransfer* is ~ 35.1% compared to non-transfer cases. Please note that the overall error rate is averaged over all the main categories and not just the six categories shown in Table 4. Performance improvement using transfer learning is most in category ‘People & Blogs’. In all the major categories, *SocialTransfer* performs better than a traditional non-transfer learner.

4.3.2 Socially Video Popularity Prediction

In this section, we discuss how to utilize the SocialTransfer in calculating the social prominence of a video and estimate its social popularity. The steps include: (A) calculate the trending score for each topic (called Tscore) and use SocialTransfer classification to find the principal topic of a video. The trending score of the principal topic of a video is its social prominence; and (B) fusing social prominence of a video with its traditional popularity (based on view count) to estimate the final trend aware popularity score (*TAP*). (C) The final goal of this work, predicting which videos will demonstrate bursty nature based on their *TAP*.

Social Prominence: Trends are temporal dynamic entities, meaning they grow for a certain period of time, after which they suffer inevitable decay. In other words, trends remain socially prominent for some time and their attractiveness fades away. It is therefore necessary to include a time decay factor when modeling the trending score.

More formally, consider *SocialTransfer* receives a set of *D* tweets in one time slot; t_{cur} being the current time slot and t_{onset} is the time slot when the trend was first observed. We can then define the trending score of a topic *z* as:

$$Tscore_z = \frac{\sum_{k=1}^{|D|} P(z|d_k, t_{cur})}{|D|.\,\delta_z} \qquad (9)$$

where $\delta_z = \varphi(t_{cur}, t_{onset})$ is the time dependent decay factor which is a function of the current time slot and the time slot when the trend was first seen. The decay factor must actively respond to trend reoccurrences (i.e. when the trend rises after an initial fall). The decay can be formulated as:

$$tr = \begin{cases} 1, & P(z|D,t_{cur}) \geq P(z|D,t_{cur}-1) \\ 0, & P(z|D,t_{cur}) < P(z|D,t_{cur}-1) \end{cases}$$

$$\delta_z = \begin{cases} 1, & t_{cur} = t_{onset} \\ \delta_z, & t_{cur} > t_{onset}\ and\ tr = 1 \\ \delta_z + \eta, & t_{cur} > t_{onset}\ and\ tr = 0 \end{cases} \tag{10}$$

where $0 < \eta \leq 1$ depends on the category of the topic z (meme, music etc.). In addition to the usual trends, active decay can capture extremely dynamic trends like memes or sports related topics, which have short life spans compared to music or entertainment related trends.

For some video v, let z_v^* be the topic to which the video has maximum membership. This membership measure can be easily retrieved using SocialTransfer classification, since the output of the classification is the topic of the video. Then the *social prominence* of video *v* is $Tscore_{z_v^*}$.

Trend Aware Popularity: In a traditional video ranking system (like in YouTube) videos with higher view counts are boosted in the rank list [30]. Thus, these videos get clicked more often, resulting in subsequent higher view counts for them [9]. Therefore, it is necessary to engineer a reasonable fusion of the traditional approach and our proposed social prominence approach. This fusion of the traditional popularity factors (like view counts) and the social prominence of the video is called the Trend Aware Popularity (*TAP*).

In formulating the final popularity score, we also need to take into account the time when the video was uploaded (t_{upl}) since we need to discount the fact that older videos already have higher view counts. Thus, the net temporal Trend Aware Popularity score that we assign to a video *v* is:

$$TAP_v = \gamma.TScore_{z_v^*} + (1-\gamma).\frac{t_{onset} - t_{upl}}{t_{cur} - t_{upl}}.\#(vc)_{t_{onset}} \tag{11}$$

where $\#(vc)_t$ represents the view count at time t and γ is a weighting factor that balances social vs. traditional popularity control. The above equation measures the social trend aware popularity of a video. The traditional popularity is reflected by the adjusted view count measure, which fractions the view count of a video based on when the video was uploaded in video domain, when the video topic trend was onset in social domain and when the prediction was performed.

The *TAP* score reflects the social popularity as well as the traditional (video domain) popularity for a certain video. Our hypothesis is that social popularity signal penetrates across media domains on the Internet. In other words, if a topic is substantially popular (trending) in the social domain, then media belonging to the same topic will gain popularity in other domains (in this case, video domain). Therefore, a ratio of *TAP* to a scaled $TScore_{z_v^*}$ value will provide us with the quantitative estimation of the impact of the social signal in boosting the overall video popularity for some video *v*. The lower the value of this ratio, the higher the impact of the social prominence of the video in comparison to the adjusted view count score. Given the same social prominence, the ratio seems to favor videos with lower adjusted view count measure. However, this is not an issue, since the adjusted view count measure is lower when the trend has been seen for longer time period ($t_{cur} - t_{onset}$), which practically means that we are more sure of the prediction if we are exposed to more of past trend data. Thus, for a certain video, if this ratio is significantly lower than for others (lower 10th percentile), we predict the video will gain bursty popularity.

Experiments: Once again, we test our social transfer learning model against traditional learners like SVM [27] which do not use any auxiliary social data in prediction. We used LibSVM with the Radial Basis Function kernel for SVM implementation [108]. Here, the classification task is: given a test video, classify whether it is bursty or not (bursty=1/0). For the experiments, we set γ=1.25, limit the power method to extracting top-34 eigenvectors and include 60% of the topic space for input supervision.

To measure the performance, we use error rate as a metric. Error rate is calculated as (1 - $accuracy$) where,

$$accuracy = \frac{\#truePositives + \#trueNegatives}{\#truepositive + \#trueNegatives + \#falsePositives + \#falsenegatives}$$

$Category \rightarrow$ $Approach \downarrow$	**Overall**	**Comedy**	**Film & Animation**	**Entertainment**	**Sports**	**People & Blogs**	**Music**
Non-Transfer	0.524 ± 0.031	0.623 ± 0.039	0.412 ± 0.033	0.386 ± 0.028	0.451 ± 0.062	0.324 ± 0.056	0.576 ± 0.028
SocialTransfer	0.311 ± 0.026	0.328 ± 0.043	0.389 ± 0.031	0.289 ± 0.022	0.225 ± 0.074	0.197 ± 0.029	0.236 ± 0.017

Table 5: Experimental Results of Error Rate in Predicting Bursty Videos for Social Video Popularity. The results are the averages of 10 random repeats along with their standard deviations. Both methods are tuned with 10-fold cross validation.

In Table 5, we report the average error in prediction for the Non-Transfer cases (SVM on training only) vs. *SocialTransfer*. Non-Transfer refers to application of the traditional SVM learner to the original target dataset with no social influence (only training features are used); *SocialTransfer* means to apply SVM on the combined feature representation learned using transfer learning from social data (training + auxiliary). The performance in Table 5 is measured in error rate by averaging 10 random repeats on each dataset by the two evaluation methods. For each repeat, we randomly select 5000 instances per category as target training data. We report the prediction error rate in each of the main categories, along with the overall error for the entire data set. The results are provided category specific to show that the algorithm does better in certain video categories, potentially due to the fact that more information about those categories can be extracted from the social media in the first place. We also report the standard deviation of the repeats in Table 5. The two methods are well-tuned using 10-fold cross validation. The overall gain using *SocialTransfer* is ~ 39.9% compared to non-transfer cases. Please note that the overall error rate is averaged over all the main categories and not just the six categories shown in Table 5. Performance improvement using transfer learning is most in category ‘Music’. In all the major categories, *SocialTransfer* performs better than a traditional non-transfer learner. The F1-score of positive bursty videos for the proposed *SocialTransfer* algorithm is 0.68 whereas for the non-transfer SVM it was 0.32.

Additionally, we ran a baseline Naive Bayes classifier, which produces an F1 score of 0.21 without any transfer of auxiliary data. If we replace the SVM in *SocialTransfer* with the Naive Bayes, the F1 score achieved is 0.49. The drop in performance of Naive Bayes in both transfer and non-transfer cases compared to SVM (-0.19 and -0.11 respectively) is expected. Naive Bayes is easy to implement, but it suffers from strong feature independence assumptions. Notice that this feature independence assumption is more costly in the transfer scenario, where the drop in performance is larger than in non-transfer scenario, potentially due to the heavy reliance of *SocialTransfer* on cross-domain feature alignment.

We also provide results of using a majority-class baseline classifier (in place of SVM in Algorithm 1). The F1 score of the final bursty video prediction in this case is 0.111. The distribution of bursty and non-bursty video in our dataset in 17% and 83% respectively. Thus, a majority-class baseline classifier, when directly applied to bursty video prediction, will classify every test video as non-bursty.

4.3.3 Social Query Suggestion for Video Search

Let us first describe an application that utilizes the topic modeling using OSLDA within the *SocialTransfer* framework. Our intuition is that lack of a collaborative cross-domain recommendation environment compels users into unguided video search (pure querying rather than smart recommendation). One effect of such activity is that users will use the words of trending issues and topics (topical words) when performing video search queries on the Internet. Learning topical words in real time from social streams could be leveraged to suggest queries for video search. We believe this is an important application of real time topical analysis from social streams. Our experimental results suggest: (1) user search queries in video search engines do contain words which we recovered as topical words from social streams using OSLDA. (2) There is a noticeable time lag between (a) OSLDA topic trend detection from social stream and (b) the increasing volume of search queries on that trend in an external (non-social stream) video search portal. This correspondence can be leveraged to augment user experience by socialized query suggestion for video search when the user is querying in the video portal.

Socialized query suggestion for video search using the OSLDA model in *SocialTransfer* aims to recommend good query words in response to users' query keywords. This will help searchers to better seek the more topic-relevant videos they are looking for, since the suggested topical words are connected to videos in the transfer graph. Said alternately, socialized query suggestion aims to localize the topic of the video the user is querying for by suggesting additional topical words. This is more effective in relevant video retrieval than just matching query keywords to video tags. Therefore, the prior knowledge of which query words the users' will use for video search will not only enable the system to suggest better topical words for the user, but also improve the system's capability in predicting which keywords the users will use for search and which videos they will potentially watch in the future.

Experiments: Experiments in this section are conducted using video query logs from a commercial video search engine and 10.2 million tweet data. The goal is to find a temporal pattern or common terms between tweet topic words and video search keywords from video logs. Fig. 19 shows the distribution of search queries with time in video query logs for the topic '*Egypt'* with real-time trend variation on Twitter as detected by OSLDA.

From Fig. 19, we clearly notice that there is few minutes time lag between a trend topic appearing on Twitter, and the same topical words being searched on the commercial video search engine. This means as trends rise and fall in Twitter, the volume of queries on the same topic rises and falls for video search. To further support our claim that people search for Twitter trends outside Twitter, Fig. 20 shows the query keywords used in a commercial video search engine on Feb 11, 2011. If we eliminate daily searches such as 'cats', 'movies', 'funny commercials' which are common (green dotted circles), then it is hard to miss that topical words (red solid circles) take up a significant portion of the remaining video search keywords. In the video search engine logs and for all queries on Feb 11th that are not daily search terms (like 'cats'), 63% of query words were detected by OSLDA.

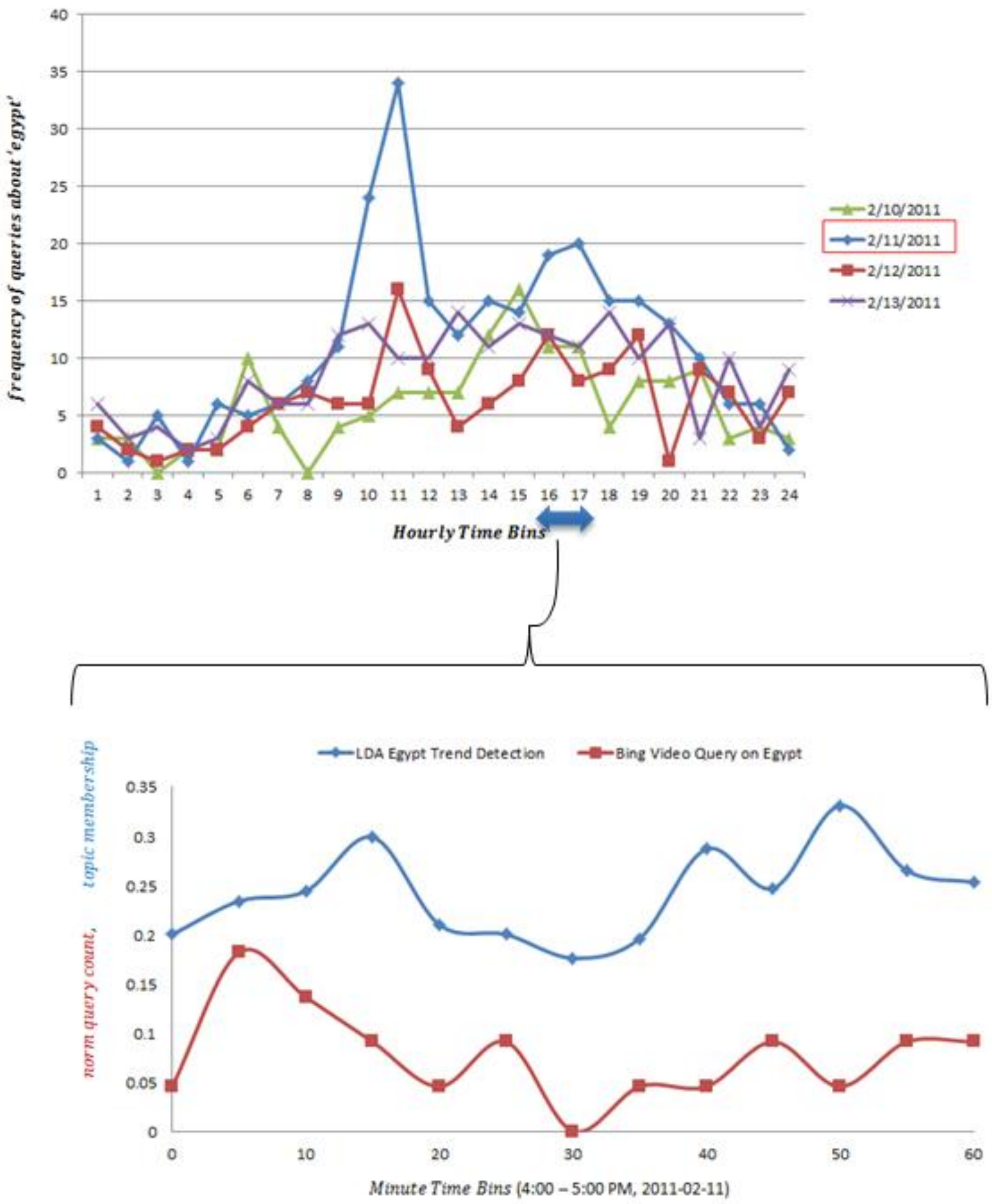


Figure 19: OSLDA trend detection on Twitter (top blue) vs. topical word search trend in commercial video search (bottom brown).

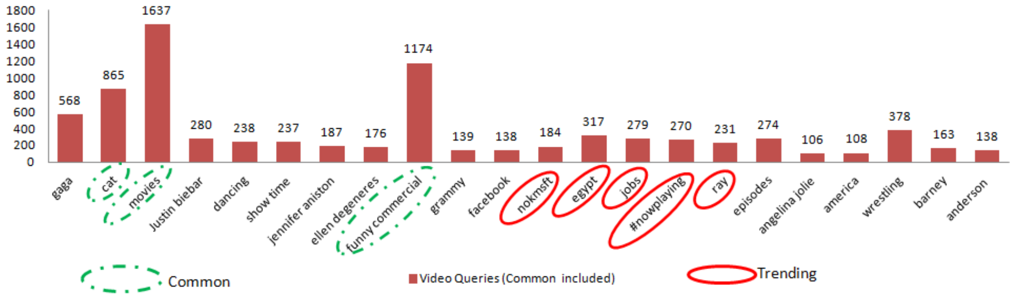
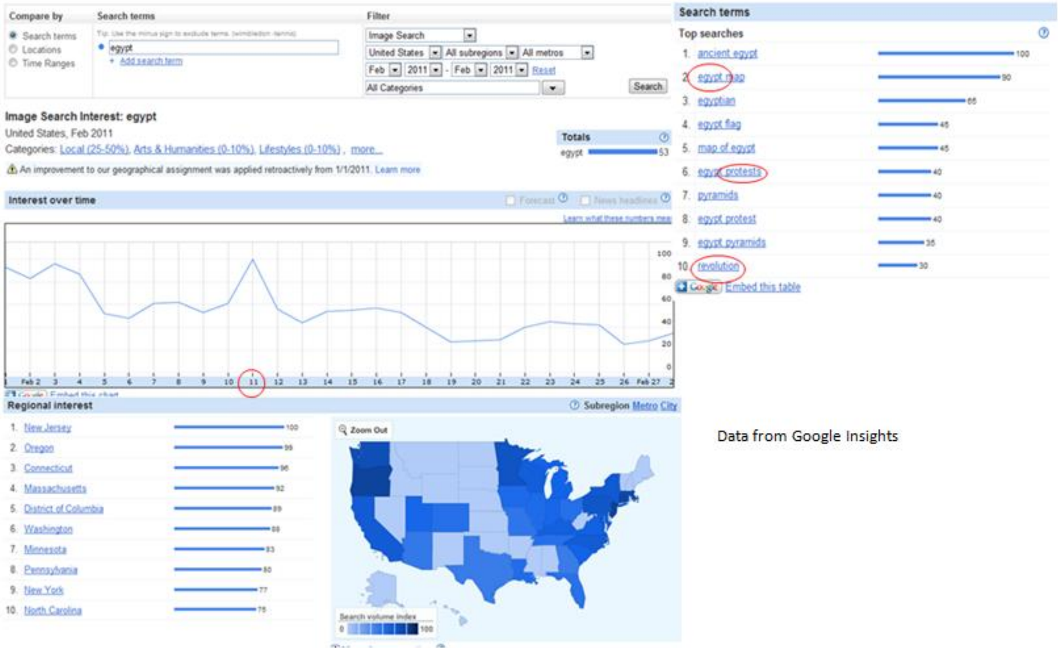


Figure 20: A significant majority of video search keywords come from trending topical words (red circles).



● Search terms

● egypt

Image Search | United States | All subregions | All metros | Feb | 2011 - Feb | 2011 | Reset | All Categories

Image Search Interest: egypt

United States, Feb 2011

Categories: Local (25-50%), Arts & Humanities (0-10%), Lifestyles (0-10%) , more...

An improvement to our geographical assignment was applied retroactively from 1/1/2011. Learn more

Totals	
egypt	53

Interest over time

Search terms

Top searches

1.	ancient egypt	100
2.	egypt map	90
3.	egyptian	65
4.	egypt flag	45
5.	map of egypt	45
6.	egypt protests	40
7.	pyramids	40
8.	egypt protest	40
9.	egypt pyramids	35
10.	revolution	30

Regional interest

Subregion Metro City

1.	New Jersey	100
2.	Oregon	99
3.	Connecticut	96
4.	Massachusetts	92
5.	District of Columbia	89
6.	Washington	88
7.	Minnesota	83
8.	Pennsylvania	80
9.	New York	77
10.	North Carolina	75

Data from Google Insights

Figure 21: Data from Google Insights shows that words detected by OSLDA where among the top searches on Google.

In fact, this technique of socialized query suggestion can be extended beyond video search. We used Google Insights to understand search patterns for web and image search on Feb 11, 2011. It was not surprising that ‘*Egypt*’ was the hottest search topic that day. Moreover, Google Web Insights provided us with the top ten web search keywords related to ‘*egypt*’; seven of which had already been detected by OSLDA earlier. For Google Image search results shown in Fig. 21, six of the top ten search keywords were detected by OSLDA. This is convincing evidence that the OSLDA detects relevant socially active topics within the *SocialTransfer* framework.

4.4 Parameter Tuning

Accuracy Variation with Stream Inflow

We test the rate at which the prediction error decreases with incoming stream of tweets every day across 12 days of the social data (Jan 26th – Feb 7th). Fig. 22 shows that there is a gradual decrease in error rate as more of the stream is seen by *SocialTransfer*. Lack of any sharp drops hints at the fact the social popularity is significantly trend category specific. On course of the 12 days, we see a 49.4% net reduction of error.

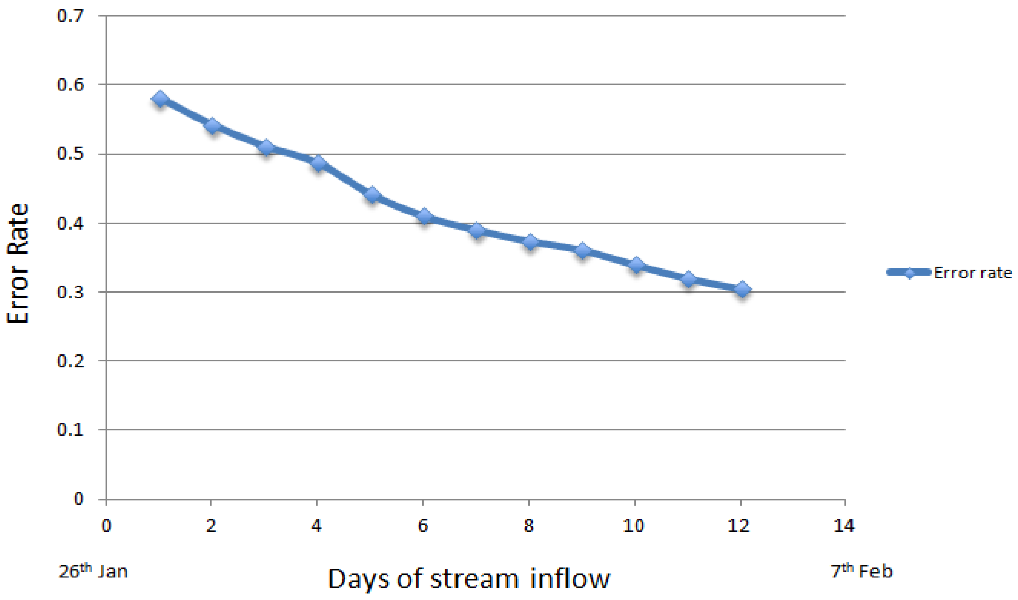


Figure 22: Drop in prediction error rate with daily stream inflow from Twitter.

The classification is done continuously at various time points. This is why the decrease in error can be tracked each day as shown in Fig. 22. However, the results shown in Table 5 are calculated at the end of the entire period of time for which the dataset is available (26th Jan - 7th Feb).

EigenVectors:

Previously we mentioned that for the experiments, we fix the number of eigenvectors to be extracted from the transfer Laplacian to 34. The reason for this choice is due to results of Fig. 23, which shows the variation of the error rate with the number of eigenvectors extracted. We see that when the number of eigenvectors extracted is greater than 34, the error rate is almost constant.

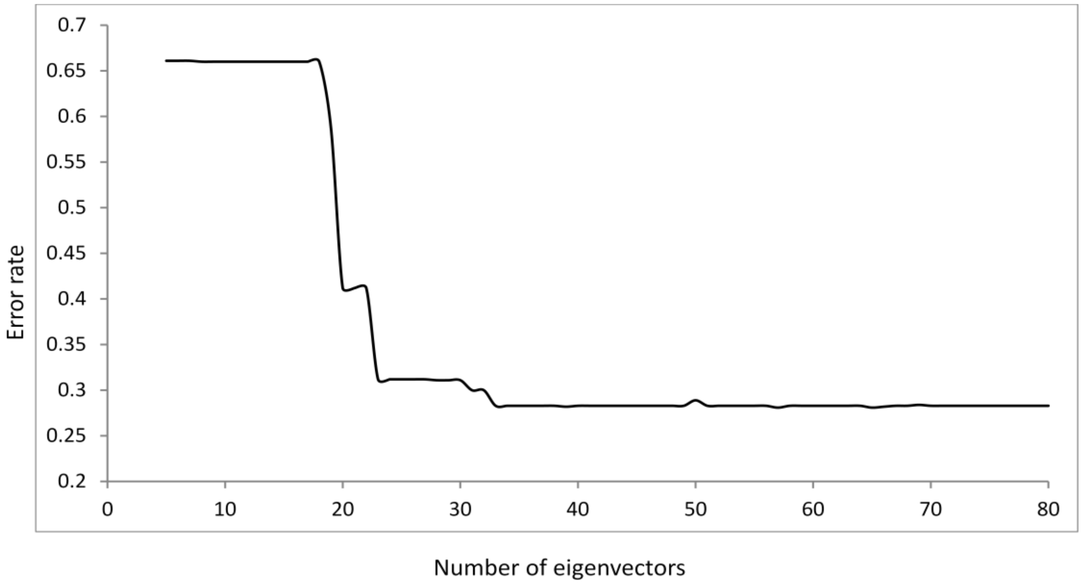


Figure 23: The influence of the number of eigenvectors extracted on the error rate.

However, there is a trade-off between the time duration required for extraction vs. error rate of prediction for a certain number of eigenvectors that can be extracted. Thus, since the variation of reduction in error rate is not significant beyond 33-35 eigenvectors, we can safely assume that the extraction of more than 34 eigenvectors is not necessary.

Scalability

The speed at which the incoming stream of tweets is explored for topics by OSLDA together with the time required for eigen feature extraction from the transfer graph using spectral learning is important for maintaining scalability with the real-time social stream. In our system, the topic modeling is done in parallel with the eigenvector extraction to save time. Thus, our main aim should be to limit the time required to complete either of these tasks within the incoming tweet flow time.

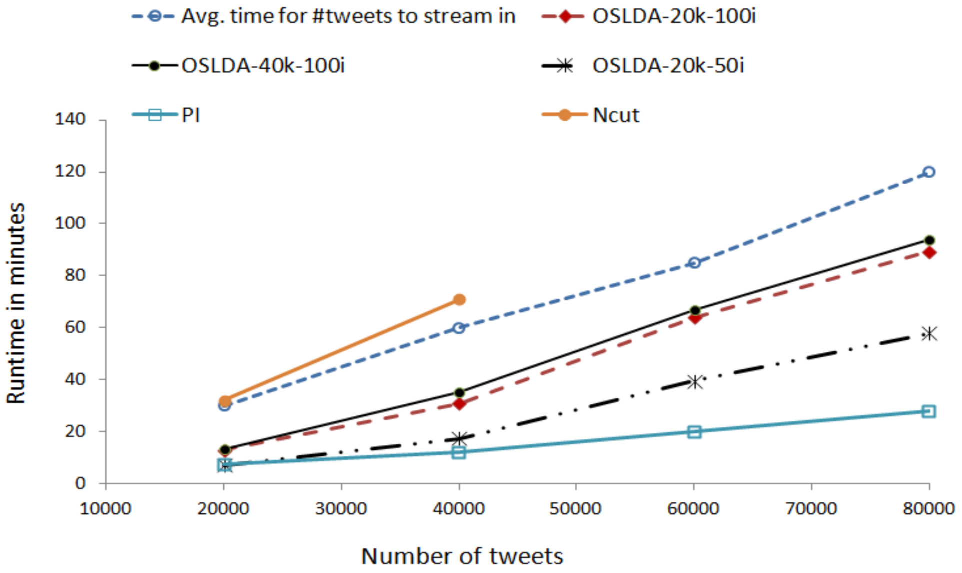


Figure 24: Runtime comparison for topic modeling and eigen decomposition with incoming tweet stream in *SocialTransfer*.

Fig. 24 shows the comparison of runtimes for various settings of OSLDA, eigenvector extraction using power iteration (PI) and eigenvector extraction using Normalized cut (Ncut) with the time taken on average for an incoming chunk of tweets to stream in. For OSLDA, '20k' (in legend) refers to 20 topics extracted and '50i' refers to 50 iterations of the generative process. Experiments were run on a IBM server with 2.67 GHz processor and 8 GB RAM.

From Fig. 24, we can safely conclude that the model scales to incoming bursts of tweets, since the matrix decomposition with Power Iteration and the topic modeling using OSLDA require less time than the speed of incoming tweets. Note that the Normalized cut method (Ncut) does not scale as it takes longer time to extract eigenvectors than the speed of the incoming burst of tweets as shown in Fig. 10. Moreover, for more than 40,000 tweets, Ncut causes our system to run out of memory.

CHAPTER 5: LEARNING FROM SEMANTIC DATA

Semantics, in its classical sense, refers to meaning in information that can be easily interpreted. When data is organized in such a way that it can be interpreted meaningfully without human intervention, we call it Semantic data. There are various ways to structure data so that machine-to-machine communication is fruitful. Most commonly though, semantic data is organized in terms of a Resource Description Framework (RDF), where each entry contains a piece of data instance, its property and the corresponding value that the data instance has for the given property. RDF data is often found in the Linked Data resources online, especially in sites like DBpedia.

A parallel view of 'semantics' is that one instance of data can never be semantic. Ideally, semantics is captured by the relationship between two data instances. Such relationship are easily captured in graphs, were nodes represent data resources and edges represent the relationship between them with respect to some property. Several fundamental problems encountered in automated search, ranking, disambiguation etc. can be handled effectively using results from graph theory. Therefore, the first step in using semantic data is to create a concept graph. We call this concept graph, semNet.

5.1 Building the Concept Graph (semNet)

A semantic network is a graph that represents semantic relations between concepts. WordNet is a popularly used semantic network. When concepts are represented by resources in RDF data, we can call the graph a semantic RDF network. Concepts can be obtained from ontologies. For example, DBpedia is an RDF dataset containing structured information extracted from Wikipedia [37]. It has been widely used in the research community to discover unknown relations in data, develop interoperable Linked Data applications and perform exploratory search and recommendation [38]. RDFs can be visualized as a semantic network, where each node is a resource from an RDF entry. RDFs are the building blocks of the semantic web, and Semantic RDF networks (also called ontology graphs in some communities) can be traversed to detect concept relations [39, 40]. Further, connecting the social web with the semantic web holds valuable promise as it gives rise to collective knowledge systems [41].

To incorporate RDF entries into a graph, we use each RDF resource is treated as a node, the RDF property as the edge label and RDF value as a node connected to resource node. Then using Algorithm 2, we can build a graph *G(V,E)* representing the semantic graph. In Algorithm 2, $label(e)$ refers to the label of the edge *e* i.e. the edge attribute.

Algorithm 2: Semantic knowledge graph from DBpedia

```
Input: RDF dataset ($R$)
Output: Semantic knowledge graph ($G$).
Initialize concept graph $G(V,E)$ where $|V|=|E|=0$.
while more unread RDF entries exist in $R$ do
  Pick an unread RDF entry, mark it as read, represent it as: <Resource><Property><Value>.
  $rNode =< Resource >$.
  $vNode = < Value >$
  if $rNode$ not in $V$:
      Add a new node named $rNode$ to $V$
  if $vNode$ not in $V$:
      Add a new node named $vNode$ to $V$
  Add a new edge $e$ to $E$ s.t. $e = (rNode, vNode)$, i.e. $e$ connects the resource and value nodes.
  $label(e) = < Property >$
```

To include newer RDF datasets into the semantic graph, we can modify Algorithm 2 as: (a) add the previous semantic concept graph G_{prev} to the input and (b) in Step 1, initialize $G = G_{prev}$. This enables us to build a huge concept graph consisting of concepts and their relations as obtained from DBpedia RDF datasets, a part of which is shown in Fig. 2. Once semNet is constructed, we use some additional subroutines to extract required information, e.g. $path(node1, node2)$ retrieves the *shortest* path between nodes $node1$ and $node2$. These simple utility-type algorithms (like shortest path etc.) are not mentioned in this paper to maintain brevity.

Source RDF Dataset[3]	Nodes (in millions)	Density = (2*|E|)/[|V|*(|V|-1)]
SKOS	0.5	0.0009
Homepages	0.41	0.0007
Titles/ Labels	7.44	0.0015
Short Abstracts	3.31	0.0028
Images	1.72	0.0023
Wikilinks	8.68	0.0017

Table 6: Parts of the Semantic Graph built from DBPedia RDF datasets

The concept similarity between two nodes in the network is identical to the semantic similarity of the concepts represented by these nodes; and can be calculated by using either the WordNet (WN) similarity metric [42] or the Normalized Google Distance (NGD) [43]. The WN similarity metric is calculated using the distance of the path length

[3] For dataset specific information, refer to: http://wiki/dbpedia.org/Downloads37

between the two concepts in WN. We use the Resnik measure based for our purposes, i.e. a lowest common subsume is to be detected in the WN taxonomy which is shortest distance from the two concepts to be compared. The larger is the distance to this lowest common subsumer, the smaller the similarity.

Sometimes, WN fails to retrieve the required similarity measure. In such cases, we use the NGD to calculate the similarity. Unlike WN, which is handmade ontology of words/concepts, NGD is derived from the number of hits returned by Google search for a specific set of keywords. Thus, keywords which are semantically similar tend to have small Google distance. NGD can be mathematically defined as:

$$NGD(x,y) = \frac{\max\{log\, f(x), log\, f(y)\} - \log f(x,y)\}}{\log T - \min\{\log f(x), \log f(y)\}} \qquad (12)$$

where T is the total number of web pages retrieved by Google, $f(x)$ and $f(y)$ are the number of hits for search items x and y respectively and $f(x,y)$ defines the number of pages where x and y co-occur. Note that unlike WN similarity which is node based, NGD is a statistical similarity measure.

Links between semantic nodes is weighted using a dissimilarity measure of concepts represented by the nodes containing the link. This dissimilarity measure acts as a cost. A common way to measure similarity of two concepts is by using WordNet Similarity [10]. Thus, the similarity between two nodes v_1 and v_2 is given by:

$$sim\ (v_1, v_2) = Wordnet\_similarity(c_1, c_2) \qquad (13)$$

where, $c_1 = concept(v_1)$ and $c_2 = concept(v_2)$ as seen on WordNet.

When similarity is unavailable in WordNet, we can used Normalized Google Distance (NGD) as the metric, as described in Eq. 12. Calculating semantic similarities for the entire graph generates a matrix $D_{ss} = [VxV]$ of similarity scores. We use reciprocals to convert D_{ss} into a dissimilarity matrix $D_{ds.}$

$$w_{v_1 v_2} = disim(v_1, v_2) = \begin{cases} \frac{1}{sim(v_1,v_2)}, & WordNet_{similarity}(c_1, c_2) > 0 \\ NGD(v_1, v_2), & WordNet_{similarity}(c_1, c_2) = 0 \end{cases} \quad (14)$$

This adds weights to the semantic graph (G), with edge weights representing the dissimilarities between two concepts D_{ds}.

5.2 Categorical Classification using Concept Graph

Finding the correct category of a word depends on the context in which the word was used. As mentioned earlier, social media trend words have different lifetimes and patterns of growth or decay based on their categories. Predicting growth and decay patterns is essential to various applications, including targeting ads, popularity and buzz estimation or user affinity towards certain brands. Since trends within the same category have similar growth-decay patterns, it would be logical to first detect the category of a trend before predicting its other attributes, such as persistence or recurrence.

A fundamental problem with real world data is that the noise associated with its generation can cause classification and categorization challenges. In other words, a particular piece of text cannot be easily contextualized in terms of pre-selected categories. For example, the Twitter trend - *'Justin Bieber come to Spain'*, is partly about a *location* but also about *music*. Categorization is a primary challenge of either topic modeling or intelligent content analysis techniques. The basic task is to separate each data instances into a pre-selected categories. The task is non-trivial especially if the data is generated in real-time and is full of noise.

The existing methods to categorize Twitter trends are ad hoc at the best. Some, like whatthetrend.com is human-curated. However, this severely limits the applicability, since there is lack of enough labeled data. It can also be inconsistent at times, due to disagreement among annotators.

The concept graph can be leveraged cleverly to detect potential categories for data instances. It is implemented in 4 simple steps, namely: (1) Detect semantic concepts related to trend words from a sample set, (2) Label categories based on semantic concepts related to trend words of this sample set, (3) Build training instances where semantic concepts have a category as class, (4) use an ensemble tree classifier [44] to find the probable category of the trend.

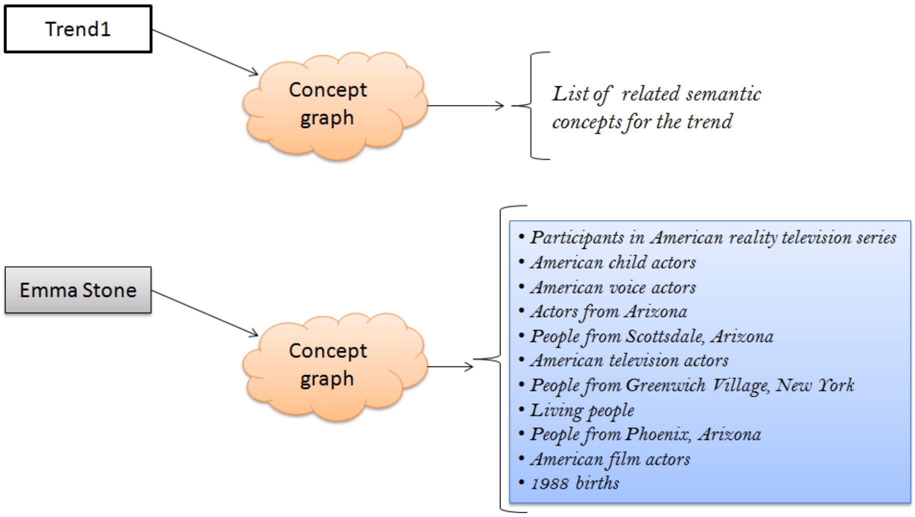

Trend1
Concept graph
List of related semantic concepts for the trend
Emma Stone
Concept graph
• Participants in American reality television series
• American child actors
• American voice actors
• Actors from Arizona
• People from Scottsdale, Arizona
• American television actors
• People from Greenwich Village, New York
• Living people
• People from Phoenix, Arizona
• American film actors
• 1988 births

Figure 25: Detecting semantic concepts of a trend from concept graph.

Step 1 - Detect semantic concepts: When the concept graph is queried with a trend like *'Emma Stone'*, it returns with a list of semantic concepts which is related to the trend. As shown in Fig. 25, the concept graph tells us that Emma Stone was born in the year 1988, has been an American reality television series and a child actor, voice actor and her place of living is Arizona and New York.

Step 2 - Label the category of semantic concepts: Our goal is to label semantic concepts with a category that is most suited for them. For example, as shown in Fig. 26, since Emma Stone is closely related to the category *'Entertainment'*, words like 'actor', 'television', 'film' will be labeled with *'Entertainment'*. Of course, this part of the process needs to be somewhat humanly curated or can be expanded from a seed set of labeled semantic concepts.

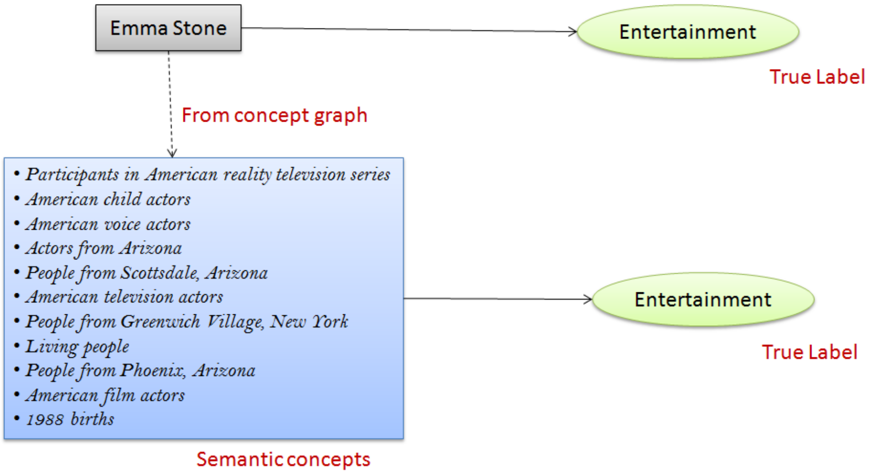


Figure 26: Labeling semantic concepts with true category.

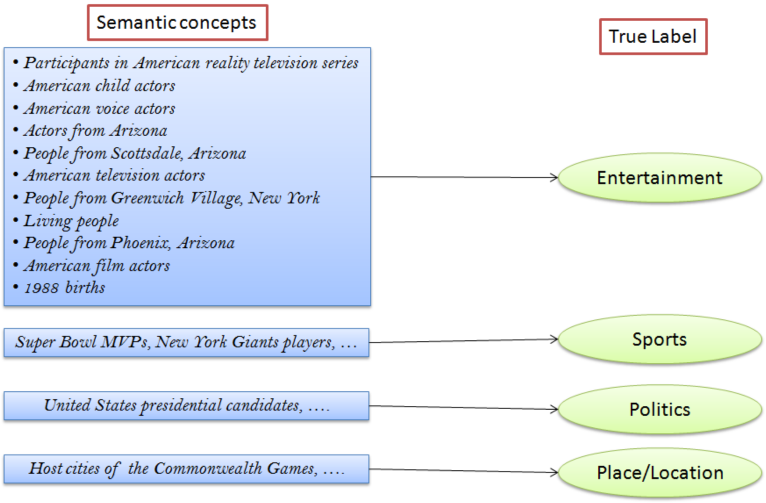

Semantic concepts
True Label
Participants in American reality television series
American child actors
American voice actors
Actors from Arizona
People from Scottsdale, Arizona
American television actors
People from Greenwich Village, New York
Living people
People from Phoenix, Arizona
American film actors
1988 births
Entertainment
Super Bowl MVPs, New York Giants players, …
Sports
United States presidential candidates, ….
Politics
Host cities of the Commonwealth Games, ….
Place/Location

Figure 27: Building training instances

Step 3 - Building training instances: In Step (3), we extend the previous step to take several semantic concepts and label them with categories, as shown in Fig. 27, which serves as our ground truth. We have 18 categories and 2000 semantic concepts as training instances. These concepts serve as training data for a decision tree, which helps in making the decision of which category might be contained within a set of semantic concepts.

Step 4 - Ensemble Tree Classifier: In step (4), we give these semantic concept words and associated true labels to a ensemble decision tree. The purpose of the decision tree is to learn which words have a high probability of belonging to certain category. This ensemble decision tree serves as a classifier where each tree votes on a category and the majority voted category is chosen as the classified category of the trend (Fig. 28).

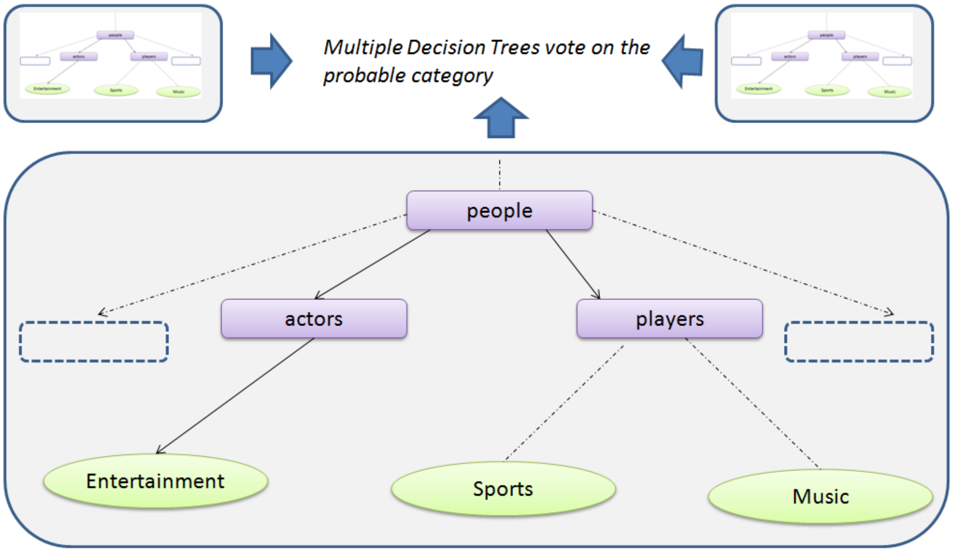


Figure 28: Ensemble Decision Tree classification

Now, we discuss three applications that uses the semNet in order to incorporate cross-domain semantic data into various learning algorithms applied to different domains.

5.3 Applications

The applications we choose are driven by the following notions (1) A collection of words can represent a topic, provided they cluster well in the semNet. (2) Network analysis and time series analysis provide techniques to reveal the evolution of dynamics systems, and (3) predictive tasks can be performed by gaining knowledge of topic and semantics in advance.

The reason to use a network is to understand the relationships among cover issue titles, between issues and topics and amid topics only. These three aspects reveal distinguishing patterns such as what topics are focused on by one of the world's top magazines and how publishing distributes the topics evenly over various issue cover features. For example, time series analysis of topics from the *Time* magazine reveals increasing, decreasing, seasonal and bursty (sudden rise) topic trends over time.

5.3.1 Evolution of Human Socio-Cultural Signals over time

There are many anthropological chronologies regarding the history of topics that has captured attention of the world population over the last century. However, there is limited computational study of relations among these chronological topics that affected humanity and the causal chain of how attention to one topic caused another future topic. In this paper, we explore relative importance and correlations among topics that captured human attention across history using web pages that contain information about the cover page of the *TIME* magazine. *TIME* is an American magazine that enjoys the world's largest circulation for a weekly news magazine. For nearly a century, the cover of *TIME* magazine epitomizes some of the most important topics facing humanity. We use two techniques for our study. First, we employ network analysis to estimate the relations among these topics and their evolution over time. Secondly, we utilize time-series analysis and illustrate the patterns of persistence, decay and correlations among topics that captured world interest over the last hundred years and shaped our current socio-cultural dynamics of existence.

Computational anthropology involves studying the dynamics of social, physical and cultural adaptations in society through computational methods [32]. Although this paper does not purely belong to the genre of anthropological studies, it strives to understand how we can extract signals of socio-cultural adaptations in humanity from the plethora of digital resources on the Internet [31]. We are swimming in a world of digital data. Online archives provide us with new means of investigating patterns governing human societies by intelligent data mining. Such patterns hold information regarding diffusion of topics through the society.

One such online archive is the cover pages of the *TIME* magazine[4]. *TIME* magazine enjoys the world's largest circulation for any weekly news magazine. Although the magazine itself largely appeals to the American audience, the cover of the magazine usually features topics that affect a significant portion of the world population, either directly or indirectly. The cover page of the *TIME* magazine might feature an individual, event or topic (we call these cover features). All of the cover features contain related meta-data, i.e. categories to which they belong to. For example, when the cover feature includes 'Obama', it belongs to *Politics*. Similarly, 'Olympics' belong to *Sports*.

The semantic meta-data associated with each cover page is valuable information (Fig. 29), as it reveals what topics were interesting to the society in a certain week of a certain year. Our methods of analyzing this data are threefold. First, we explore what topics have been most featured (popular) in *TIME* magazine covers over the last century.

[4] The archive is available at http://www.time.com/time/coversearch

Each topic is associated to a cover feature. Each cover feature contains a cover title in words (Fig. 29). Thus, we can build a network of word co-occurrences and perform community detection [33] for the entire dataset, revealing the clustering of similar topics based on the cover feature.

Secondly, since the dataset is essentially a time series, we employ dynamic network analysis [34] to reveal how the key properties of the network (e.g., clustering coefficient, radius etc.) evolve over time. Our final technique does not use co-occurrence. Instead, for a given topic, it produces a signal of the topic's popularity with time based on the frequency of appearance on the cover. With simple signal processing and time series analysis, we can reveal interesting patterns of persistence, decay and correlations among topics.

Issue Date	**Cover Title**	**Cover**	**Topics Meta-data**
April 9, 1928	Henry Ford Sinclair		Finance, Business
April 9, 1956	American Express President Reed		Business
April 9, 1979	Nuclear Nightmare		Nuclear Power, Environment
April 9, 2001	Global Warming		Weather, Global Warming, Environment

Figure 29: Example TIME magazine covers and corresponding topic meta-data from 4 different decades.

Collecting TIME Magazine articles: The archive of all *TIME* magazine cover features since 1923 is available online as archive. We wrote a simple crawler to collect all the web pages containing cover page of issues published. With each associated cover, there is topic meta-data regarding the main theme of the cover feature. For example, if the cover title is 'Global Warming' then the related topic meta-data includes *Weather*, *Environment* etc. For each cover page, we store a triple entry. The triple entry describes three aspects of the cover, namely, its date of publishing, the cover title and the topics meta-data list. Thus, for the issue published on April 9th, 1956 (see Fig. 29), the triple stored is: {*04/09/1956, American Express President Reed,* [*Business*]}. After the entire crawling process, we collected 4,676 such triples, starting from March 23rd, 1923 to May 14th, 2012.

Meta-data and pre-processing: As mentioned earlier, with each issue cover feature, there is available a list of meta-data topics that signifies related topics with respect to the issue title. It is not known to us how *TIME* exactly performed this categorization of cover titles or how they decided on the seed set of categories. However, in most cases, we found the topic categories make sense and relate closely to the cover title. The pre-processing involves handling the usual problems related to text processing, e.g., tokenization of title words, stop word removal etc. In certain cases, we also employ stemming for terms such as "America's" i.e. we use *Politics* instead of 'political'. We do not use any significant natural language processing tasks like parts-of-speech tagging since the meta-data is not in natural language form. The topic meta-data list for an issue is essentially a collection of words that have semantic closeness.

The Network of Topics-Issues: We build a network where each node is either a issue title or a topic. For a given issue title node, all the topics in its meta-data are set as its neighbors in the network. For two issue titles, if they have common words (co-occurrence), then there exists an edge between them in the network. The edge weight between issue title and topic in its meta-data is always set to 1. The edge weight between two issue titles with co-occurring words is the number of common words with some normalization, i.e. if titles A and B have $|w|_A$ and $|w|_B$ words respectively and the number of words in common between A and B is represented as $|w|_{AB}$, then the edge weight between these two issue titles *A* and *B* can be written as:

$$e_{AB} = |w|_{AB} / \max(|w|_A, |w|_B) \qquad (15)$$

Although our network is built using co-occurrence of words in different issue titles, it is fundamentally different from the usual 'co-occurrence network' (Ozgur 2008) where two words found in the same title would have an edge between them in the network. Instead, in our case, if two titles have at least one common word, then they are sure to have an edge connecting them in the network. As mentioned earlier, we pre-process the data to remove stop-words so that an edge reflects reasonable similarities between the two titles. Nevertheless, the same word can reappear in many titles (e.g., the word 'World', 'America' etc.), which would give rise to too many cliques in the network. Therefore, when the same word connects more than 10 titles, we create a super-node (of that common word) and connect all the title nodes to that super-node instead of having edges between each pair of title nodes. A clique of *n* nodes has *n**(*n*-1)/2 edges. The super-node hierarchy reduces this to *n* edges with (*n*+1) nodes. There are 350 such super-nodes in the network.

Properties of Topic-Issues Network: are 4475 nodes in the network and 22,768 edges. The average degree is 10.176 with a density of 0.002, signifying that the graph is pretty sparse. The average path length is 3.61, which is remarkably short. It is a desirable characteristic since shorter average path length indicates better chances of information diffusion and small-world nature [109]. The diameter of the network, defined as the greatest distance between any pairs of vertices, is 7. On the other hand, the radius of the network is 4, which can be thought of as how far a node (title/ topic) is from another node most distant from it in the graph. The degree distribution of the entire network is shown in Fig. 30, which closely resembles a scale-free degree distribution, implying few topics are most often discussed.

The main purpose of building the network is to explore two aspects of the topics, (1) which topics have been most discussed over time, and (2) how have topics and cover features evolved in relation to one another. The first task is handled by centrality measures and community detection in the network. For the second task, we will use longitudinal network analysis.

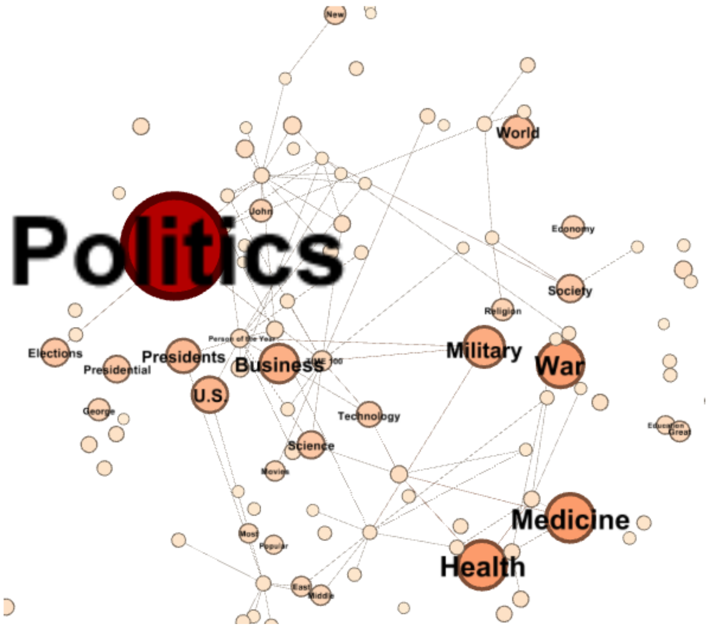


Figure 30: Node sizes represent the magnitude of Eigen Vector centrality of the node in the network. Only 4.14% of the nodes are shown in this figure to effectively illustrate the important topics.

Most popular topics: The centrality of a vertex within the network represents the relative importance of the topic node in the network. We used Eigen vector centrality for our analysis [33]. It measures the influence of a node in a network by assigning relative scores to all nodes based on the concept that connections to high scoring nodes contribute more to the score of the node in question than equal connections to low scoring nodes. By this definition, we expected topic nodes have higher centrality than issue title nodes. The observations (Fig. 30) justify our expectations, as the top 15 nodes with highest centrality were: {*Politics, Health, Medicine, War, Military, Business, U.S., Presidents, World, Elections, Society, Science, Technology, Economy and Religion*}. The higher the centrality, the more important the node in the network. By the same analogy, the higher the centrality, the more number of times the topic appeared in the issue cover. In other words, issue covers in *TIME* strongly relate to the topics : *Politics, Health, Medicine, War, Military and Business* etc.

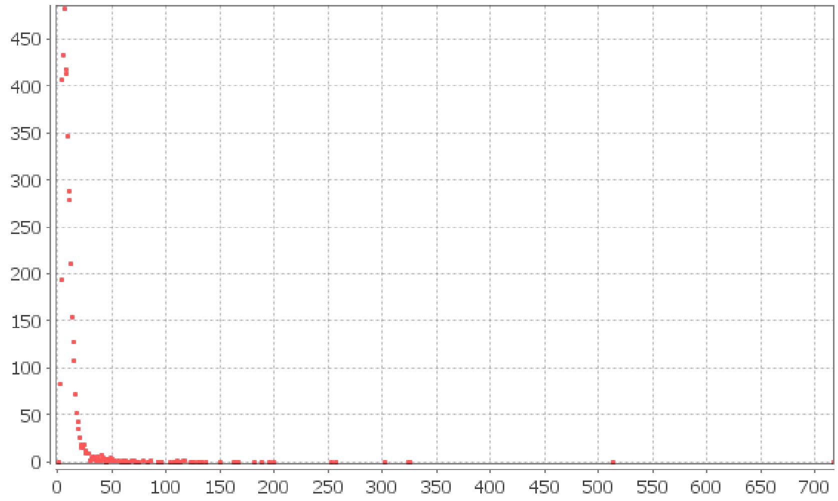


Figure 31: Degree distribution of the entire network

Community formation among topics: In order to understand the relationship among issue title and between issue titles and topics, we need to further detect community structure in the network. A community structure implies the network divides into natural groups of nodes that are densely connected with others in the group/community and rarely with nodes outside the community. Sometimes communities might overlap. Our results revealed that the network possesses 21 different communities. 5 of these communities have a significant more number of nodes than others (Fig. 32). The 5 communities formed around issue titles belonging to the topics of *Politics, War, Health, Business* and *Middle East*. The titles which are part of these communities is visualized in Fig. 32.

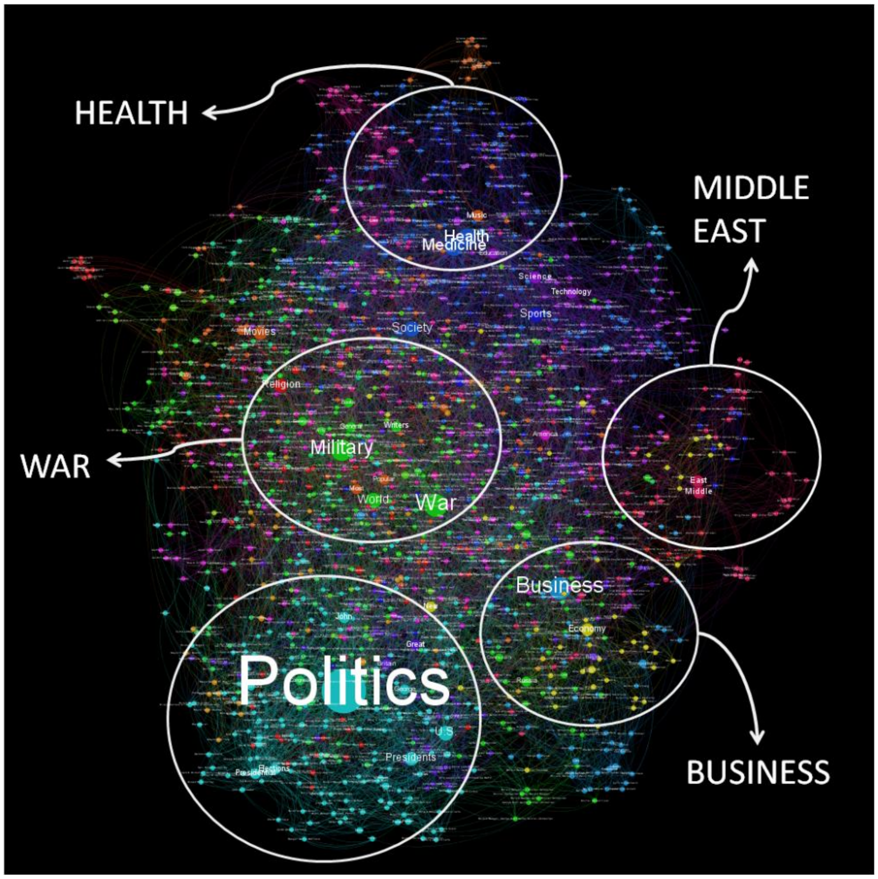


Figure 32: The five major communities in the topic-issues network

Longitudinal Network Analysis: Longitudinal network analysis involves study of changes in the network topology with time. Analyzing networks over time is important to comprehend the decision cycle and causal chain of major events and topics [35]. For example, *terrorism* as a topic was not discussed before 1970s, and as such that node would not be present in the initial network. However, network structure and flow dynamics would be strongly affected when a topic which gains future prominence enters the network. The temporal variation in the network topology is visualized in appendix xx along with the numerical values of changes in network size (Fig. 34) and rate of change in number of nodes and edges for each passing decade (Fig. 33).

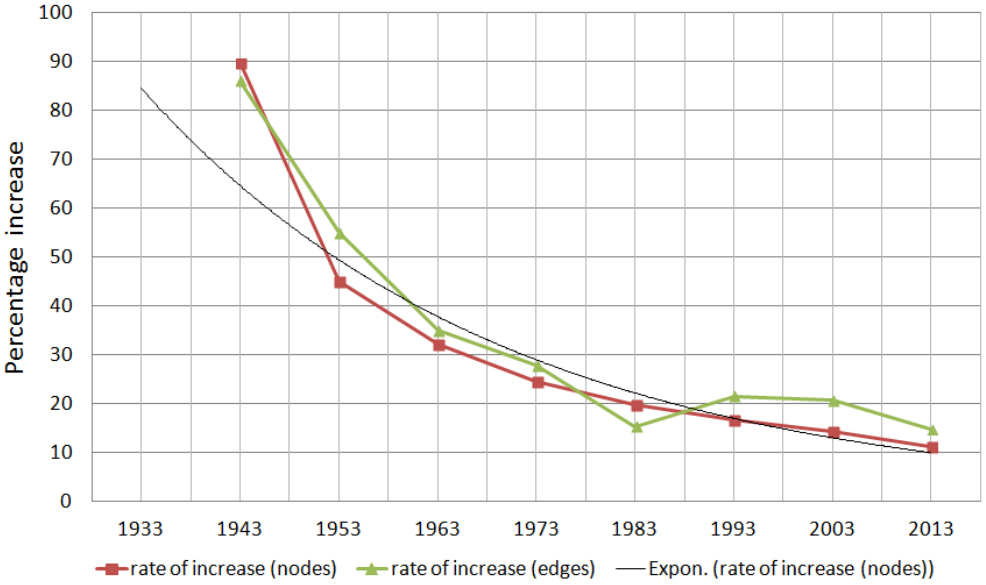


Figure 33: Variation in the number of nodes and edges in the network with each passing decade.

There are two main observations to be made from Fig. 33, 34. (1) The rate of increase in both the number of edges and nodes in the network follows an exponential decay. The decay rate for the first 20 years is approximately $e^{-(0.15x)}$ whereas after 1953, the decay rate stabilizes at around $e^{-(0.05x)}$. This shows the saturating effect on the network growth, i.e. a substantial number of recent issue titles belong to topics that were important even before 1953. (2) Secondly, for one particular decade, 1973-1983, the rate of increase of nodes is more than that of edges. This means during that decade, a number of new issues nodes were introduced in the network that had fewer relations to past topics. In other words, some important topics were first addressed in that decade. Going back to the data, we find that four topics that gained significant future prominence but had almost no discussions before 1970s were introduced in this decade. They were: *Terrorism, Brain, Environment* and *Fraud*.

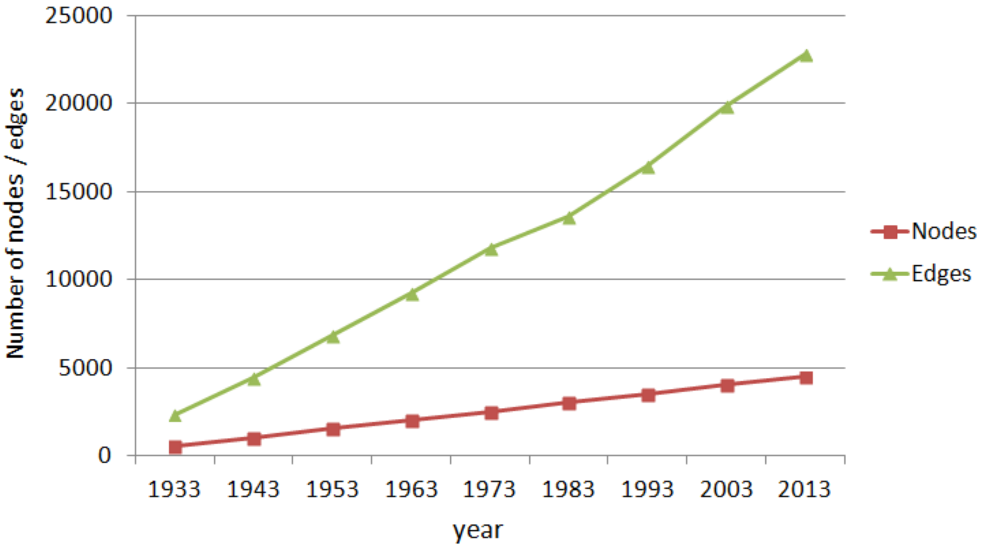


Figure 34: Variation in the rate of increase in the number of nodes and edges in the network with each decade.

We record the variation of the average degree, average path length, average clustering coefficient and number of detected communities over time [35]. In this case, we plot the quantities against the network size for better understanding of the underlying phenomenon. Fig. 35 shows that the average degree of the network remains almost constant (low variance) with increase in network size, implying there is lack of preferential attachment. This also establishes that the topics covered by *TIME* issues over a decade does not bear a normal distribution. Instead, cover features are quite evenly distributed over topics.

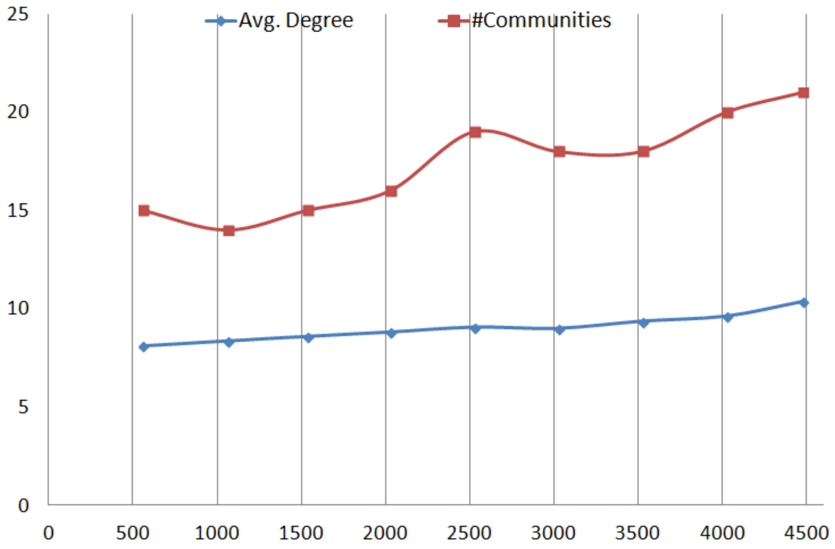


Figure 35: Variation of the Average degree and the no. of communities detected with increase in network size.

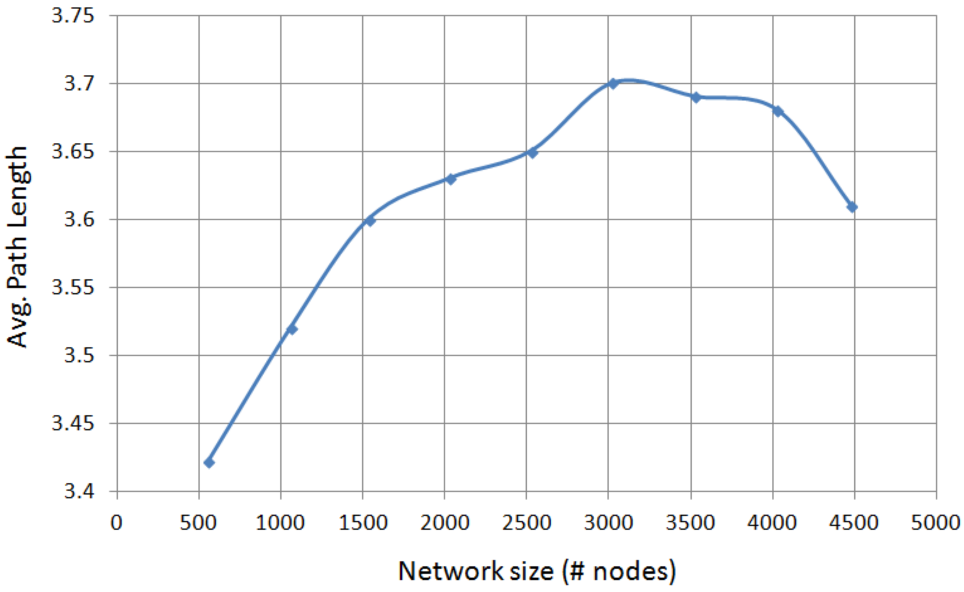


Figure 36: Variation of the average path length with network size.

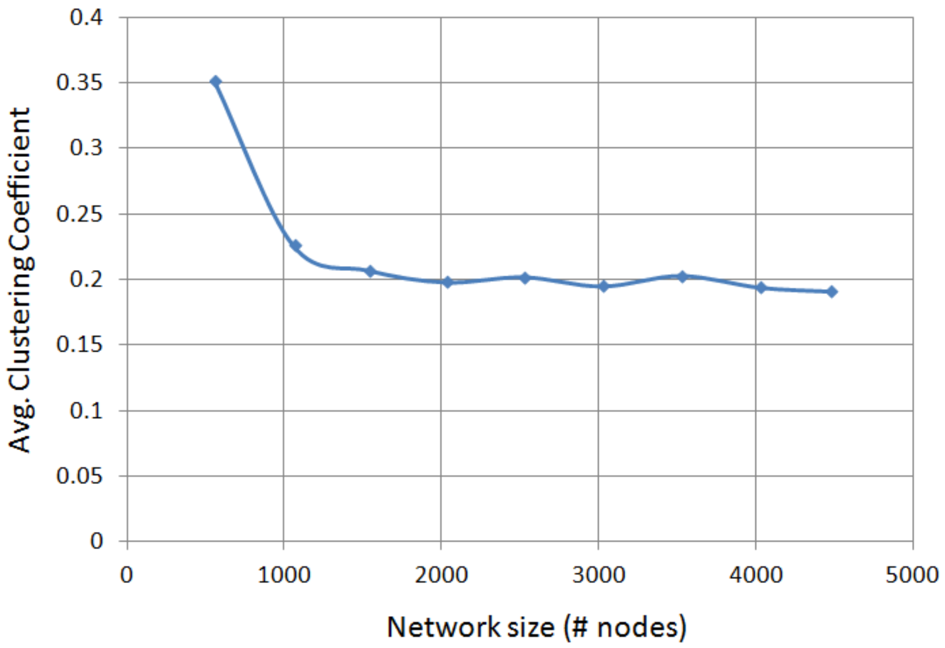


Figure 37: Variation of average clustering co-efficient with network size

Although, the average path length remains around 3.5 for the entire 100 years of network evolution (Fig. 36), it drops after 1993, indicating current issues have strong connection to past topics and not too many new topics have been discovered. The average clustering coefficient is quite low, signifying the network is more random than small world. This is potentially due to the even distribution of topics covered in *TIME* issues over a year.

Topic-Issues Network Evolution: The evolution of the network is a synergy of the evolution of the various topics represented by nodes in the network. In this section, we track the evolution of the topic signals over time and estimate which signals have strong temporal correlation. For each topic word, we measure how many times it appeared in issues over one year as an estimate of the importance of that topic in that specific year. Thus, the frequency of a topic word in the year's published issues represents its relative importance in that year. Since, there are 48 issues per year, the maximum possible value of this frequency can be 48. Empirical results however showed that the maximum frequency achieved was 26, by the topic '*war*'.

In Appendix A2, we show 54 most-occurring topic signals over time. The y-axis represents the frequency of the topic in a year whereas the x-axis represents the years from 1923 to 2004 (values not shown for brevity). There are certain notable observations that can be made from Appendix xx. There are 5 types of topic signals that we observed. There are topic signals that approximately (a) increase gradually, (b) decrease gradually, (c) tend to be seasonal and (d) increase suddenly over time and (e) stay evergreen. Topic signals that seem to have a gradual positive trend over time (getting popular) are *Health, Disease, Research, Middle East, Terrorism, Technology, Computers* etc. Certain other topics have a negative trend over time, (i.e. their popularity is decreasing as we get closer to present day) are *France, Congress, Britain, Military, Transportation, Books, Theatre* etc. Some topics maintain a more or less constant trend over time (evergreen topics), such as *Employment, Industry, Television, Education, Singers, Baseball and Politics.* There are also topic signals that display a seasonal pattern in rise in importance. Some topics that display strong seasonal behavior are *Religion, Republicans, Elections, Broadcasting* etc. Finally, there is also a type of topic signal that did not show a gradual rise in importance, but a very sudden rise. These topic signals include *Iraq, Computers, Vietnam, Environment, Scandals* and *Terrorism*. We believe this last type are motivated by major unforeseen global events.

Correlation among topics: The correlation among topics is demonstrated using a corrgram [36] in Fig. 38. The topics signals used for this analysis are (shown in Fig. 38) - from top to bottom, *Vietnam, Russia, Latin, Diplomacy, Journalism, Military, Labor, Germany, Singers, Industry, Congress, Cars, Britain, France, Transportation, Theater, Baseball, Aviation, Sports, Movies, Books, Business, Finance, Education, Broadcasting,* *Research, Technology, Scandals, Medicine, Computers, Science, Women, Health, Children, Iraq, Employment, Social, Society, Crime, Middle East, Africa, Israel, Environment, America, Presidents, Television, Elections, Economy, Weapons, Space, NASA, Republicans, China,* and *Energy*. In the corrgram, the larger bubble in a cell indicates higher absolute magnitude of correlation. Blue indicates positive correlation whereas red indicates negative.

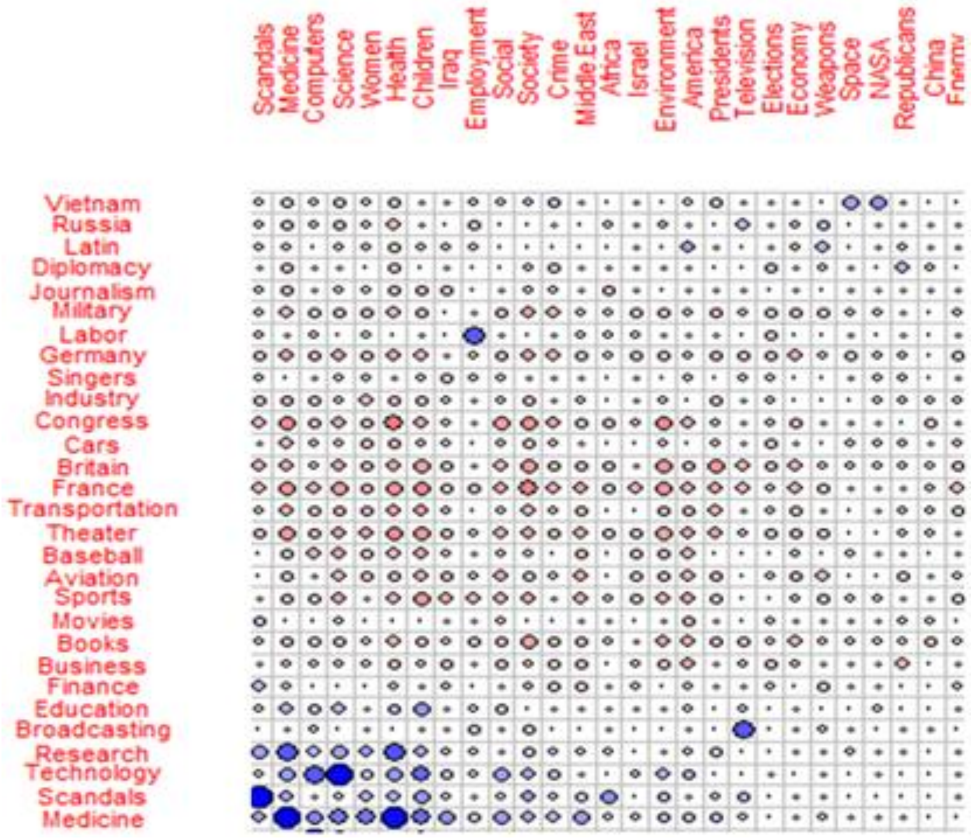


Figure 38: Corrgram of selected topics

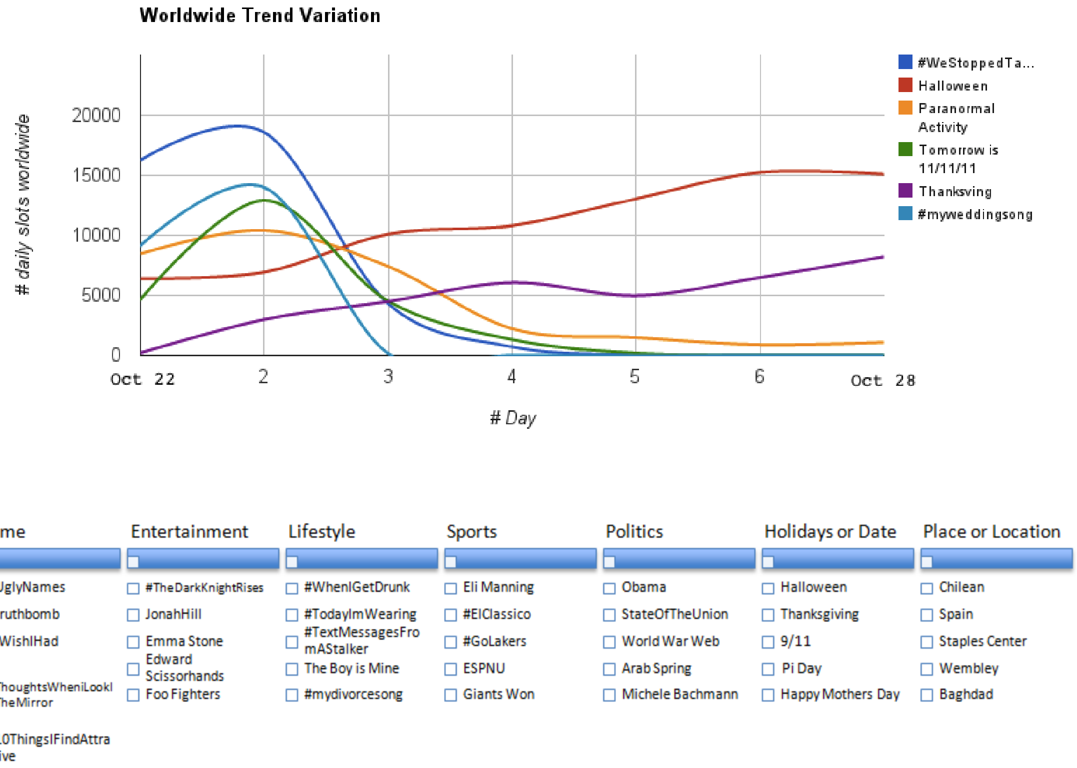
There are some interesting insights revealed from the corrgram. Certain strong positive topic signal correlations are obvious, for example, between *Labor* and *Employment*, between *Germany*, *France* and *Britain* (world war) or between *Cars* and

Industry. Other positive but intuitive correlations are between *television* and *broadcasting*, *books* and *theatre*, *health* and *medicine*, *science* and *technology*, *republicans* and *elections*, *iraq* and *middle east*, *NASA* and *space* etc. However, we noticed other correlations that we less expect. For example, *congress* and *business*, *france* and *books*, *women* and *medicine*, *children* and *scandals*, *employment* and *latin*, *crime* and *society*, *africa* and *scandal*, *israel* and *social*, and *weapons* and *china*. Analyzing the cultural basis of these unintuitive correlations are part of our future work, as is detecting if they were essentially spurious caused by latent agents. Negative correlations were found between *congress* and *medicine, society* and *books* etc.

5.3.2 Predicting spatio-temporal evolution of social media trends

Trends, observed in social network sites like Twitter or Facebook, are the aggregate effects of posts by many users who are spread geographically. These posts arrive in a sequences or batches, giving rise to a unique spatio-temporal trend signal pattern generated by user activity. A trend is a word, a phrase or multi-word posted by a substantial number of users over a small period of time. The top trends make it to the Trending Topic List (TTL) shown in Fig. 72. Chapter 7 describes Twitter trends in more detail. Twitter collects trends based on users from various locations, thus we can say each trend is a 3-tuple (time, location, is it in TTL).

Since trends represent the most popular topics at a given time, it is highly attractive to advertisers, marketers and even to network traffic and scalability researchers to know how the grow and decay. This makes predicting spatio-temporal trends an increasing lucrative field of research. A primary observation I made during my research is that different categories of trends behave differently in space and time. In other words, trends belonging to *gaming* behave significantly different to trends belonging to *music* or *sports*. For example, a *meme* might hold a very high trending score for a small amount of time. On the other hand a trend concerning *holiday* will probably slowly increase in trending score over a long period of time. The growth and decay of different categories of trends are quite different, as shown in the following Fig. 39.



Meme	Entertainment	Lifestyle	Sports	Politics	Holidays or Date	Place or Location
☐ #UglyNames	☐ #TheDarkKnightRises	☐ #WhenIGetDrunk	☐ Eli Manning	☐ Obama	☐ Halloween	☐ Chilean
☐ #truthbomb	☐ JonahHill	☐ #TodayImWearing	☐ #ElClassico	☐ StateOfTheUnion	☐ Thanksgiving	☐ Spain
☐ #IWishIHad	☐ Emma Stone	☐ #TextMessagesFromAStalker	☐ #GoLakers	☐ World War Web	☐ 9/11	☐ Staples Center
☐ #ThoughtsWhenILookInTheMirror	☐ Edward Scissorhands	☐ The Boy is Mine	☐ ESPNU	☐ Arab Spring	☐ Pi Day	☐ Wembley
☐ #10ThingsIFindAttractive	☐ Foo Fighters	☐ #mydivorcesong	☐ Giants Won	☐ Michele Bachmann	☐ Happy Mothers Day	☐ Baghdad

Figure 39: The growth-decay patterns of categorical trends and (below) examples of some trends in different categories.

Not only do trends have different growth patterns in time, they also extend to different ranges of space geographically. This means some trends will extend to a few cities, while others will engulf the entire globe. One of the main questions in connection is being able to predict if a trend will persists for x number of hours, or a trend, that has fallen off the TTL will re-appear in the TTL after y hours. These two characteristics are called persistence and recurrence respectively. Both these properties are in turn affect not only by the category of the trend but also the time and location.

Since this is a panel data where the random variables geo span, persistence and recurrence are affected by multiple dependent factors, like category, location and time of the trend in addition to each, we have to be careful in designing a model that will not over fit or bases on abrupt assumptions. Therefore, we must analyze the each variable separately to begin with.

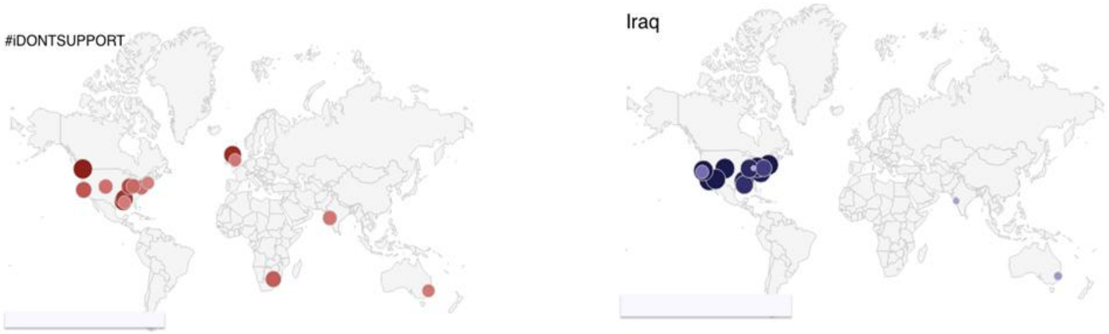


Figure 40: The high volatile geographical locations

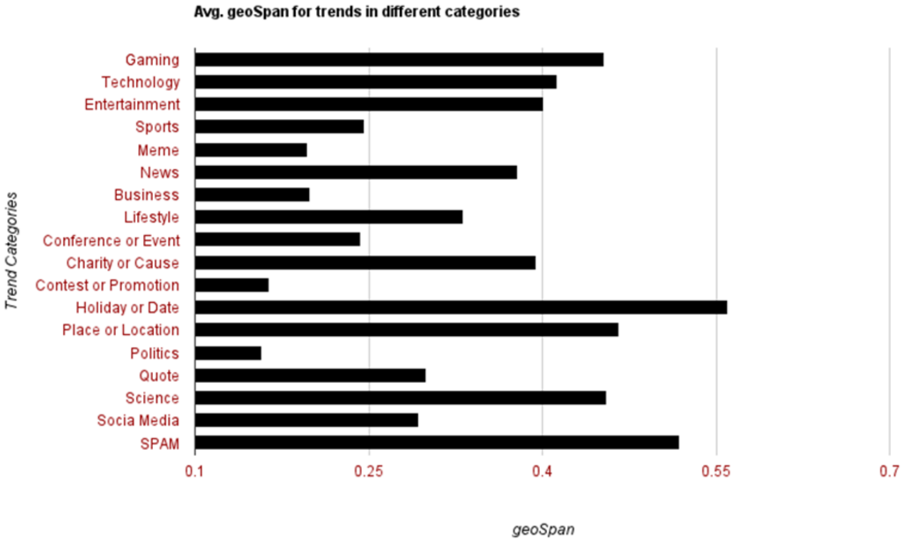


Figure 41: The geo-span of the various trend categories

Persistence: The persistence of a trend is the duration of continuous time units during which a particular trend resides in the TTL. Shown below in Fig. 42. is the trending topic 'Didier Drogba' belonging to category *sports* illustrated as a dispersion chart. A continuous blue line shows the persistence of a trend at some location. A break in the line represents the trend dropping out of the TTL.

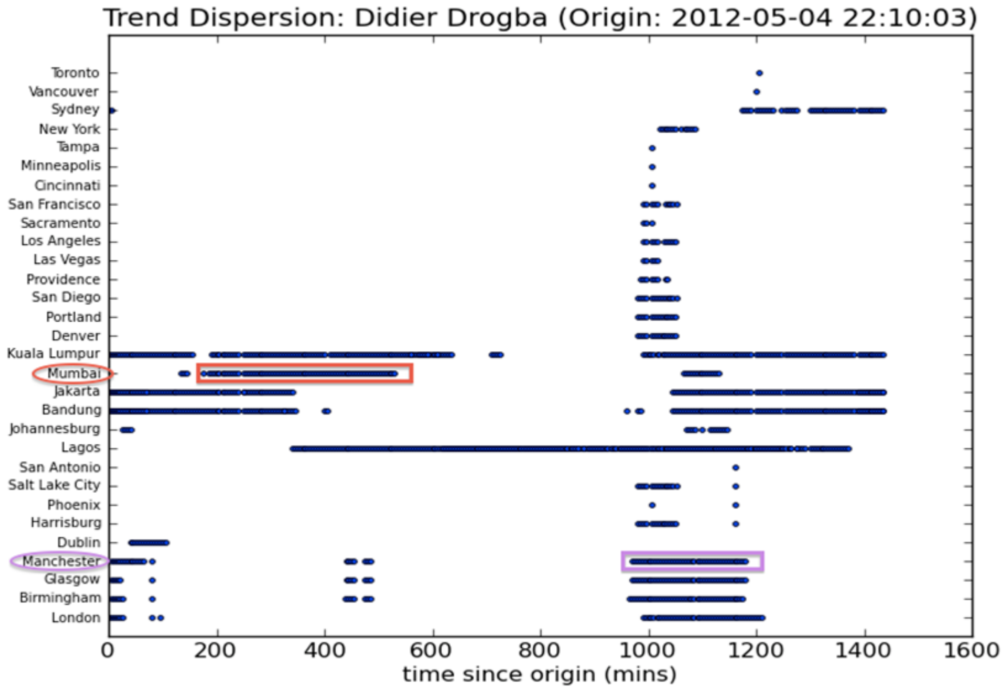


Figure 42: The dispersion chart showing persistence of a trend in 'sports' category

Recurrence: The recurrence of a trend is the number of times it reappears in the TTL after initially dropping out the TTL.

Path Analysis. The dependency among the set of variables (persistence, recurrence, geospan and volatility) is explored through a statistical technique called Path Analysis [96]. Fig. 45 explains the basic idea, where the variables are modeled to be correlated using edges. Edge weights represent the correlation coefficient between the two variables (nodes), also called path coefficients. The expected correlation between two variables that do not share an edge is the product of the path coefficients in the chain connecting them. Equations 16, 17 and 18 represents the standardized regression equations that embodies the path analysis process.

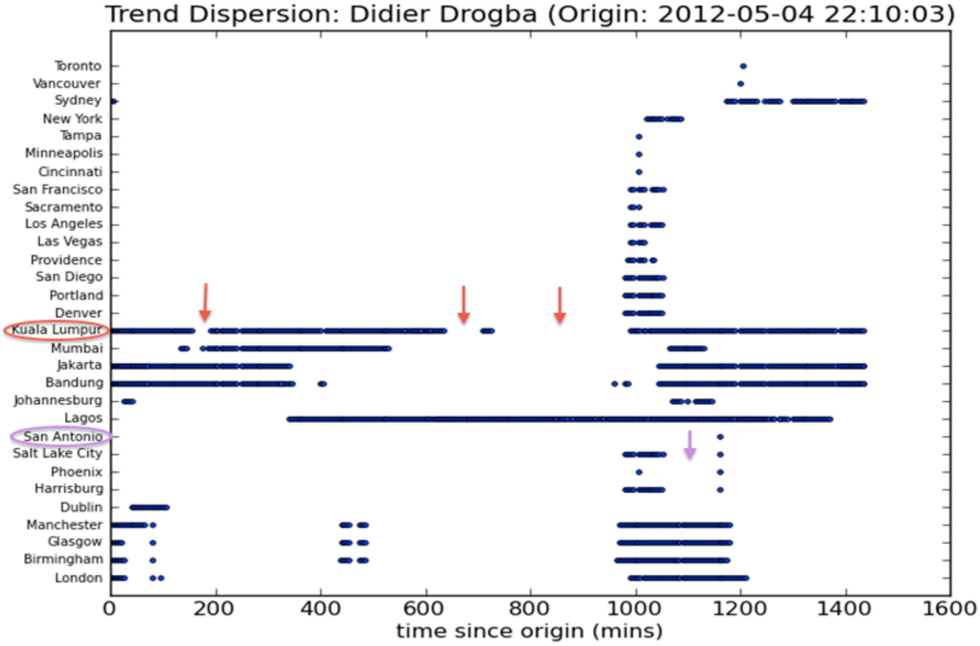


Figure 43: The dispersion chart showing recurrence of a trend in the 'sports' category

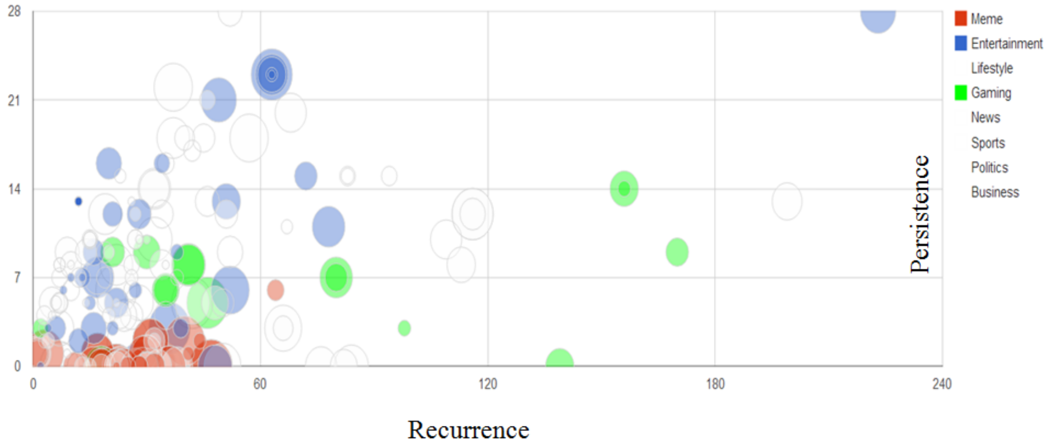


Figure 44: The variation of persistence and recurrence for some categories.

$$geospan = \alpha_{11}.(persistence) + \varepsilon_1 \qquad (16)$$

$$persistence = \alpha_{21}.(volatility) + \alpha_{22}.(\,recurrence) + \varepsilon_2 \qquad (17)$$

$$recurrence = \alpha_{31}.(\,volatility) + \varepsilon_3 \qquad (18)$$

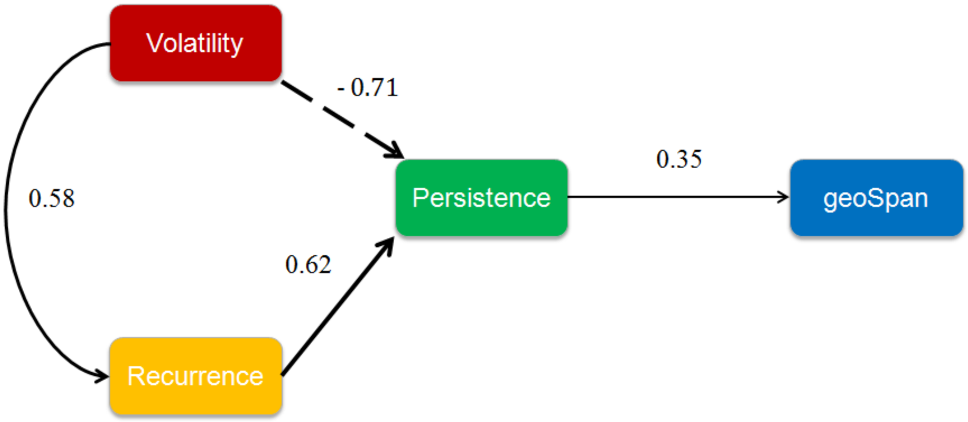


Figure 45: The path analysis model for predicting trend attributes.

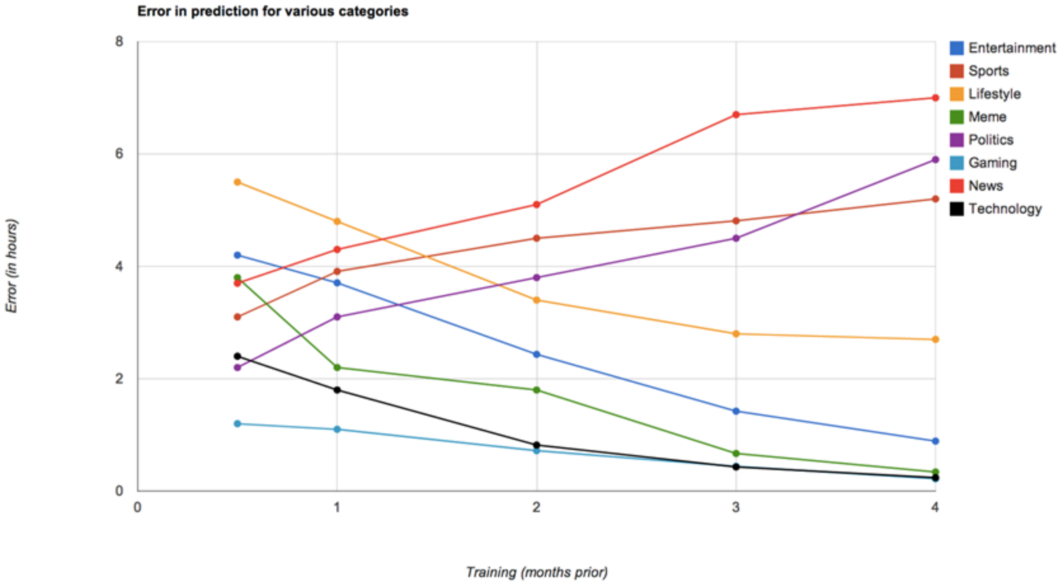


Figure 46: The persistence prediction error (in hours) for trends in certain categories

Fig. 46 illustrates the results of using path analysis for persistence prediction of trends with varying periods of training data. We can observe that for some trend categories (e.g., lifestyle, memes), more training data (looking further into past) reduces the prediction error, whereas for other trends (e.g., sports, politics) looking too far into past data reduces performance. The error here is calculated as (1-accuracy) as explained in Pg. 69. The experiment shows that a path analysis model can predict trend persistence to significant accuracy.

5.3.3 Forecasting movie profitability by using the fine-grained semantic data

The movie genome concept is similar to the Music Genome concept [3], aiming to capture the fine-grained features of multimedia, beyond genre, title etc. A taxonomy created by film professionals including attributes such as mood, tone, story, plot development etc. is being used as movie genome by Jinni [5]. Movie genome has several multimedia applications ranging from movie discovery and semantic search to powering movie recommendation engines [4, 16].

A variety of factors determine whether a movie will grow into a timeless classic or bomb at the box office. Some factors are extrinsic to the real content of the movie, such as the studio creating it or the budget considerations in production. Other factors are intrinsic to the movie content, including the story, plot development, genre, cast etc.

Every movie is composed of a set of intrinsic elements that contain semantic meta-data about the movie. Examples of such elements could range from fine-grained semantics such as mood, plot, audience type, praise, style and whether it is based on a book or not to more traditional classes such as, genre, musical score, flags of violent content, Oscar-winners etc. These set of semantic features for a movie is called its genome [5]. Alternately, each semantic feature (e.g., mood) represents a gene.

There are three interesting questions to be explored from movie genome data - (1) which set of genomes constitute good movies, (2) which set of genomes constitute unpopular movies and (3) is there a way to predict the best set of genomes that will give rise to a successful movie. The questions are of vast importance in the media and entertainment industry, due to the inherent risk involved in selecting scripts and pre-production efforts that is involved prior to a movie begins shooting [11].

Previous attempts at predicting movie success has preferably used traditional box office data such as gross revenue of the movie, advertising budget, number of opening theatres etc. [12]. Other researchers have attempted to tackle the problem using social media signals as indicators of popularity [13]. However, results indicate that prediction is often inconsistent [14]. This paper attempts to answer the above mentioned questions using network science [10] and genetic algorithms [7]. We take a different approach, in the sense, we use a genetic algorithm based on fine-grained semantic meta-data surrounding the movie, represented by its genomes.

Firstly, we attempt to understand which genomes produce positive impact in audiences and which do not. Our approach to studying this problem is by constructing a network, where each node represents a specific value for a gene and edges represents genes elements that have been found in the same movie. This is analogous to word co-occurrence networks used in text mining [15]. Consequently, we try to detect communities of nodes in such networks, which represents the group of genes that had positive impact, given the network was formed out of successful movies. To answer the third question, we use genetic algorithms to find the strongest group of genes that identifies with most success. Our fitness function is comprised of variables chosen from the network topology metrics of the gene co-occurrence network. Thus, the structural properties of the network is embedded in the genetic heuristic, allowing for better convergence due to the natural dependence on network motifs.

We use Internet Movie Database (IMDB) Top 250 movies (http://www.imdb.com/chart/top) as a dataset of 250 most successful movies. This list contain a good mixture of box office hits and Oscar winners. We also use IMDB lowest 100 ranked movies as a set of unsuccessful movies, which received very poor ratings from critics and users. For purposes of evaluating the utility of our genetic algorithm, we exploit an additional test dataset of 675 movies ranging from 2007 to 2011, released by the Motion Picture Association of America (MPAA).

Genetic algorithms is a class of evolutionary algorithms that depend on a search heuristic and mimics biological evolution. The four major steps in any genetic algorithm include inheritance, mutation, selection and crossover [7]. Starting from a random population of candidate solutions, an optimization problem is evolved towards better solutions. The optimization is necessarily a fitness function, which needs to be scaled linearly or exponentially [1]. Every candidate solution is fundamentally a genome, consisting of several genes that can be crossed over, dropped or mutated. Genetic algorithms find wide application in bioinformatics, search, economics and phylogenetics [7].

Our results indicate that there are four key communities of genes that have positive impact and five communities of genes that could have negative impact on audience acceptance of a movie. Moreover, the genetic algorithm we develop improves the accuracy rate of predicting successful movies by 26% over baselines and 31% over traditional classifiers, including a 71% chance of accurately predicting high profitability movies.

Data: We utilize three datasets in this research. Two datasets are crawled from IMDB for building the Movie Gene Co-occurrence network (MGC). One other dataset is obtained from MPAA and contains success ratings of 675 movies released between 2007 -2011. The latter is used to test the performance of our proposed genetic algorithm. The first two datasets are augmented by us, by attaching genomes to each movie. The third dataset of movie success ratings contains profitability, box-office revenue and gross-overall revenue for movies.

Movie Genome: Each movie is represented by its genome as shown[5] in Fig 47. To get the genome of a movie given its title, we use Jinni and Wikipedia. Meta-data from Jinni are structured. When we need to use Wikipedia, words from the plot and other sections of the related wiki page that are essentially web links is extracted as gene elements. We also utilize DBpedia RDF data for the corresponding movie, using similar techniques as mentioned in [16].

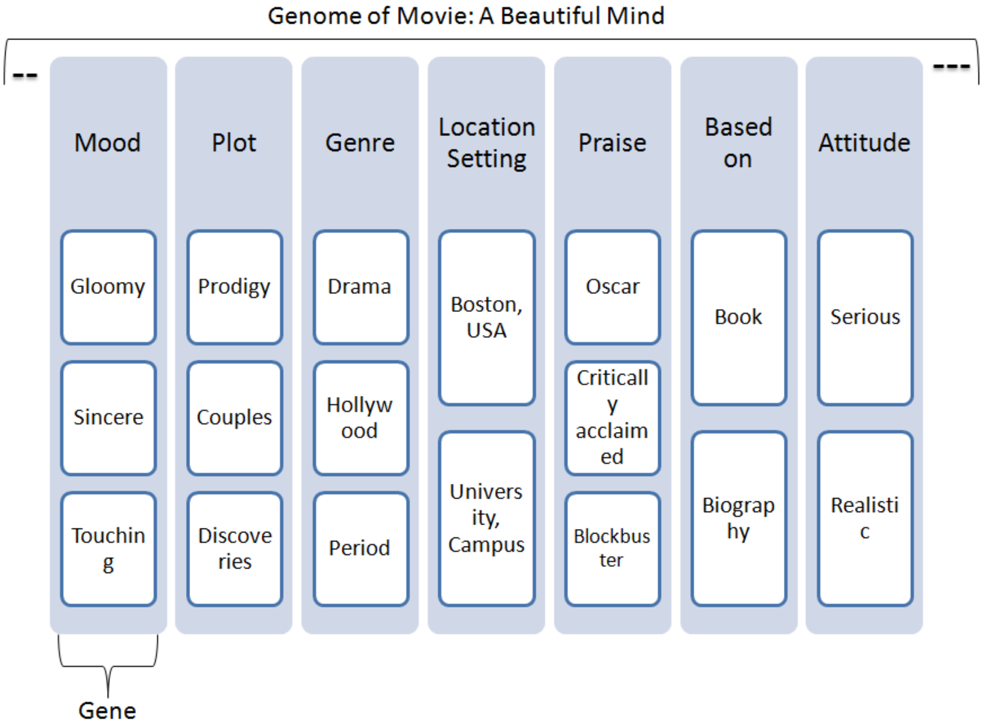


Figure 47: Genome for the movie - 'A Beautiful Mind' (2001).

A movie genome is composed of several genes, each gene indicating a certain feature of the movie. A gene is further composed of several *gene elements that describe the movie gene using fine-grained semantic information*. There are 667 unique gene

[5] Higher resolution images are available at: http://bit.ly/18yZht0

elements in our dataset. At this point, it is important to understand the difference between genome, gene and gene element in our model. As shown in Fig. 47, various combinations of gene elements can give rise to a gene. The set of genes for a movie is called its genome.

Given the movie genome, our goal is to understand whether some genes are stronger than others, in sense the appear more often in successful movies. An elegant way to represent this data is by means of a network of gene elements, as described below and shown in Fig. 48.

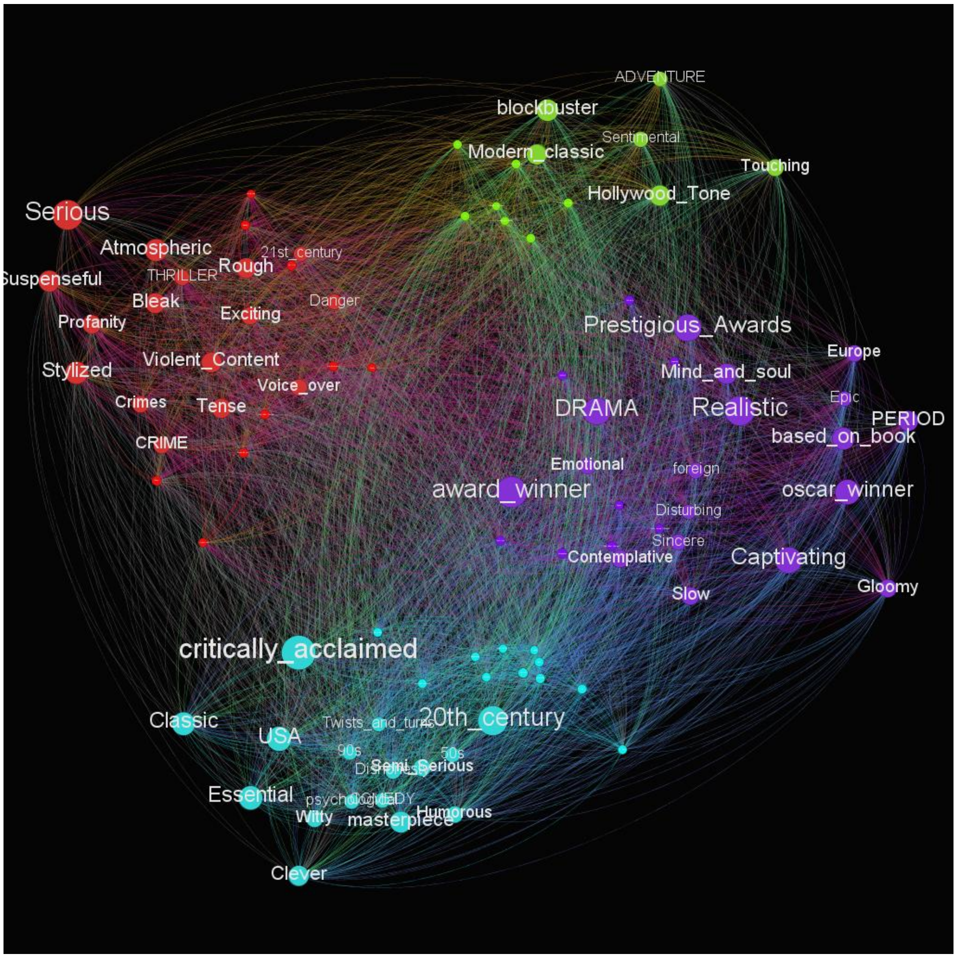


Figure 48: The MGC network of IMDB Top 250 Movies.

MCG Network: Given the genomes for a set of movies, a simple algorithm is implemented to create a MGC network. The elements of a gene is represented by a node in the network. For example, in Fig. 47, 'gloomy', 'sincere', 'drama' etc. are elements of the gene, and thus appear as nodes in the network. The corresponding gene name, i.e. 'mood' or 'plot' is not a node in the network.

Edges indicate two elements that co-occur in the genome. Thus, 'gloomy' and 'serious' will have an edge in the network since they both occur in the movie genome (Fig. 47). Every time the algorithm sees 'gloomy' and 'serious' in the same movie genome, it increases their edge weight in the network by 1. If one gene element in the pair is missing from the network, a node for that gene element is created an a corresponding edge added.

Thus, in MGC network each node represents an element of some movie gene. Edges between two nodes indicate that the two corresponding gene elements co-occurred in some movie genome. Edge weights represent the number of times such a co-occurrence was seen over the entire dataset of movies. This is a standard way of building co-occurrence networks [15].

Genome Communities: For the purpose of detecting communities in the MGC network, we use the adjacency matrix (*A*) representation of MGC network, where $A_{ij} = 1$ represents an edge exists between nodes (elements) *i* and *j*. The modularity Q of MGC of n nodes can then be calculated as described in [2]:

$$Q = \sum_{ij} \left[\frac{A_{ij}}{2m} - \frac{k_i . k_j}{(2m)(2m)} \right] \delta(c_i c_j) \qquad (19)$$

where k_i is the degree of node *i*, $m = \frac{1}{2} . \sum_i k_i$ and c_i represents the community of node *i*. One way to detect the community is to use edge-between-ness repeatedly, as is described in the Girvan-Newman algorithm [6]. The method systematically removes edges of highest betweenness and then recalculates the between-ness of the surviving edges. At some point the network breaks into two or more isolated sub-networks, representing the partitions (or communities).

Two MGC networks are built from the IMDB-top-250 movies and the IMDB lowest 100 movies respectively. Separate community detection is employed on each network. For the top-250 movies, four distinct communities were detected. On the other hand, five communities were detected for the lowest 100 ranked movies. The most influential elements in each community is identified by the node (gene element's) Eigen Vector centrality, which is expressed as:

$$EVC_v = \frac{1}{\gamma} \sum_{t \in MCG} a_{v,t} \, EVC_t \qquad (20)$$

where EVC_i represents the Eigen Vector centrality of vertex i, γ is a constant and $a_{v,t} = 1$ if vertex v is linked to vertex t in MGC (0 otherwise). In Fig. 48 and 49, node/label size indicates the magnitude of Eigen Vector centrality of the gene element within the MGC network.

We take note of gene elements with highest Eigen Vector centralities in each community for the IMDB top-250 and the lowest-100 movies. Our results hints at four sets of combined features that captivates audiences. One type is violent, rough, stylized movies similar to the cult favorite *Pulp Fiction*. Another type is movies that are blockbusters. *Inception* and *Iron Man* fall into this category. Community 3 contains movie gene elements that relate to 'Drama' or 'realistic' movies, like *The Prestige.* Finally, community 4 contains movie gene elements for critically acclaimed movies, like *The Dark Knight* or *We need to talk about Kevin*.

It is interesting to note that the same element may occur in two communities of good and bad movies, but it is the co-occurring elements in that community which provide semantic sense to any element. Thus, 'blockbuster' and 'adventure' may go very well together, but 'horror' and 'family' potentially do not. Note how communities in the low-100 dataset have high influence nodes which fail to make semantic sense together, e.g., 'action' and 'family'.

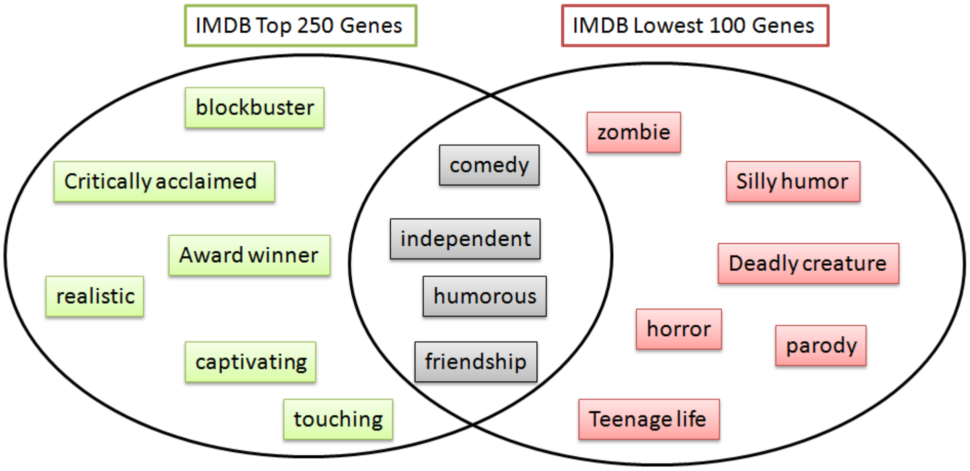


Figure 49: A Venn diagram showing gene elements unique to top-250 movies and lowest-100 movies of the IMDB dataset.

As a statistic, there are 667 unique gene elements present in the IMDB movies. Among these, 333 gene elements are observed in the lowest-100-movies while 418 are present in the top-250 movies. Some gene elements are found in both the top and lowest ranked movies. Fig. 49 shows some gene elements present in each group using a Venn diagram representation.

The reasons for detecting communities will now be made more clear. A community, in sense, is a semantic cluster of gene elements that together contribute to forging good or bad movies. Thus, the influence of a node within the network and its community become key measures in the study of ideal ingredients (genes) for making a movie successful.

There are two key measures that we can calculate from the network in order to use as part of the fitness function of the proposed genetic algorithm: (1) The *influence of the gene element* within the MGC network, and (2) the *tendency of the gene element to cluster with other gene elements* in the MGC network. The first characteristic is exhibited by the Eigen vector centrality of the gene element within the network. The second property can be measured using a local clustering co-efficient of the node within its network community. Note that the first measure is with respect to the global network, while the second property is confined to the community in which the gene element finds itself. This balance is important, since a group of medium influential nodes might possess strong local clustering.

Evolution of Movie Genomes: The problem we are trying to solve is searching for strongest cluster of genes that will produce a successful movie. The solution to the problem lies in the evolution of movie genomes using a genetic algorithm (GA). The main idea is to select an initial population of genes, develop a fitness function and then continuously update the gene combinations until the fitness function no longer improves or the population remains constant. Due to lack of space, we avoid discussing the entire detail of how GA's work in general; please take a look at [1]. The GA for movies that we developed is described in Algorithm 3.

As shown in Fig. 50, each gene can be pictured as the strip of blue (indicating element present) and red (element missing). A crossover occurs when two sections of two genes are interchanged. A mutation refers to flipping one gene element (turns red to blue and vice versa). New genes are produced as a result of crossover and mutation. The next task then, is to measure which of these genes to select for the future population. This depends on the quality of the gene.

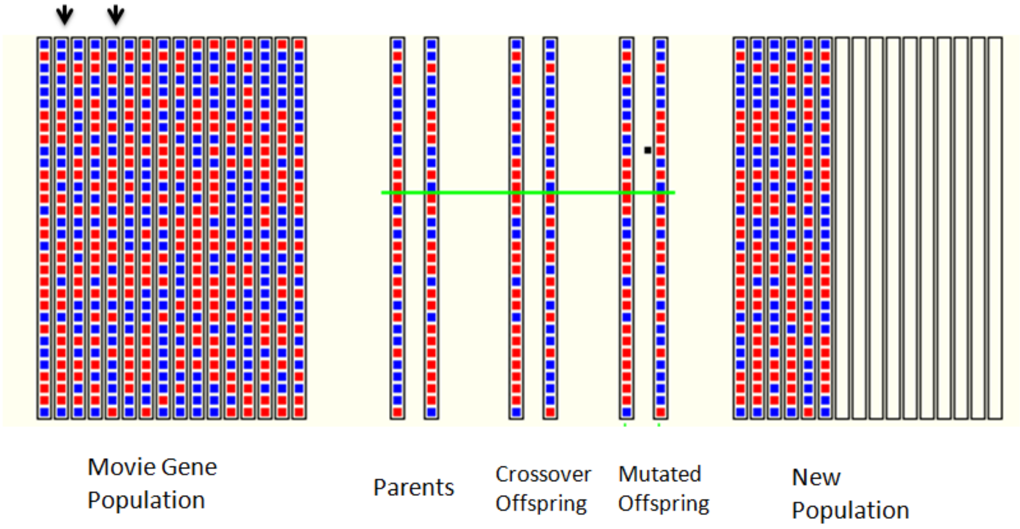


Figure 50: Genetic Evolution of movie genes.

The quality of a gene (x) is represented by its fitness. The fitness function depends on the location of each gene element in the MGC network of IMDB-top-250 movies (MGC^{t250}) and the MGC network for lowest-100 movies (MGC^{l100}), as well as the communities within which they exist in the networks and their clustering coefficients. The clustering co-efficient for a graph s, is a measure of the tendency of the vertices of s to cluster together. As described in [2], it can be defined as:

$$\delta_s = \frac{3 * (no.of\ triangles\ in\ s)}{(no.of\ connected\ triples\ in\ s)} \tag{21}$$

where, a triangle refers to a sub graph of s with 3 vertices and 3 edges, whereas triples refer to a sub graph of s with 2 edges and 3 vertices. Thus, a triangle is a closed triple. The clustering co-efficient is the ratio of the number of closed triples (i.e. 3*triangles) over the total number of triples (both open and closed).

The fitness function we use for our evaluation is:

$$fitness_x = 2 \sum_{i \in MCG^{t250}} \alpha_i - \sum_{j \in MCG^{l100}} \alpha_j + \frac{1}{k} \sum_{k=1}^{C_k} \delta_{C_k} \qquad (22)$$

where $i, j \in x$ are gene elements of gene x, α_i is the eigen vector centrality of the node element i (Eq. 20), k represents the number of different MGC^{t250} communities in which the elements of *x* lie, $1 \leq k \leq 4$, C_k is the subset of *x* that lies in community k of MGC^{t250} and δ_s is the clustering co-efficient of nodes in set *s*.

The right hand side of Eq. (22) is not difficult to interpret. *The stronger the influence of the node (element) in MGC^{t250}, the greater is its chances of being chosen for evolution. The stronger the influence of the node (element) in MGC^{l100}, the lesser is its chances of being chosen for evolution.* Thus, the first term represents the number of *hits* (influence) by good nodes from top-250 gene elements, the second term represents the chances of *deaths* (influence of the element in the low-100 network). Finally, the last term represents the average clustering co-efficient of the elements within their communities in MCG^{t250} network indicating their *survival rate* (strong clustering means higher chance of survival). The stronger the clustering, the more difficult is to find the gene element isolated, without its triangle-d elements.

Algorithm 3: *Genetic Algorithm for Movie Genomes*

```
1. Produce an initial population of genes X randomly selected from the movie gene
   database. Set $\varphi = 10$.
2. Evaluate the fitness of each gene ($fitness_x$) $\forall x \in X$ using Eq. 22.
3. While $\varphi > 0$:
4.         Filter/ select genes with a probability $p_f$.
5.         Cross-over genes with cross-over probability $p_c$.
6.         Mutate genes with mutation probability $p_m$.
7.         Re-evaluate fitness and generate new population.
8.         If new population is same as old population:
9.             $\varphi = \varphi - 1$
```

φ, the cycles of evolution, is initially set to 10. It is decreased every time the population becomes stable until it reaches 0, which serves as the termination condition for Algorithm 3.

Three key parameters that need to be set are selection probability p_f, the crossover probability p_c and the mutation probability p_m. They are defined as follows. The selection probability is decided using the popular roulette wheel method of proportionate selection:

$$p_f = \frac{fitness_x}{\sum_{x \in X} fitness_x} \tag{23}$$

We set $p_c = 0.72$ and $p_m = 0.03$. The choice for these values for the parameters will be made evident from results obtained(shown later). Of course, these parameters can be made dynamic as well, depending on the topology of gene elements in MGC network. However, the dynamic setting of genetic parameters is left for future work.

Evaluation: For testing, we use the third dataset containing movie profitability scores released by MPAA after the 2011 Academy Awards. Since this was released by MPAA, we consider it to be authentic. It has a total of 675 movies from year 2007 to 2011 and serves as our ground truth. The attributes in this MPAA dataset include success measure for a movie, in terms of its average revenue earned on the opening weekend, total domestic gross earned by the movie, total foreign gross and the overall worldwide gross revenue. All these indicate towards the profitability of the movie. We divided the movies into 3 major categories based on the Profitability, which ranges from 10% to

766%. Movies were labeled 'Low' category when their profitability lied 65.7- 298.6%, Medium for 298.7- 532.4% and High profitability for 532.3- 766%.

The population size is chosen at 220 genes. The initial set thus consists of 220 randomly selected genes. After each round of evolution, we select the top selection probabilities according to p_f and retain the population size. The task of GA is to search for the best possible set of genes that determine movie success. This set is produced over several rounds of evolution (in the order of 1000s). Result is the set 'success genes'. Thus, for each test movie, we calculate the fraction of genes of the movie that were 'success genes', and define the potential success as:

$$Potential\ Success\ = \frac{|Movie\ Genes| \cap |Success\ Genes|}{|Movie\ Genes|} \qquad (24)$$

Thus, given the 'success gene' sequence generated by the GA and the test movie genome, we can calculate the potential success of the movie. After calculating the potential success of the movies in the dataset, we classify a top 65% quantile score into high class, the 30-65% quantile into medium and the lower 30% quantile is classified as low class. Note that Eq. 6 is very similar to the 'precision' statistic of information retrieval. In other words, we consider the successful movie prediction problem as a high-precision search scenario.

As a benchmark to the proposed approach of network communities-based genetic algorithm (NCGA), we employ a naive genetic algorithm (NGA) and a decision tree (DT) classifier on the test dataset. NGA does not utilize any properties of the network structure in its genetic algorithm, instead its fitness function is defined as:

$$fitness_x = \sum_{i \in x} [t(i) - \ l(i)] \qquad (25)$$

where $t(i)$ represents the frequency of occurrence of element i among movies of IMDB-top-250 dataset and $l(i)$ is the frequency of occurrence of element i in IMDB-low-100 dataset. For example, when i='*critically acclaimed*' then $t(i) = 192$ whereas $l(i) = 0$. Similarly, when i = '*comedy*' then $t(i) = 32$ whereas $l(i) = 24$. Eq. (7) serves as the fitness function for the Naive GA method (NGA).

Table 7 shows the F-scores obtained for the MPAA test dataset. The results are averaged over 10-fold cross validation. From the results in Table 7, we can observe that NCGA outperforms the other benchmarks comprehensively. Note that DT performs well for high class, but is outperformed overall by NGA. This is caused due to some skew-ness in the data, where the number of movies in high classes are lesser than medium or low class. The DT can easily determine the features that establish high successful movies but not so easily for movies with medium or low success.

For all the methods, usually the F-score for high > medium > low, except for NGA where F-score for medium class takes a dip. This is potentially due to the fact that for the fitness function of NGA, the fitness score reflects well the case of very bad genes and very good genes, but genes with a mixture of good and bad elements are given a balanced score. This adversely affects the evolution, which creates uncertainty and results in lowering the recall for 'medium' classes.

Different tests were conducted to check how the efficiency of NCGA varied as we changed the parameter values in Algorithm 1. These parameters are called genetic operators. There are two main criteria to determine the efficiency of a GA. One criteria is the reliability, which is the fraction of correct classifications. The F-score in Table 7 is obtained using the number of correct classifications. The second criteria is the number of fitness evaluations (or populations) required to evolve - until a stable population is reached. According to [8], a simple equation can be used to judge efficiency of a GA:

$$efficiency = reliability - w.\left(\frac{\#\ fitness\ evaluations}{N}\right)$$

We use $w = 3$ and $N=100,000$ for our tests. Let us now discuss parameter setting for each of the following genetic operators.

Mutation: The mutation probability p_m measures the likeness that random elements in the gene are flipped/changed in order to introduce some diversity into the next generation. The flip occurs when a gene element was present in the original population (indicated by 1), but the mutation causes the child to not possess that particular gene element (set to 0). A p_m of .05 means 5 out of a 1000 gene elements picked at random will be flipped. Empirical results shown in Fig. 51 indicate that $p_m = 0.03$ produces best efficiency.

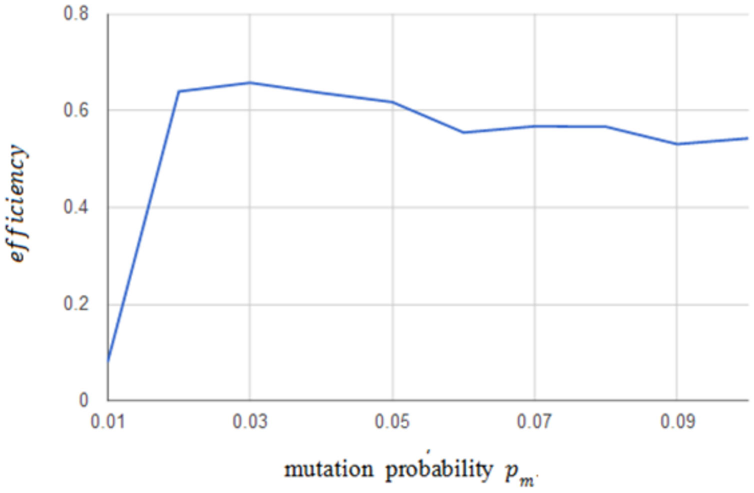


Figure 51: Variation of efficiency with mutation.

CrossOver: Crossover probability refers to interchanging two sections of the gene. Again, empirical results illustrate that the efficiency is maximized when p_c nears 0.72.

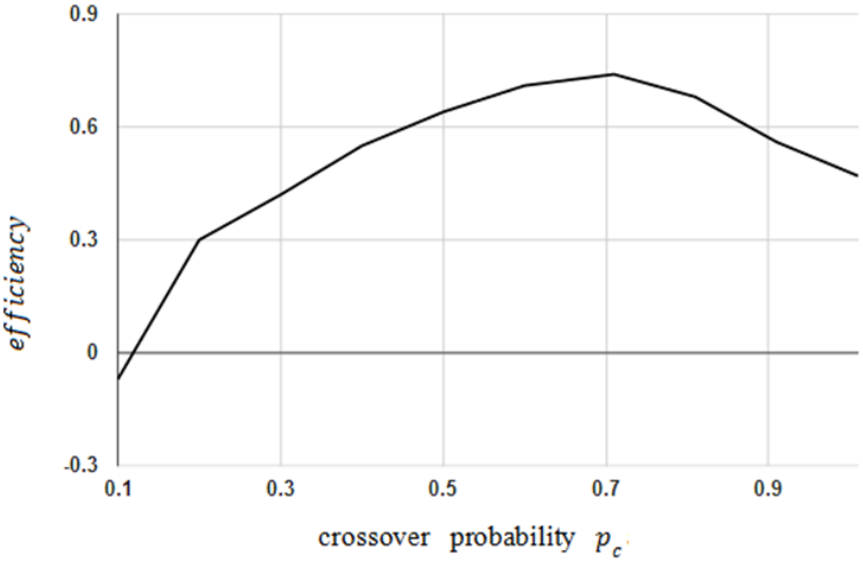


Figure 52: Variation of efficiency with crossover probability.

Population Size: Another parameter that requires setting in a GA scenario is the choice of how many genes to use in a population. Shown in Fig. 53 is the variation of efficiency with the population size. As the results indicate, the efficiency begins to drop beyond a population size of 220 genes. Thus, we choose 220 genes as our population size in Algorithm 3.

Selection Criteria: The previously mentioned Eq. 22 follows the famous Roulette wheel fitness proportionate selection routine, where fitter genes are less likely to be eliminated. An alternative option is to not use fitness in the selection process at all. Instead, a selection scheme called tournament selection involves holding several 'tournaments' among a random selection of genes from the population. The winner of each tournament is chosen for the next evolution. Thus, whereas, roulette wheel selection depends on an individual's relative fitness, tournament selection depends on an individual's rank and is not affected by the fitness distribution [9].

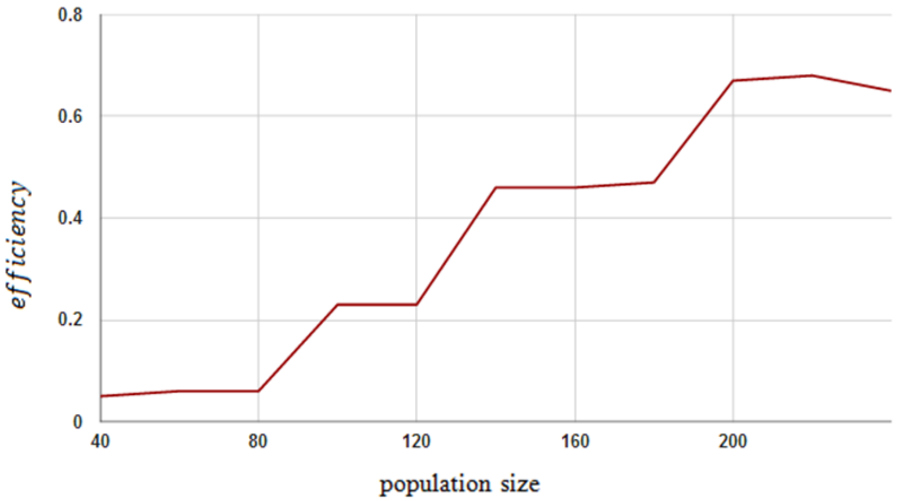


Figure 53: Variation of efficiency with population size.

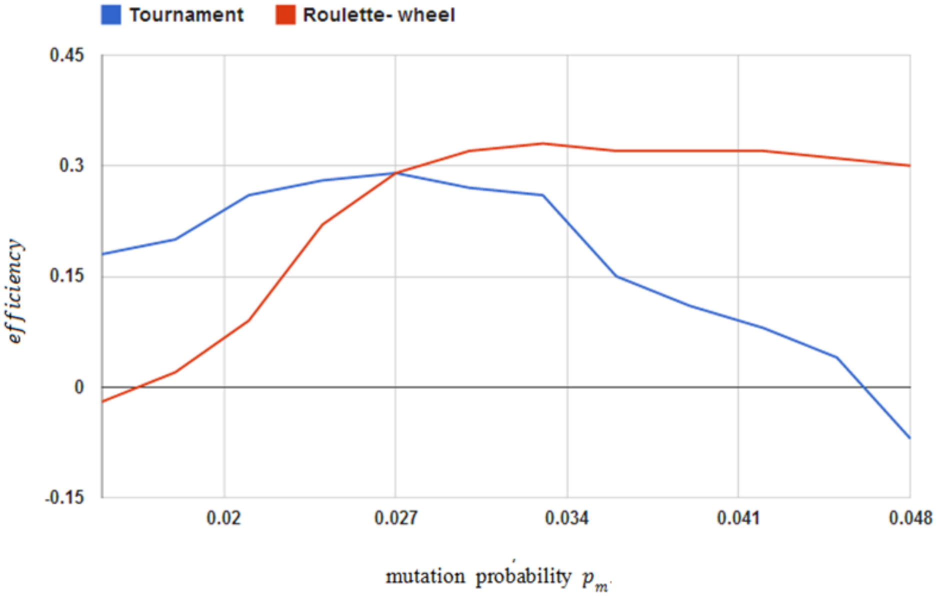


Figure 54: Variation of efficiency with mutation probability.

In Fig. 54, the efficiency is plotted against the mutation probability for both the tournament selection and the roulette-wheel (RW) selection scheme. It seems that initially, the efficiency increases for both the tournament scheme and RW. However, around $p_m = 0.028$, RW starts performing better. This test was run with a tournament size of 10. The bigger the size of the tournament, the more it will behave like RW in selection.

Method	Low	Medium	High	Overall
DT	0.381	0.404	0.536	0.44
NGA	0.496	0.44	0.503	0.481
NCGA	*0.575*	*0.642*	*0.71*	*0.642*

Table 7: Experimental F-Score results. Movie classes are low, medium, high.

5.4 Semantic Coherence in Topics

For the purposes of this section, we will use topics extracted from social media. Social topic modeling aims to extract topics from a stream of social data. In our experiments, we use 10.2 million Twitter tweets as the social data. The model represents words in tweets as a mixture of Z topics which are multinomials over a vocabulary of size V. The probability that word w belongs to a topic $z \in Z$ is represented as $p(w|z)$ and the probability of a tweet $d \in D$ originating from a topic z is $p(z|d)$, where D is the chunk set of tweets. Both the multinomial parameters for topics-given-tweet and words-given-topic are drawn from Dirichlet priors with parameters α and β respectively.

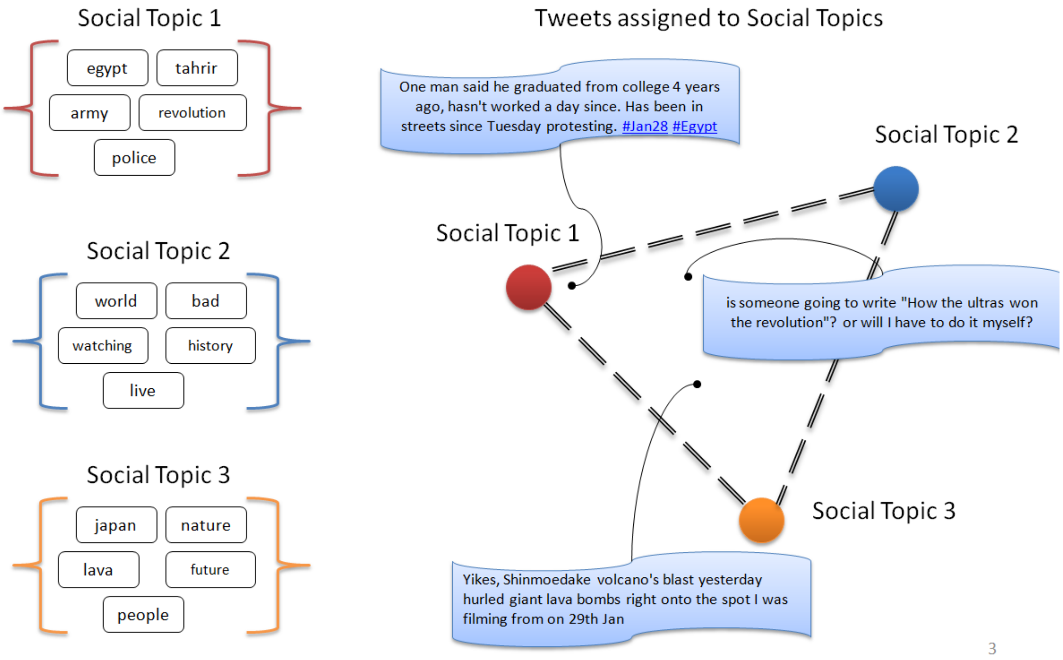
The task of topic modeling is to use the training data in order to populate two matrices; a $V \times Z$ topic-word matrix and a $Z \times D$ tweets-topic matrix. The matrices are learned using collapsed Gibbs sampling (alternatively Variational Bayes' can be used) [1], which iteratively samples the topic assignment z to every word in every tweet, using the update:

$$p(z_{id} = t | x_{id} = w, Z - \{w\}) \propto \frac{N_{wt} - \{w\} + \beta}{\sum_w N_{wt} - \{w\} + V\beta} \cdot \frac{N_{td} - \{w\} + \alpha}{\sum_t N_{td} - \{w\} + Z\alpha} \qquad (26)$$

where, $z_{id} = t$ assigns the *i-th* word in tweet *d* to the topic *t*, $x_{id} = w$ represents the current observed word i.e. *w*, and N_{wt} represents the integer count arrays and α and β are Dirchilet priors. Then the maximum a posterior (MAP) estimate of the topics $p(w|z), z \in Z$ is given by:

$$p(w|z) = \frac{N_{wt}+\beta}{\sum_w N_{wt}+V\beta} \qquad (27)$$

Similar to previous work [6], we use the top-*n* topical words to represent the topic (we choose *n*=20 in our experiments).



One man said he graduated from college 4 years ago, hasn't worked a day since. Has been in streets since Tuesday protesting. #Jan28 #Egypt

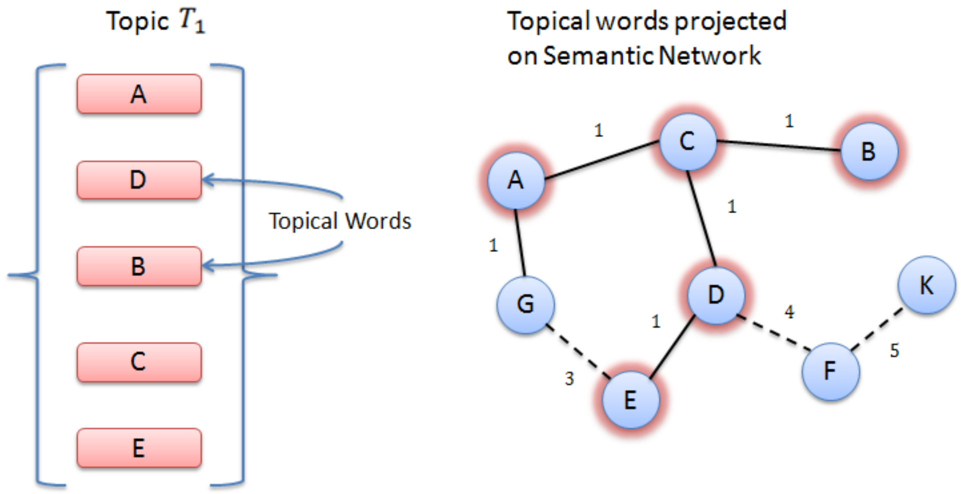
is someone going to write "How the ultras won the revolution"? or will I have to do it myself?

Yikes, Shinmoedake volcano's blast yesterday hurled giant lava bombs right onto the spot I was filming from on 29th Jan

Figure 55: A visual depiction of topic modeling.

Fig. 55 shows an example of topic space extracted from Twitter data by applying Online LDA [103] for 50 topics and 100 rounds of iteration with Dirchilet priors set to 0.5. Social topics (on the left in Fig. 55) are basically clusters of words, where each word has some membership score towards the cluster, defined in the $V \times Z$ topic-word matrix. Each tweet also has some membership score towards a topic (see position in triangle on the right in Fig. 55) defined in the $Z \times D$ tweets-topic matrix. In Fig. 55, we show only 3 topics and 3 tweets. Thus, ‘topic’ is an abstract name given to these cluster of topical words.

To calculate the semantic quality of a social topic, we first need to project all the topical words onto semNet. The idea is shown in Fig. 56 and 57 using a small example graph and topic. Projecting topical words on semNet requires locating the topical words as concept nodes within the semantic network, which can be done in *O(1)* time for each projection due to dictionary data structure storage. Projection is requisite before we can calculate the centrality of the topical words comprising the social topic based on their structural occurrence (or motif) in the semNet. The notion of projecting topical words onto semNet is also illustrated in Fig. 56, which shows projection results from a real life dataset of three topical words on a section of semNet that comprises of ~ 22,100 concept nodes and ~ 42, 329 edges. We shall call the nodes in semNet which are obtained from topical word projection as projected topical nodes.



$$STC(A) = .001 + (0.5)^{2-1} + (0.5)^{1-1} + (0.5)^{2-1} + (0.5)^{3-1} = 2.251$$

$$STC(D) = .001 + (0.5)^{2-1} + (0.5)^{2-1} + (0.5)^{1-1} + (0.5)^{1-1} = 3.001$$

$$STC(B) = .001 + (0.5)^{2-1} + (0.5)^{1-1} + (0.5)^{2-1} + (0.5)^{3-1} = 2.251$$

Figure 56: Analyzing semantic coherence of topical words (strong coherence).

For example in Fig 56. , the topical words of topic T1 are projected on the semantic network. Solid edges refer to direct edges (weight=1) between concept nodes, while dotted lines refer to multiple/indirect edges (weight>1) between concept nodes. Semantic topical centrality (STC) calculation (using Algorithm 2) for each of the top 3 words in T1 with respect to the top-5 topical words is shown in this figure. Alpha = 0.5.

Traditionally, centrality determines the importance of a node v in the network *G(V, E)* based on other nodes of the network. When the centrality for a node v is dependent only on the neighbors of the v, it is called degree centrality [8]. Other variations, notably the Katz centrality, uses every node that is connected (has a path) to v to measure the centrality of v [8]. The purpose of projecting all the topical words onto semNet is to enable us to calculate the semantic topical centrality (STC) of a projected topical node v using only the linkage of v to other topical words (i.e. other projected topical nodes). Thus, STC can be treated as a variant of Katz centrality for topical words only.

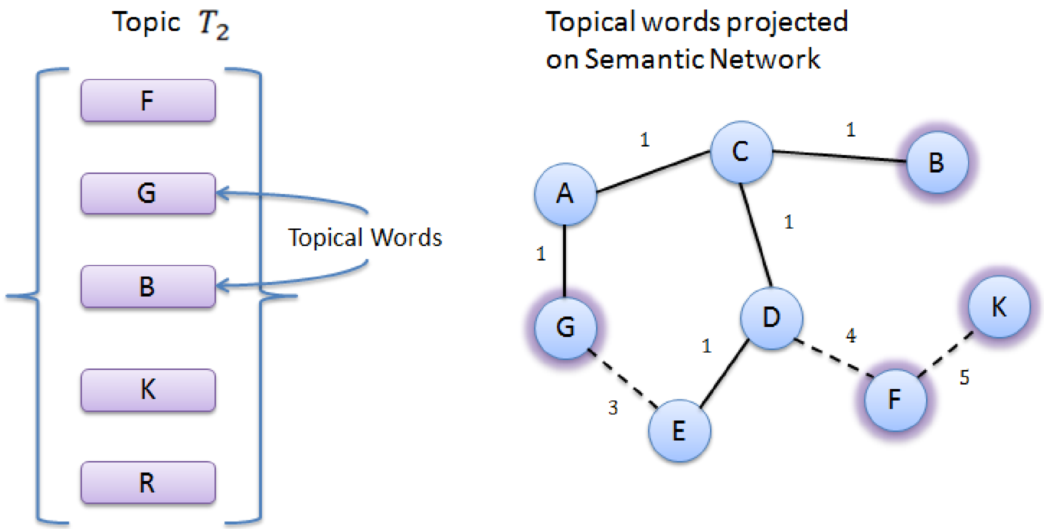
The semantic topical centrality (STC) of a topical word represents the importance of the word within the topic by endorsement from the semNet. Algorithm 2 shows how to calculate STC for topical words within a topic. STC of a topical word depends on the nearness of the projected topical word with respect to all other projected topical words for the topic. An attenuation factor α is used to discount the nearness between a pair of projected topical words ($0 \leq \alpha < 1$). The larger the value of shortest path length (l) between the pair of projected topical words in semNet, the smaller the effect it has on semantic centrality. In other words, STC counts the number of walks from one projected topical node to another, while penalizing longer walks. For each topic z, we calculate the semantic topic centrality dictionary ($\overline{STCD_Z}$). The STC for the i-th topical word in topic z can be extracted from the dictionary as $\overline{STCD_Z}(i)$.

Algorithm 4: Semantic topical centrality of a topical word

Input: Set of topical words in a topic (T), attenuation factor ($0 < \alpha < 1$)

Output: Semantic topic centrality dictionary ($\overline{STCD_T}$).

```
for topical word $w$ in $T$ do
    Initialize $STC(w)$= 0.001.
    for each word $n$ in the set $T - \{w\}$ do
        if a $path(w, n)$ exists in the semNet:
            $l$ = length of $path(w, n)$
            $STC(w) += \alpha^{l-1}$
    Add [$w$: $STC(w)$] to $\overline{STCD_T}$ as [key: value] pair
```



$$STC(F) = .001 + (0.5)^{7-1} + (0.5)^{6-1} + (0.5)^{5-1} = 0.111$$

$$STC(G) = .001 + (0.5)^{7-1} + (0.5)^{3-1} + (0.5)^{12-1} = 0.267$$

$$STC(B) = .001 + (0.5)^{6-1} + (0.5)^{3-1} + (0.5)^{11-1} = 0.283$$

Figure 57: Analyzing semantic coherence of topical words (poor coherence).

Example of topical words of topic T2 projected on the semantic network. Solid edges refer to direct edges (weight=1) between concept nodes, while dotted lines refer to multiple/indirect edges (weight>1) between concept nodes. Semantic topical centrality (STC) calculation (using Algorithm 4) for each of the top 3 words in a T2 with respect to the top-5 topical words is shown in this Fig 57. Alpha is set to 0.5. Far occurrence of projected topical words produce low STC values.

We can now construct a preliminary semantic topic centrality measure $\theta_z^{\sim}$ based only on centrality without statistical influence from topic modeling i.e. without word-topic membership $p(i|z)$ as:

$$\theta_z^{\sim} = \sum_{i \in z} \overline{STCD}_z(i) \qquad (28)$$

where i is a topical word of topic z. Finally, given the semantic topic centrality dictionary for each topic ($\overline{STCD_Z}$) and the available word-topic distribution $p(w|z)$ obtained from topic modeling, the semantic quality (θ_z) of a social topic z can then be calculated as:

$$\theta_z = \sum_{i \in z} \overline{STCD}_z(i) * p(i|z) \qquad (29)$$

where i is a topical word of topic z. The significance of Eq. 4 is that it provides us with a combined measure of both the semantic and the statistical membership of a topical word towards a particular topic. $\overline{STCD}_z(i)$ indicates the semantic membership of the topical word i towards topic z whereas $p(i|z)$ indicates the statistical membership of the topical word i towards topic z.

A higher value of θ_z implies that statistical results produced by the generative topic model (distribution of words over topic) is well-aligned to the inherent semantic network structure of the topical words. A low value of θ_z might imply one of the two things: (1) the probability distribution of words produced by the generative statistical model for the topic does not match the semantic structure of the topical words, or (2) the topical words do not possess significant semantic interconnection. In either case, a topic with low θ_z value indicates an inferior quality of topic recovered. Thus, our intuition is that close/near and semantically connected occurrence of topical words when projected on the semNet is a good indicator of the quality of topic.

Experimental Setup: For our experiments, we use a tweet collection consisting in total of 10.2 million tweets ranging from Jan 26th 2011 to Feb 7th 2011. Topics for the entire period of tweet data (approximately 2 weeks) are obtained by analyzing Twitter trends. For each day, we extract a set of 50 social topics. We randomly choose a total of 30 topics per day for scoring by the annotators. There are 5 annotators that score the quality of each of the 30 social topics on a 4-point scale: 3 = "Very good", 2 = "Good", 1 = "Neutral" and 0 = "Bad" based on the semantic coherence among the top-20 topical words in a topic.

Tweet Stream Day	**Social Topics**	
	Topical Words	***Annotator Rating***
1/26/2011	beach, shower, economy, summer, holiday	2
1/27/2011	null, yahoo, angel, glad, war	0
1/28/2011	government, jesus, allah, hate, watching	1
1/29/2011	egypt, tahrir, army, revolution, police	3
1/30/2011	love, time, heart, promise, moments	3

Table 8: Selection of social topics and annotator ratings

The annotators are given guidelines on how to judge a topic into the four quality classes mentioned above. Guidelines include showing illustrative examples of coherent topics or searching online. The main factor is deciding whether the topical words are interpretable, coherent, intuitive or meaningful; specifically whether the topical words would be natural choices for use when writing an article about that topic or if it is easy to find a one word abstract label for the topic by seeing the topical words (e.g., '*strings*', '*music*', '*spanish*', '*strum*' might refer to the topic label '*guitar*'). Eventually, it is left to the human annotators to make a final decision on topic quality, which is precisely the point, since humans use semantics to interpret language – and we use a semantic network to measure topic quality. We report the inter-annotator agreement scores in Table 8. On average, the annotators agreed on same quality class (based on the 4-point scale) for a topic on 83% occasions.

In Table 8, we show an assortment of the social topics that were scored under different ratings by annotators. Note how low scoring topics display limited coherence in terms of word semantics. Topics were extracted using Online LDA [103], with parameter settings as: 50 topics, 100 round of iterations, batch size (for sampling) of 200 tweets and a chunk size of ~25,000 tweets. For STC calculation, we set $\alpha = 0.5$.

Benchmarks: We compare our proposed approach to the Pointwise Mutual Information (PMI) technique based on term co-occurrence [95] and the Google Title Matches (GTM) [110]. The two benchmarks are described below:

(1) *Pointwise Mutual Information (PMI)* scores word pairs using term co-occurrence, such that for any two words, it represents the statistical independence of observing them in close proximity within a given corpus. In [111], Wikipedia is chosen as the corpus. Fixing the sliding window size to 10 words in order to identify co-occurrence, the PMI for two words, *x* and *y* are calculated as:

$$PMI(x, y) = log \frac{p(x, y)}{p(x). p(y)} \qquad (30)$$

When evaluating topics, the authors in [6] found PMI to produce better results in representing word similarity compared to previous semantic relatedness techniques, which are mostly based on Wikipedia page links or WordNet [42]. We consider the mean PMI as the representative score for the topic.

(2) *Google Title Matches (GTM):* We also compare our proposed technique against search-engine-based similarity methods described in [110]. Using an external data source like the World Wide Web and Google Advanced Search, we query the top-10 topical words in a topic and find the number of matches in the top-100 search results. For example, using the topical word set $\boldsymbol{w} = \{egypt, tahrir, army, revolution, police, \dots\}$[6], there were 134 matches with the top-10 words in the top-100 search results, so the $GTM(\boldsymbol{w}) = 134$.

Results. The results of the proposed approach when compared to the benchmarks are provided in Table 9 and Table 10. For each of the social topic evaluation methods, we use Spearman Rank Correlation and report the ρ values. The inter-annotator agreement (IAA) is also provided in the final column of Table 9 and Table 10 and serves as the gold standard for this task. IAA is calculated by using the Spearman Rank Correlation between an annotator and the mean of the remaining annotators for the particular topic.

Results show that the proposed approach performs better than PMI and GTM consistently. We believe this improvement can be contributed to the centrality features of the semantic network built from the rich RDF data (semNet). Although PMI uses Wikipedia, its semantic relatedness is bounded by co-occurrence between topical word pairs alone. Thus, PMI cannot capture the *pattern or motif in which the topical words exist within the semantic network, whereas θ_z characterizes the interconnection in terms of centrality importance among the topical words when projected onto the semantic network.* An alternative perspective is that θ_z characterizes the diffusion behavior of the topical words in semNet. Diffusion behavior refers to the spread of information using the

[6] The actual query performed on Feb 22nd, 2012: [egypt, +tahrir+army+revolution+police+egyptian+watching+world+support+jail]. Use '+' to prevent Google from using synonyms or lexical variants of the topical words.

topical words. Considering interpretability as semantic information, it means that the higher is the centrality; the better is the diffusion and better is the interpretability. On average, our technique improves performance by 10.3% when compared to existing benchmarks for human interpretation of social topics.

Tweet Stream Day	**Methods**				
	PMI	***GTM***	$\boldsymbol{\theta_z^{\sim}}$	$\boldsymbol{\theta_z}$	***IAA***
Jan 26th	0.61	0.60	0.67	0.69	0.71
Jan 27th	0.62	0.66	0.72	0.72	0.73
Jan 28th	0.55	0.61	0.66	0.67	0.70
Jan 29th	0.62	0.55	0.76	0.76	0.79
Jan 30th	0.53	0.61	0.67	0.67	0.69
Jan 31st	0.61	0.66	0.71	0.73	0.73

Table 9: Spearman rank correlation values for proposed approach against benchmark in Jan 2011.

Tweet Stream Day	**Methods**				
	PMI	***GTM***	$\boldsymbol{\theta_z^{\sim}}$	$\boldsymbol{\theta_z}$	***IAA***
Feb 1st	0.65	0.71	0.75	0.76	0.78
Feb 2nd	0.52	0.55	0.66	0.68	0.68
Feb 3rd	0.65	0.66	0.73	0.74	0.75
Feb 4th	0.62	0.68	0.72	0.72	0.76
Feb 5th	0.60	0.65	0.71	0.72	0.73
Feb 6th	0.62	0.66	0.71	0.71	0.75
Feb 7th	0.62	0.63	0.69	0.70	0.70

Table 10: Spearman rank correlation values for proposed approach against benchmark in Feb 2011.

Moreover, $\theta_z^{\sim}$ and θ_z are surprisingly close in performance, indicating that it is the semantic backbone which is responsible for the major improvement over the benchmarks. However, a combination of semantic and statistical relatedness performs best, i.e. $\theta_z^{\sim}$ does not have better performance than θ_z. The combined approach (Eq. 29) possibly smoothes over irregularities which can be introduced when very common words of the English language are modeled as topical words, causing significant semantic connectedness but low statistical word-topic membership scores.

We also note that GTM usually outperforms PMI, except on certain occasions. One such scenario is when the social topics for the day do not account for sufficient breaking news stories. Breaking news stories are indexed as articles/documents in search engines in much larger proportion compared to some other topics (a music video release), which makes GTM perform better for news than music or entertainment related topics. The second case where the performance of GTM is degraded compared to PMI is when there is lack of sufficiently good topics. For example, on Jan 29th, there was a Blackberry outage in North America and Twitter had significant downtimes. The inconsistent temporal data produces poor topics, which causes GTM to perform considerably worse than PMI. It is unclear if this observation is attributed to the search engines ranking scheme.

There are some interesting possibilities that arise from this line of research. The semantic centrality proposed in this paper is the first in a plethora of options for using complex network feature that are beyond similarity based techniques for judging the role of word connectedness in topic modeling (social or otherwise). Three avenues of future work can be considered: firstly, the attenuation factor (α) in measuring STC of a word *w* (Step 6, Algorithm 4) can varied by making it sensitive to the importance of another topical word *n*. Thus, α could have higher values based on the named entity recognition for *n* (is *n* a person/place?) or the word-topic membership score. Secondly, the centrality of topic networks [34] (network of documents based on topic modeling) can be explored as an alternative evaluation technique of the topics extracted. One limitation of the current work is that it often fails to link emotion words with concept words in the semantic network. Emotions occur very frequently in social data. Thus, future research could also combine sentiment analysis (connected emotion words) with the semantic network concepts to enrich the understanding of sentiments regarding a concept and opinion mining. The performance of all the existing topic models in terms of human and semantic interpretability can be verified by our technique.

We address an important issue in this paper: the problem of automatically evaluating the quality of social topics based on semantic interpretability. Fundamentally, topic modeling is a generative statistical technique to find latent topics in data. For social data however, retrieved topics are often non-interpretable and lack semantic richness. This is a non-trivial problem, since social topic mining aims to serve social applications used by humans; and humans prefer intuitive information. Thus, automatic detection of good quality topics is necessary for most applications based on social information, e.g. social recommendations (video, ads etc.).

This paper attempts to evaluate the quality of social topics using the centrality of topical words found in a semantic network. Unlike previous work which uses a corpus for evaluation, we utilize a semantic network built from DBpedia RDF data. Network analysis on these interconnected topical words reveals rich patterns of diffusion, which are used to score the quality of a topic. Our technique proves to be better at reflecting human interpretability of social topics compared to existing benchmarks. This work will allow researchers and developers to automatically detect quality topics in the topic space with greater accuracy, thus eliminating the chance of spurious recommendations by rejecting bad quality topics.

CHAPTER 6: LEARNING COGNITIVE MODELS FOR NATURAL LANGUAGE SEARCH

The future of natural language (NL) search hinges on semantic concept detection in queries, which includes detection of keywords in a query, followed by construction of some meaningful connected network comprising of such keywords [45]. This connected network of keywords is called a semantic subnet [46]. These keywords, represented as vertices in the subnet, together comprise what is called a semantic field [47]. The goal of this chapter is two-fold: (1) to efficiently recover the semantic sub-network from NL queries and (2) to generate robust concept detection techniques that improve semantic search We show how information from the semNet can be used to detect concept hierarchies, which boosts the concept relevancy detection for natural language search.

Prior research suggests three main motivations for extracting semantic subnets from NL queries. Query subnets can be used to generate a candidate set of concepts within a larger ontology (like of DBpedia RDF network/ Google knowledge graph), which may align to the words in the query subnet [46]. Said alternately, a pattern isomorphic to the query subnet can be detected in the larger ontology, which assists domain identification and query expansion [46]. Secondly, a query subnet can act as a NL interface to concept graph databases [48], facilitating semantic information retrieval [49] and improved query understanding and semantic search [50]. Finally, semantic subnets enable identification of event structures within sentences [51] and assist higher-level NLP tasks, like Question Answering (QA) [52].

It is possible to detect semantic keywords in NL queries using methods like Named Entity Recognition (NER) or Semantic Role Labelling (SRL) [53]. Both these techniques provide a higher level of abstraction than the basic syntax tree. However, our task goes a step further: we aim to find out *how these keywords are semantically connected* in terms of a network. This is very difficult to achieve using NER alone, since detecting the named entities provides limited information about their relations. SRL does a better job at concept level parsing using predicate logic, but is bound by the strict predicate grammar. Therefore, although techniques such as NER and SRL is core to NLP, there is an inherent gap between requirements of intelligent tasks (like QA, textual entailment) and several state-of-the-art NLP techniques [54].

As search is becoming more collaborative and social, queries turn noisier [50]. Often, the conceptual structure of the NL query is difficult to extract using simple (Parts-Of-Speech) POS-based dependency parsing. Imprecision of NL usage is a major obstacle to computation with NL. Therefore, it is necessary to develop a technique that partially relaxes the rigid grammar of the language. While imprecise or varied grammatical constructions are difficult to capture using POS or predicate logic, note that the human cognition can often eliminate such noise to interpret meaning. At first this sounds like a baffling fact; but everyday experiences reveal that human cognition is significantly more robust in extracting meaning from poorly constructed sentences compared to state-of-the-art NLP techniques for NL understanding. If we assume that ‘meaning’ of a NL sentence is captured in its semantic subnet, then it would be logical to conclude that human cognition possesses a more noise-resistant process of extracting semantic subnets. A rational explanation for this cognitive robustness is the presence of an improved model for detecting semantics in NL and subsequently constructing semantic information in the brain.

Cognitive psychology has some interesting theories as to how the mind deals with imprecision, uncertainty and complexity of language [55]. One such theory, called the structure-of-intellect model, proposes that humans perceive concepts contained within the words of a sentence as a semantic *form* [56]. Guilford referred to *forms* as ‘products’ - entities that describe granularities in any content perceived. His model has been widely used to study the cognitive intellect and the kinds of information that humans can extract from any observed semantic data (like NL sentences) [57]. In the context of NLP, semantic *forms* reflect the kinds of information that the human cognition can process from any semantic field. Five such *forms* were proposed by Guilford, namely *units, classes, relations, systems,* and *transforms*. *Forms* resemble levels of granularity, which allows extraction of finer or coarser information depending on the noise level of perceived data. The physical interpretation of this cognitive model is that *no matter what the data is: at different resolutions or granularities, different features and relationships emerge.* The model argues that human cognition is robust to noise because it dynamically changes the resolution at which data is to be semantically interpreted [47].

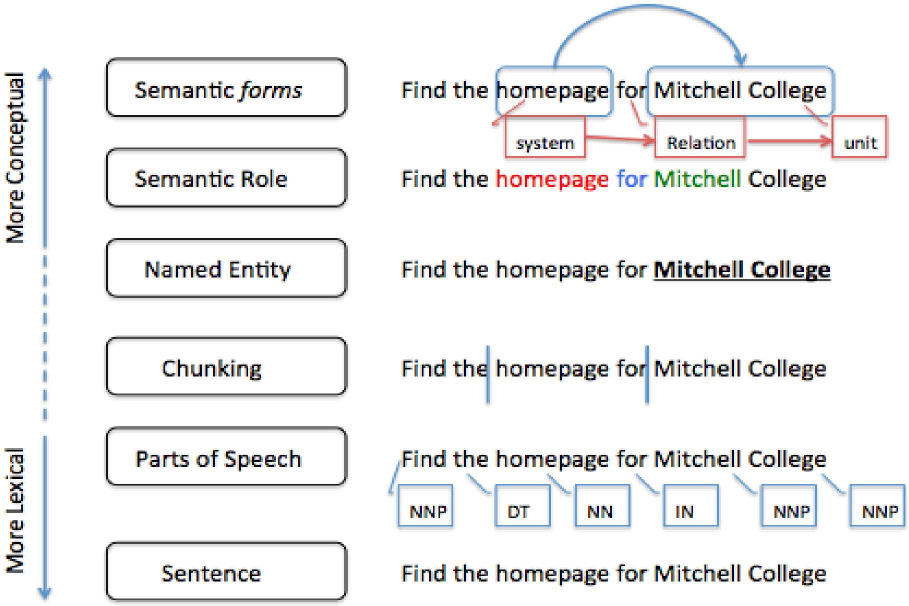

More Conceptual
More Lexical
Semantic forms
Semantic Role
Named Entity
Chunking
Parts of Speech
Sentence
Find the homepage for Mitchell College
system
Relation
unit
Find the homepage for Mitchell College
Find the homepage for Mitchell College
Find the homepage for Mitchell College
Find the homepage for Mitchell College
NNP
DT
NN
IN
NNP
NNP
Find the homepage for Mitchell College

Figure 58: Level of abstraction in different NLP techniques: from lexical to conceptual.

Recognizing the potential of cognitive approaches in semantic information modelling, we propose to leverage semantic *forms* in the extraction of semantic sub-networks from NL queries. These semantic *forms*, when connected in some networked pattern, becomes responsible for understanding the scope and context of a concept, and assists functional retrieval of related concepts and question answering/response [57]. Thus, our main insight in modelling semantic *forms* and their interaction patterns in NL is grounded on the idea: *the subsurface form space demonstrates the query intent (expresses semantics) better than superficial (lower) query syntactical features, which might vary depending on diverse query construction.* In other words, the higher is the level of abstraction for labelling, the more robust the extraction should become. This idea of cognitive abstraction provided by semantic *forms* is shown in Fig. 58.

The main contributions of my work here are:

- We propose the use of semantic *forms*, borrowed from cognitive science, as label category for NL sequence labelling tasks.
- We propose a conditional random field based method of implementing the structure of intellect model, by labelling query words with semantic *forms* and analyzing the interconnected patterns in which such *forms* exist within a semantic field.

We perform experiments on three diverse query datasets consisting of TREC, QA-type and Web queries to justify the robustness of our approach to varying noise levels. The proposed approach comprehensively outperforms existing works on query subnet detection [46].

6.1 The Structure of Intellect Hypothesis

J. P. Guilford introduced the structure-of-intellect model in [56], which covers the notion of semantic *forms* as ‘products’. ‘Products’ are the result of applying some cognitive operation (cognition, retention etc.) on specific content (semantic, symbolic etc.). The model has since been used, studied and analysed substantially in the cognitive science community. A detailed view of human cognitive semantics in linguistics is provided in [47]. Probabilistic models of cognitive linguistics are described in [55]. An insightful introduction to human cognitive abilities is available in [57].

The main hypothesis proposed by the Structure-of-Intellect model is that *human cognition is robust to noisy sentence constructions because it strives to detect semantic information at different levels of granularity*. The noisier the sentence, the coarser is the granularity of semantic information detection employed by the human cognition. In this section, we qualitatively introduce the different semantic *forms* from Guildford’s structure-of-intellect cognitive model and describe how *form* interaction patterns play a key role in semantic subnet extraction.

Semantic *forms* consist of five entities that capture the structure of information contained within a natural language sentence as perceived by the human cognition. A remarkable thing about semantic *forms* is that they are structured as granular hierarchies (i.e. one *form* is composed of other *forms*). Following is a description of the semantic *forms* starting with finer granularity:

Unit. Every item of a query sentence can be regarded as part of some chunk, of which *units* are the most basic entities. *Units* will cover most words of a sentence, from intangible ideas like ‘love’ to tangible objects like ‘cars’. For example, the name ‘Anna Chakvetadze’ is a *unit*. The cognition of semantic *units* has to do with one’s ability to recognize words, i.e. one’s vocabulary [56].

Class. When *units* have one or more attributes in common, they can be grouped in *classes*. In a semantic network, *units* belonging to a *class* will share connectivity to at least one common attribute node. *Classes* can be narrow or broad. For example, the *unit* 'Anna Chakvetadze' can belong to the very broad *class* 'female', a moderately broad *class* 'Russia' or a narrow *class* 'Tennis'. The size of the *class* (narrow/ broad) qualitatively determines the size of the search space for related concept retrieval.

Relation. Relations are *kinds* of connections between *units*. When any two entities are connected in the semantic network, there are three items of information involved – two *units* and the *relation* between them. *Relations* between search keywords play an integral role in realizing *class* or *unit* interconnections in the query. For example, 'Steffi Graf' and 'Andre Agassi' could be connected by the *relation*: *married*, while both belonging to the *class: tennis players*.

System. A *system* is the most complex item in semantic information. *Systems* are composed of more than two interconnected *units*. *Systems* may also comprise of overlapping *classes*, multiple interconnecting *units* and diverse *relations*. They often occur as an order or sequence of *units*. Add 'Maria Sharapova' and 'Sasha Vujacic' to the previous example of 'Steffi Graf' and 'Andre Agassi', and we get a *system*: *married sportspersons*.

Transform. A *transform* is a semantic *form* that captures any sort of change in the information perceived from a query word. This change (transformation) in itself is a semantic *form*. *Transforms* are usually caused due to the existence of polysemy in a sentence. *Transforms* occur when *units* can be represented as coarser granularities, like *classes* or *systems*. We will explain the implementation of *transforms* in further detail a little later. Some examples of some word-*form* pairs are shown in Table 11.

Word	***Form***	**Word**	***Form***	**Word**	***Form***
Thursday	*unit*	market	*system*	Mansion	*unit*
witches	*class*	driving	*relation*	school	*system*

Table 11: Examples of word-*form* pairs.

Strata based on node centrality

Nodes that possess a higher degree in the network have more neighbors with direct edge connections. In terms of topology, the higher degree node is more central [58], i.e. it is a more general concept node (super category). We employ this aspect to understand the *strata* of words, since *strata* are granular information entities. Thus, for every node (which represents a concept word), we can calculate the normalized degree centrality, which gives us an estimate of the generality of the concept node. This follows the intuition that a crawler will encounter the concept 'Russia' many more times than the concept 'Anna Chakvetadze' while browsing Wikipedia.

Therefore, we calculate the degree distribution for the semNet nodes. This distribution follows a power law, as is the characteristic of semantic ontologies [59]. Then for a node, if its centrality belongs to the greater 80 percentile for the nodes centrality distribution, we suppose it is a *system*. For nodes with greater than 50 percentile but below 80, we assign it to *strata class*. Nodes with centrality belonging to less than 50 percentile are treated as *units*. These thresholds are designed based on the properties of systems that follow a power-law degree distribution, meaning they obey the 80-20 rule[7].

[7] The 80-20 rule is sometimes referred to as the Pareto Principle, which states that for many systems, approximately 80% of the effects is generated by 20% of the causes.

6.2 Computational Modeling of the Cognitive Hypothesis

In our proposed approach, consider each observed symbol as the tuple: {word, POS tag, NP chunk number}. We can employ basic sequence labeling idea here, by considering the chain of *forms* that link the tuples as hidden states. Using the training data, a CRF model [60] can then assign optimal state chains to samples of observed symbols, from which we learn the kinds of *form* chains (interactions) that exist. Steps for computationally modeling the cognitive notion of semantic *forms* are described in this section.

We begin with formal definitions, followed by describing some pre-processing techniques and finally, the detailed description of model features.

Consider an NL sentence *Q*. Our assumption is that *Q* is a carrier of information. Every word is a linguistic variable in *Q*. It is well known that information is expressible as a restriction (i.e. a constraint) on the values that a variable can take [61]. By this flow of thought, consider *W* as a constrained variable in *Q*, let *R* be the constraining relation in *Q* and ζ (zeta) represent how *R* constrains W. Then, every NL sentence *Q* can be represented as:

$$Q \rightarrow W\ \zeta\ R$$

It is possible for *W* to be a vector-valued random variable. The primary constraint *R* is a restriction on the *strata* values of *W* and is probabilistic in nature. Hence, *W* can take up values of different *strata* from the set (*unit, class,* … , *transform*) with probabilities ($p_{unit}, p_{class}, \ldots, p_{transform}$) respectively. Thus, *W* is constrained using the probability distribution *R* as:

$$W\ \zeta\ (p_{unit}\backslash unit +\ p_{class}\backslash class + \cdots + p_{transform}\backslash transform)$$

The singular variable W takes values from the universe of discourse U, such that values of W are singletons in U (see Fig. 9). On the other hand, the semantic *strata* of W is a variable whose values depend on the granular collections in U. Said alternately; *the granular precision of a word in U is expressed through its semantic strata*. The type of *strata* assigned to a word depends on the cluster size of elements in U that have common attributes or behavior related with the concepts of that word.

The overall process is described at an abstract level in Fig. 59, where ellipses represent the *stratum* of a word. Consider four key words W1, W2, W3, and W4 in the query (Q) that need to be connected as some semantic subnet. Let $stratum(w_i)$ denote the *stratum* associated with the word w_i. In step (i): we are uncertain of the semantic subnet or canonical form of Q. In (ii), our goal is to label the words with semantic *strata*. In (iii), we use the *strata* interconnection patterns to retrieve the connection among the *strata* for the four words when they exist together in some Q. Finally, in (iv), we can connect the original words as a query subnet by shadowing the connected *strata* pattern.

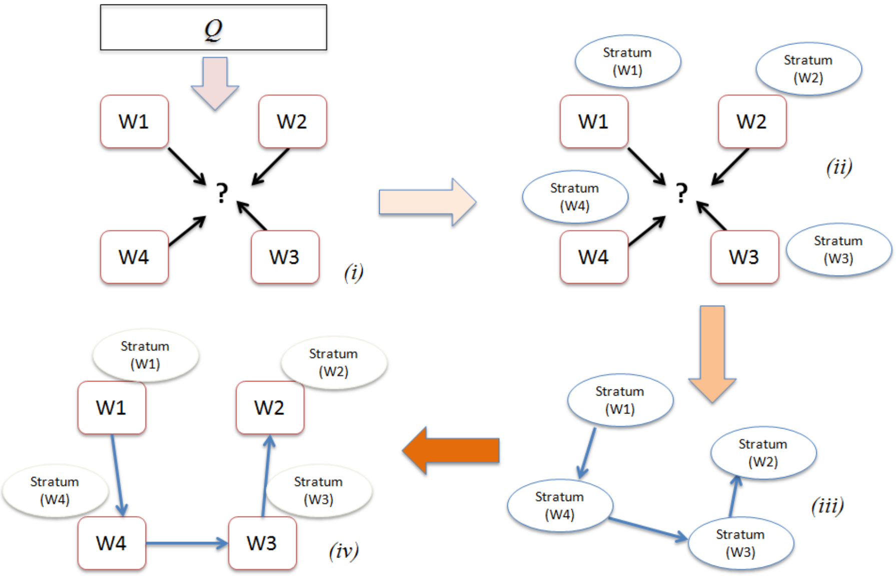


Figure 59: Overall Process of subnet extraction.

We employ basic pre-processing techniques such as stop-word removal, POS tagging and chunking before we proceed to *strata*-based tagging. Stop-word removal is performed using the well-known Python NL toolkit stop word list. We used the Stanford POS tagger for POS tagging. For long queries, chunking is necessary. The chunking process is inspired by [52]. Let us briefly describe the chunking process before we proceed further.

Chunking. Consider Q to be a query sentence in natural language L containing words belonging to the vocabulary set V. Let S_Q be the sequence of POS-tagged symbols associated with a query Q, i.e.

$S_Q = \{s_1, s_2, \dots, s_N\}$, where $s_n = \langle w_n, t_n \rangle$, $w_n \in V$, $t_n \in T'$ for N words in Q and T' is the set of possible POS in English grammar. Given S_Q we can define the kth chunk $(\mathcal{C}_k)$ as:

$$\mathcal{C}_k = (\langle w_i, t_i \rangle, \langle w_{i+1}, t_{i+1} \rangle, \dots, \langle w_j, t_j \rangle)$$

for some $i < j \leq N$ and $1 \leq k \leq M$ for a total of M chunks in the query. Then, the task involves determining all the M chunks based on S_Q, s.t $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \dots \mathcal{C}_k, \dots, \mathcal{C}_M\}$. This generates the chunked query set:

$S_{QC} = \{\langle s_1, \mathcal{C}_1 \rangle, \dots, \langle s_l, \mathcal{C}_1 \rangle, \langle s_{l+1}, \mathcal{C}_2 \rangle, \dots, \langle s_N, \mathcal{C}_M \rangle\}$, where $\mathcal{C}_1 = (s_1, \dots, s_l)$, $s_l \in S_Q$, for some l, $1 \leq l \leq N$.

Following similar methods as used in [52], we can find $p(\mathcal{C}_1, \mathcal{C}_2, \dots \mathcal{C}_k, \dots, \mathcal{C}_M \,|S_Q)$ as:

$$p(\mathcal{C}_1, \mathcal{C}_2, \dots \mathcal{C}_k, \dots, \mathcal{C}_M \,|S_Q) = \Psi \prod_{i=1}^{M} p(s_{i-1}|s_i, \mathcal{C}_i)\, p(s_{i+1}|s_i, \mathcal{C}_i)\, p(s_i|c_i)\, p(\mathcal{C}_i) \tag{31}$$

where, $\Psi = {}^{1}\!/_{p(s_1, s_2)\, p(s_2, s_3) \dots, p(s_{N-1}, s_N)}$

Estimating Probabilities. Consider the training set $T = \{(\boldsymbol{x_k}, \boldsymbol{y_k})\}_{k=1}^{K}$. Each element of the training set is a vector such that for $\mathrm{N_k}$ words in the $\mathrm{k^{th}}$ sample, $\boldsymbol{x_k} = < x_{k1}, \dots, x_{kN_k} >$, where $\mathrm{x_{k,i}}$ denotes the $\mathrm{i^{th}}$ word in the $\mathrm{k^{th}}$ training sample and the corresponding labels $\boldsymbol{y_k} = < y_{k1}, \dots, y_{kN_k} >$. Let us denote a word as w and the label as γ. Let us also denote the current index as 'curr', the next index as 'next' and the previous index as 'prev', such that the previous word is $\mathrm{w_{prev}}$ or the current label is γ_{curr}. Note from Eq. 1 that we are trying to find three conditional probabilities: $p(w_{curr}|\gamma_{curr})$, $p\left(w_{prev}\middle|w_{curr}, \gamma_{curr}\right)$ and $p(w_{next}|w_{curr}, \gamma_{curr})$. Also, let $I = \{(k, n)|k = 1, \dots, K, n = 1, \dots., N_k\}$. Then the required probabilities can be estimated as (where #{} denotes the size of the set):

$$p(w_{curr}|\gamma_{curr}) = \frac{\#\{(k,n)\in I \mid x_{k,n}=w_{curr}, y_{k,n}=\gamma_{curr}\}}{\#\{(k,n)\in I \mid y_{k,n}=\gamma_{curr}\}} \qquad (32)$$

$$p\left(w_{prev}\middle|w_{curr}, \gamma_{curr}\right) = \frac{\#\{(k,n)\in I \mid x_{k,n-1}=w_{prev}, x_{k,n}=w_{curr}, y_{k,n}=\gamma_{curr}\}}{\#\{(k,n)\in I \mid x_{k,n}=w_{curr}, y_{k,n}=\gamma_{curr}\}} \qquad (33)$$

$$p(w_{next}|w_{curr}, \gamma_{curr}) = \frac{\#\{(k,n)\in I \mid x_{k,n+1}=w_{next}, x_{k,n}=w_{curr}, y_{k,n}=\gamma_{curr}\}}{\#\{(k,n)\in I \mid x_{k,n}=w_{curr}, y_{k,n}=\gamma_{curr}\}} \qquad (34)$$

The task of tagging words of a sentence with semantic *strata* from the set of *strata* (*F*) makes use of the CRF model described here. The result is the set S_{QF} of *strata* labeled words. First, we briefly describe CRF in the light of our problem, followed by feature functions and learning weights.

Conditional Random Field (CRF). Consider two random variable sequences ***X*** and ***Y*** of the same length. Let ***X*** be the input sequence and ***Y*** be the output sequence and let us denote $\boldsymbol{x} = [x_1 \dots x_n]$ and $\boldsymbol{y} = [y_1 \dots y_n]$ for the generic input and *strata* label sequence respectively. A CRF on (*X*, *Y*) is specified by two vectors: a local feature vector $\boldsymbol{l_f}$ and a corresponding weight vector $\boldsymbol{\lambda}$.

A *state feature* is an element of $\boldsymbol{l_f}$ of the structure $state(y, \boldsymbol{x}, i)$ where i is the input position, *y* is a label and $\boldsymbol{x}$ is the input sequence. A *transition feature* is an element of $\boldsymbol{l_f}$ of the structure $tran(y, y', \boldsymbol{x}, i)$ where *y*, $y'$are labels.

The global feature vector for an input sequence $\boldsymbol{x}$ and a label sequence $\boldsymbol{y}$ is:

$$\boldsymbol{F}(\boldsymbol{y}, \boldsymbol{x}) = \sum_i \boldsymbol{f}(\boldsymbol{y}, \boldsymbol{x}, i) \qquad (35)$$

Individual feature functions are described a little later. A conditional distribution that obeys the Markov property, which is:

$$p\left(Y_i \middle| \{Y_j\}_{j \neq i}, \boldsymbol{X}\right) = p(Y_i | Y_{i-1}, Y_{i+1}, \boldsymbol{X})$$

can be written as:

$$p_{\boldsymbol{\lambda}}(\boldsymbol{Y}|\boldsymbol{X}) = \frac{\exp\{\boldsymbol{\lambda}.\boldsymbol{F}(\boldsymbol{Y},\boldsymbol{X})\}}{Z_{\boldsymbol{\lambda}}(\boldsymbol{X})} \qquad (36)$$

where $Z_{\boldsymbol{\lambda}}(\boldsymbol{x}) = \sum_{\boldsymbol{y}} \exp\{\boldsymbol{\lambda} . \boldsymbol{F}(\boldsymbol{y}, \boldsymbol{x})\}$.

Note the denominator of Eq. 36 is independent of $\boldsymbol{y}$. Then the most probable sequence of *strata* labels *(**y***)* for the input sequence $\boldsymbol{x}$ is:

$$\boldsymbol{y}^* = \arg max_{\boldsymbol{y}}\, p_{\boldsymbol{\lambda}}(\boldsymbol{y}|\boldsymbol{x}) = \arg max_{\boldsymbol{y}}\, \boldsymbol{\lambda}.\boldsymbol{F}(\boldsymbol{y}, \boldsymbol{x}) \qquad (37)$$

Eq. (37) can be solved using the Viterbi Algorithm [60]. In order to optimize the maximum likelihood of the training set, we use the preconditioned conjugate gradient method [62]. This is a well-known technique for linear and non-linear optimization. Fig. 60 illustrates the CRF model with its feature functions.

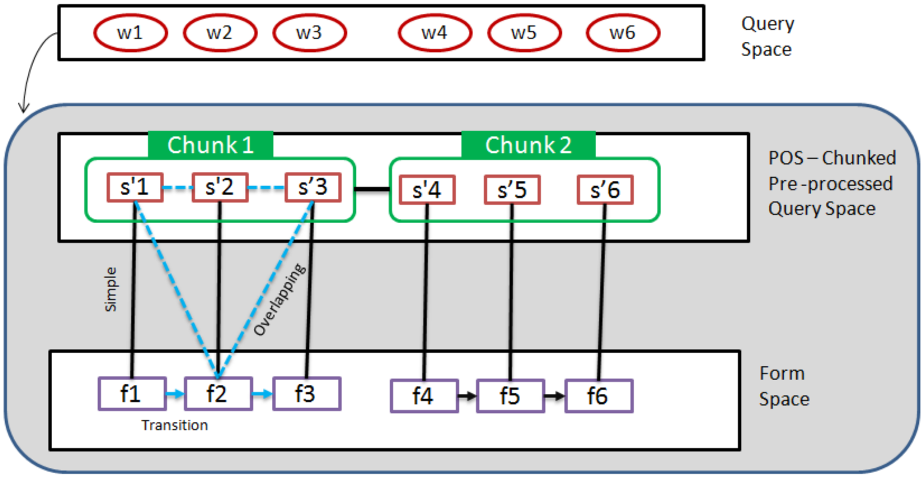


Figure 60: Types of feature functions in CRF.

Feature Functions. Feature functions are key components of CRF. The general structure of a feature function is $z(f_{i-1}, f_i, \boldsymbol{x}, i)$ which looks at two adjacent states f_{i-1}, f_i and the whole input sequence $\boldsymbol{x}$ where i is the current location in this sequence, and assigns some weight to the observed feature. They can be defined in different ways, e.g., we have a feature like: if the current word is 'Nile' and the current state is '*unit*' then we give the feature a positive weight, otherwise not. Each feature function $z(f_{i-1}, f_i, \boldsymbol{x}, i)$ has a binary output and can take as inputs particular values of the current *strata* f_i and the previous *strata* f_{i-1}.

We use the training corpus queries to build the atomic feature set for the CRF. Let F_u represent *unit,* F_r represent *relation*, F_c represent *class* and F_s represent *system*. In the examples below, a binary value of 1 indicates the presence of the feature, and 0 indicates the lack of the feature. '∧' denotes logical AND. We implement four types of binary atomic features:

(1) *Simple Feature Function*: A simple feature function depends only on a word and its connected *strata*. For example,

$$z(f_{i-1}, f_i, \boldsymbol{x}, i) = \begin{cases} 1, & if\ x_i = '\ music'\ \wedge\ f_i = F_s \\ 0, & otherwise \end{cases}$$

(2) *Overlapping Feature Function*: An overlapping feature function depends on *strata* of a word and on its successor word. Under normal conditions, Hidden Markov Models are unable to realize overlapping features (unlike CRFs). A suitable example would be:

$$z(f_{i-1}, f_i, \boldsymbol{x}, i) = \begin{cases} 1, & if\ x_i = '\ and'\ \wedge\ f_{i-1} \neq F_r \\ 0, & otherwise \end{cases}$$

(3) *Strata Transition Feature Function*: A *strata* transition feature function depends on successive *strata* such as:

$$z(f_{i-1}, f_i, \boldsymbol{x}, i) = \begin{cases} 1, & if\ f_{i-1} = F_r\ \wedge\ f_i = \{F_u, F_c\} \\ 0, & otherwise \end{cases}$$

(4) *Mixed Feature Function*: A mixed feature uses successive *strata* and preceding/following words. For example,

$$z(f_{i-1}, f_i, \boldsymbol{x}, i) = \begin{cases} 1, & if\ f_{i-1} = F_s\ \wedge\ f_i = F_r\ \wedge\ x_{i-1} = '\ homepage' \\ 0, & otherwise \end{cases}$$

In Fig. 60, each '*s*' element in the POS-chunked pre-processed query space represents a tuple <word, POS, NP Chunk number>. There are 828 atomic features in our system, obtained from words in the vocabulary. This initial feature set is then grown using feature induction [60], resulting in a total of 23,713 features. Feature Induction is a well-known technique in machine learning to increase the feature set. It begins with a small seed set of features. Then it creates a set of candidate features consisting of observational tests. The candidate features are evaluated based on the highest gain and a subset is added to the model. A quasi-Newton method is used to adjust all parameters of the CRF model to increase the conditional likelihood. When training the CRF, we use pre-conditioning to ensure fast convergence of the conjugate gradient method [62]. We tested how fast (in terms of number of iterations) the objective function reaches close to its maximum attainable value. On the average, our technique requires 12-13 forward-backward iterations to reach an objective function value, which is in close proximity (~96%) to the maximum.

Feature Generation: Given a query Q, we can now score a labeling (***y***) by summing up the weighted features over all the words in Q as was described in Eq. 37. There are two individual probabilities involved in the process that need to be learned from the training data. These are the emission and the transition probabilities [62]. The emission probability estimates the probability that a word belongs to a certain *strata* when it is observed at some index in Q. The transition probability estimates the probability of observing two adjacent *strata* in a label chain (sequence of *strata*-tagged words).

State Functions. The CRF labeller's state feature set is assorted using a feature builder. The feature builder is trained using a seed set of words and their related *strata* obtained using DBpedia RDF resource. Fig. 61 illustrates how words and their tagged *strata* are collected from a training query. Keywords are identified from a query by stemming and eliminating stop words. Using DBpedia to classify the word into a semantic *strata* follows this, as described in Section 4. The vocabulary containing words → *stratum* is updated as more training examples are seen and used by the feature builder.

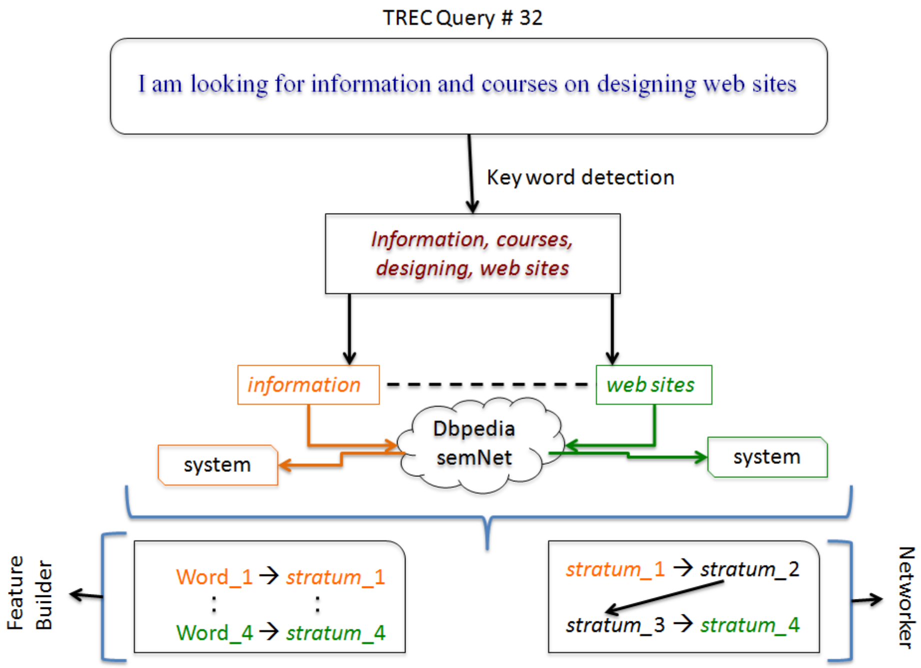


Figure 61: Using queries to build training set

Transition Functions. Given enough training samples of the sentence *Q*, the variable *W* and constraint *R*, we can deduce the pattern ζ, which identifies how *R* constrains *W*. This pattern ζ contains information about the ordering in *Q* with respect to *W* (remember *W* could be vector-valued) such that they are mapped to *R*. That is to say, every *stratum* has some specific interaction pattern with other *strata* when they exist together/adjacent in *Q*. This interaction pattern among *strata* in *U* is captured in ζ. Interaction patterns provide insight into the question: how are three or more *strata* connected when they appear in Q? For example, if we see three words {*A, B, C*} having *strata* {*relation, class, unit}* respectively, then would the query subnet be of the ordering A-B-C, or B-A-C or C-A-B?

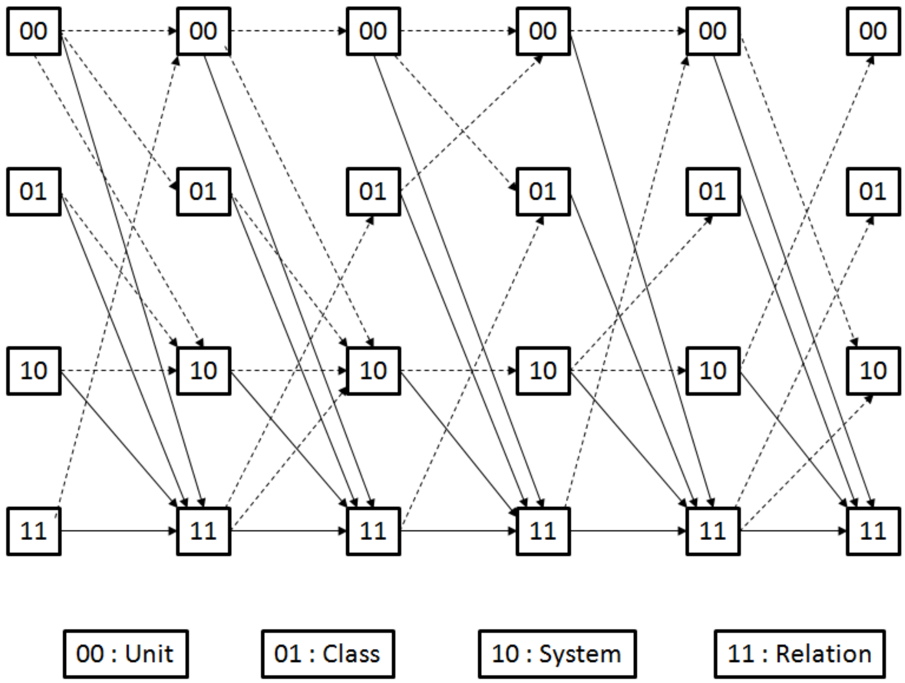


Figure 62: Trellis diagram of possible Viterbi paths representing sequence of labeled forms.

From the training data, a 'networker' learns interaction patterns at the chunk level, modelling each chunk as a potential branch for a rooted semantic query subnet. This viewpoint is derived from the observation that branches of most annotated query subnets are composed of individual or contiguous chunks of the original query. The *strata* interaction set $\tilde{F}$ is a simple ordered set: $\{(f_i, f_j)\}$, where $f_i, f_j \in F$ representing a complete or part of a directed chain $f_i \rightarrow f_j$. We only use *strata* connected within a chunk to populate one element of the set $\tilde{F}$. That is to say, *strata* at the border of two chunks are not considered part of interaction. Fig. 61 shows results obtained from the training data by means of a Trellis diagram.

The key property of the Trellis decoder in Fig. 62 is that: to every possible state sequence in Q, there exists a unique path through the Trellis decoder. Solid arrows indicate probabilities greater than 0.5 whereas dashed arrows indicate probabilities < 0.5. Individual edge probabilities are not shown to avoid cluttering the figure.

Handling Transforms and Word Sense Disambiguation. As mentioned earlier, *transform* is a *stratum* that captures the change in granularity of an entity in the universe of discourse. Fig. 63 describes the *transform* phenomenon in the universe of discourse. In certain cases, entities that have been initially labeled as *units* by the labeler can be better represented as a *class* or *system* once more words in the query are seen by the labeler. For example, when a query contains the word 'apple', a labeling algorithm might not understand the users' intent, since there are multiple possibilities: a 'fruit' or the company 'Apple Inc.' or the idiom – 'apple of the eye'? The *strata* of the word 'apple' could change from being a *class* (fruits) to a *system* (company) to just a *unit* in the idiom.

This is similar to issue of word-sense disambiguation in NLP [63]. A naïve way to resolve this is by first retrieving information regarding the specific concept nodes which can potentially change *strata* during labelling. This information can be obtained using Wikipedia disambiguation pages [64]. There also exists an RDF dataset in DBpedia containing all of Wikipedia's disambiguation pages. Usually the disambiguation pages can be sorted into various levels of granularity. That is, using a similar technique of centrality as described in earlier chapters, the disambiguation concept pages can be classified ranging over *units, classes* and *systems*. Thus, we know the possible *strata* of each disambiguation concept. The idea is illustrated in Fig. 64.

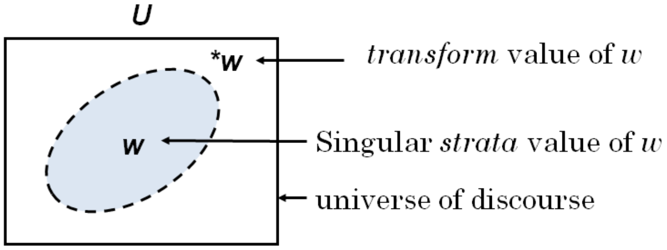


Figure 63: Transforms in the universe of discourse

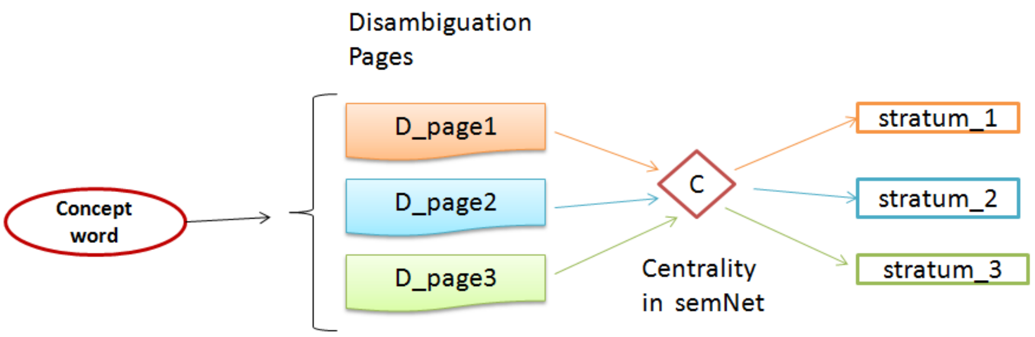


Figure 64: Handling transforms

Now, let us maintain a dictionary of words for various disambiguation documents as well as the co-occurrence words in those documents. Thus, given a disambiguated word w in some query Q, we can find the set of disambiguated documents D which contain w. Probability of a possible transform for w → $*w$ with respect to some $d \in D$ is calculated as:

$$P(d, w) = |co(d, Q)|/|Q| \qquad (38)$$

where $|Q|$ represents the number of words in the query and $|co(d, Q)|$ represents the number of words that co-occurred in d and Q. Then, the *transform* can be represented as:

$$transform(w) = (*_w) = stratum(d^*),\ where\ d^* = arg\max_D P(d, w) \qquad (39)$$

Eq. 39 refers to us choosing the document (d^*) as representative of the concept word w for which there is most co-occurred words between the document and the query Q.

6.3 Applications

We strive to better model the conceptual linkage among words in a query sentence, such that the query subnet signifies a canonical form for the query, meaning it can be searched for in large graph databases with various data attributes (social or semantic), including DBpedia RDF graph. Labeling using semantic *strata* might seem close to SRL in the sense that both produce some sort of parse tree. However, SRL produces a syntactic tree (POS-heavy) using predicate logic (verb-actions) whereas semantic *strata* aim to retrieve semantic information at a higher-level of abstraction than a syntax tree, principally motivated by information granularity. This means unlike SRL, we are not specifically concerned with the 'action' of every predicate (target verbs) in the sentence. On the contrary, what interests us is the *granularity* of the semantic information, i.e. whether we can represent a word as a *class* or a *system*, as detailed in Section 3. In addition to that, SRL is incapable of finding relations between multiple actions in the sentence [65], an issue that *strata* can alleviate.

We deal with the problem of noise (varying query constructions having same intent) at the level of cognitive semantics, by making use of the concept of semantic *strata*. According to existing cognitive psychology, *strata* are used by the human psyche to process and store semantic information structures in the brain [57]. *To the best of our knowledge, computationally modeling semantic strata borrowed from the domain of cognitive psychology has not been previously used in semantic query understanding in the domain of natural language processing*.

6.3.1 Natural Language Parsing

We first describe the data and illustrate the tree-like subnets. Then we present metrics, benchmarks and results of two different experiments performed on three NL query datasets. The first experiment compares results of the cognitive canonicalized subnet to traditional POS-based subnets [46]. The second experiment compares the semantic coherency of the words connected in the canonical subnet pattern to SRL parse trees using graph ontologies.

Data. We test our model on each of these three datasets: (a) The TREC 2011 (TREC) web topic dataset has 50 topics [67]. Each topic has 1-7 queries associated with it. All queries within a topic resemble similar search intent. There are a total of 243 queries in the TREC topic dataset. 77% of the queries in the TREC dataset have 11-14 words. (b) The Microsoft Question Answering Corpus (MSQA), which is aimed at querying documents belonging to the Encarta-98 encyclopedia [66]. There are 1365 usable queries in this dataset and 85% of the queries have 5-10 words. (c) The last dataset consists of ~ 3400 raw search query feeds collected from a commercial web search engine (denoted as 'WSE'). Queries containing 4-20 words are chosen for evaluation. The distribution of average number of words per query for the three datasets is shown in Fig. 65.

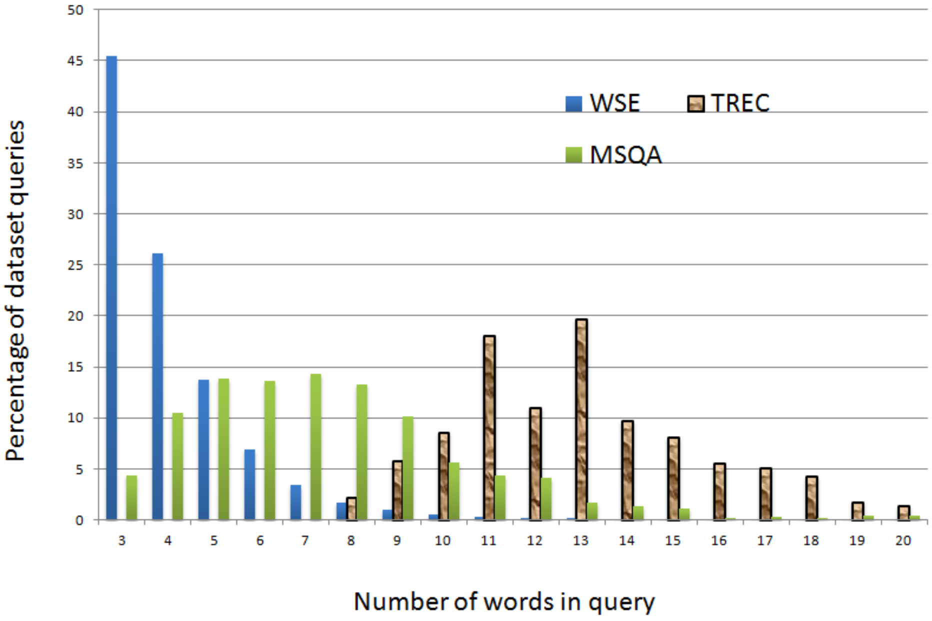


Figure 65: Distribution of number of words per query in the three datasets

The three datasets represents gradually rising levels of challenge in terms of query construction diversity, number of words in query and interpretability, with TREC being the least diverse and WSE being the noisiest. For experimenting on each dataset, we use 60% of the instances of the dataset for training and the rest 40% for testing.

Query Subnets. Table 12 shows an example of machine generated query subnet as a result of the proposed approach. We only visualize *units, class* and *system* tagged words as vertices in the final query subnet. *Relations* are used to connect the rest of the *strata*-tagged words. Other techniques (like stemming and stop-word removal) found commonly in NL toolkit are used to improve the visualization of the subnet. Thus, the canonical subnet usually consists of fewer vertices than the number of words in the original sentence.

Several other small optimizations are implemented: (a) we collapse consecutive *units* into a single *unit* when creating the subnet. (b) We use a simple root selection algorithm: when only *relation* words are found connecting two chunks $\mathcal{C}_i , \mathcal{C}_j \in \mathcal{C}$, we search $\mathcal{C}_{j+1}$ for *units* or *classes*. If $\mathcal{C}_{j+1}$ lacks a *unit* or *class*, we search $\mathcal{C}_{i-1}$instead. For example, in the query TREC#43 (Table 12) from TREC dataset, '*various TV'* and '*movie adaptations'* are connected by the conjunction '*and'*. Therefore, we search in $\mathcal{C}_{j+1} =' of\ The\ Secret\ Garden'$ and since we find a sequence of two *units* ('Secret', 'Garden'), we collapse it to a single *unit* and represent it as root.

We used annotators to hand label the queries in the datasets to build query subnet trees. The inter-annotator agreement on subnet structure was 72.3%. Disagreements were limited to just 1 node position in 82% disagreed cases. Thus, we consider this hand labelled set as the gold standard for comparing the machine generated subnet.

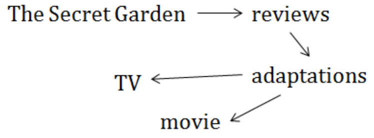
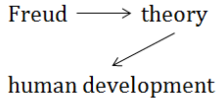
Query ID	*Query*	*Canonical Subnet*
TREC # 43	Find reviews of the various TV and movie adaptations of The Secret Garden	
MSQA # 812	What was Freud's theory on human development.	

Table 12: Examples of queries and their subnets

Cognitive Canonicalized Subnets vs. Traditional POS-based subnets.

Since our output (query subnet) is a tree where each node belongs to the set of query words, a 'tree-likeness' metric is essential to judge quality of results produced in terms of *structure*. We use *Consistency Index* (*CI*) as a metric to judge the quality of the subnet generated [68]. *CI* was first used in the field of computational phylogenetic, where it is used to study the evolutionary relatedness among groups of organisms. It estimates the structural similarity between two trees *T1* and *T2*. Mathematically, *CI* can be defined as:

$$Consistency\ Index\ (CI)$$
$$= node\ position\ matches\ between\ T1\ \ and\ T2/\#\ nodes\ in\ T2$$

where, *T1* represents the query subnet tree generated by a machine algorithm, *T2* is the query subnet tree of the gold standard and # represents the number of nodes. Thus, *CI* measures the number of correct node position matches of the machine generated output w.r.t the number required for best match. In [46], the authors evaluate their subnets using simple measures like 'nodes correctly resolved' or 'semi-correctly resolved'. However, we believe that *CI* captures the effect of *structural relatedness* more intuitively. Table 13 lists the average *CI* values obtained for various datasets for the proposed approach and the comparison benchmarks

Benchmarks. We compare our proposed cognitive model (*formNet*) against 3 benchmarks. As an external benchmark, we test our model against (a) the POS based approach introduced in [46] for generating subnets from query sentences (called *posNet*). We also compare our performance against: (b) a non-*strata* CRF (denoted as *nfCRF*) used in [62], whose features are based on POS only (not *strata*), and (c) a non-chunked version of our model (denoted as *noChnk*), to compare the gain due to semantic *strata* vs. gain due to chunking.

We measure the average CI for queries in each dataset with our technique (denoted as '*formNet*') against the benchmark techniques described above. Results are reported in Table 13. For each dataset, we provide a detailed bar graph describing percentage of queries that produced outputs in some particular CI range. For the average CI measures in Table 12, we include results of parse tree generated by SRL. The SRL parse trees is generated using the NLP software described in [69], which processes an input query and generates a SRL parse tree.

TREC dataset. Fig. 66 shows that *formNet* achieves *CI*=1 for 63.1% queries. In fact, only 9.2% of the queries produced a *CI* < 0.5 using *formNet*. The benchmark *posNet* does considerably well in retrieving half the query subnet pattern (*CI*=0.5), but fails to generate the exact human annotated subnet pattern (*CI*=1) for almost 80.2% queries. Net improvement of *formNet* over *posNet* benchmark is 52.5%. The performance of *noChnk* is significantly better than *nfCRF* as shown in Table 13, indicating that use of *forms* in CRF is more important than using a standard CRF.

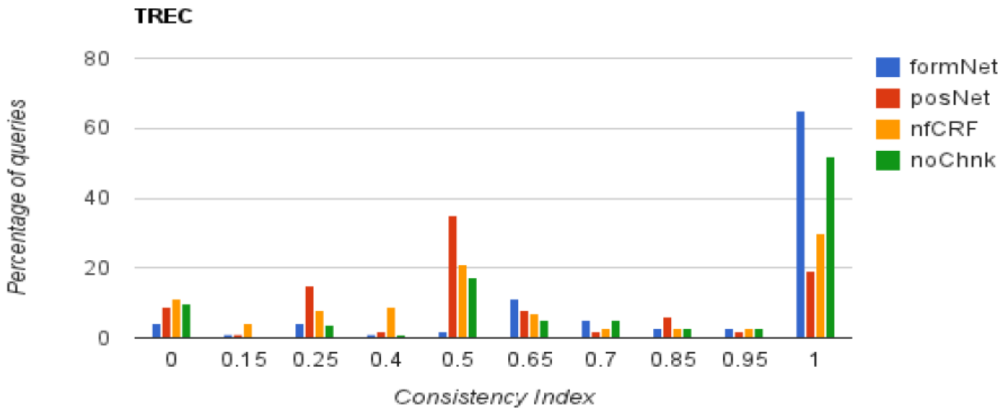


Figure 66: Consistency Index results on TREC dataset

MSQA dataset. Table 13 shows that *formNet* provides an average CI=79.5 for MSQA queries whereas the benchmark *posNet* produces an average CI=53.6. This signifies ~ 49% improvement in performance. Fig. 67 shows that *formNet* can retrieve 55% queries with perfect match and produces a *CI*>0.5 for 85% queries in the dataset. In contrast, the benchmark *posNet* could only produce *CI*>0.5 for 38% queries. TREC queries are grammatically richer than MSQA; therefore a drop in overall performance is expected when evaluating MSQA. Interestingly, *strata* seem to be playing a stronger role in MSQA, since a traditional CRF performs poorly in this case.

WSE. WSE queries are most diverse in construction and number of words. In Fig. 68, we see that performance is reduced for all techniques, but *formNet* still performs better than *posNet* by 51.86%. Observe that *noChnk* performs worst for TREC when compared to *formNet* than for any other dataset as indicated in Table 13 (difference between average CI for *formNet* and *noChnk*). This reaffirms our previous observation from the query data: TREC queries consist of longer sequence of words. Chunking has relatively larger effect on performance improvement for TREC, but not so much for MSQA or WSE queries that are shorter.

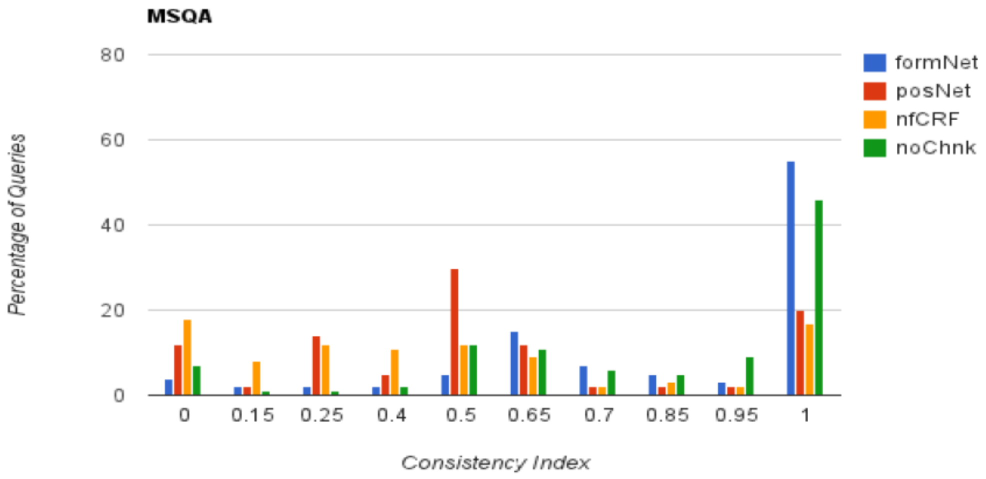


Figure 67: Consistency Index results on MSQA dataset

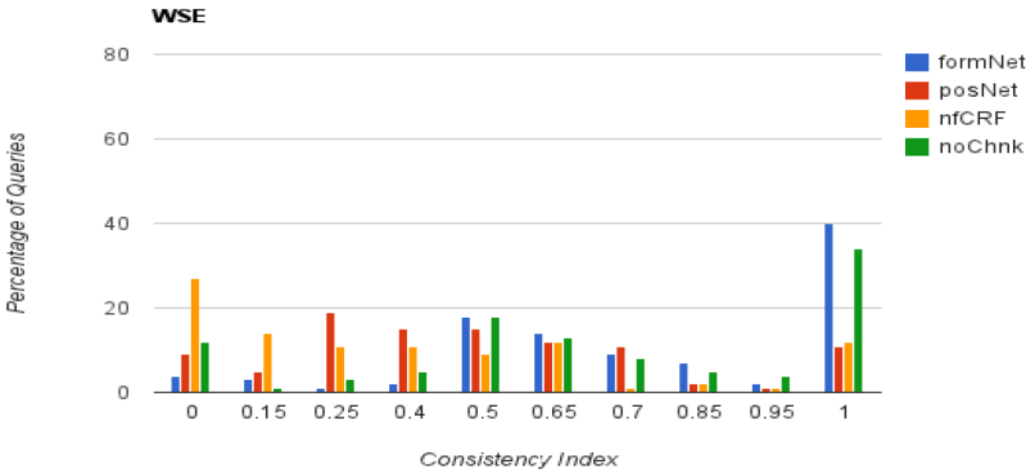


Figure 68: Consistency Index results on WSE dataset

Tree	**TREC**	**MSQA**	**WSE**
posNet	54.8	53.6	48.2
nfCRF	58.6	43.4	36.1
Canonical Subnet	*83.6*	*79.5*	*73.2*
noChnk	74.0	77.3	68.4
SRL	68.9	66.0	51.6

Table 13: Average consistency indexes for benchmarks on different datasets

Our results in Table 13 suggest certain interesting points: (1) We notice that *formNet* outperforms *nfCRF*, which implies that the boost in performance is not due to the CRF model specifically, but due to the feature functions consisting of semantic *forms*. (2) Also, *formNet* does not perform substantially better than *noChnk* for MSQA and WSE datasets, whereas no chunking for TREC significantly deteriorates performance. This indicates that chunking has a stronger impact in TREC, a dataset where 77% queries have more than 11 words (Fig. 65). In comparison, only ~10.8 % queries in MSQA and 7.6% queries in WSE have more than 9 words. (3) SRL performance is significantly degraded if the grammar is improper (the drop in performance in WSE for SRL is much more than the drop in case of Canonical Subnets or noChnk). This is due to the noise in WSE query constructions and SRL is sensitive to the noise.

Cross-dataset Testing. In cross-dataset testing, we train on one dataset and test on another. Our intuition behind cross dataset testing is that different datasets differ in query structure, context and the length of query (Fig. 65). Thus, to ensure robustness to different training environments, we perform cross dataset testing. Here, we report *formNet* performance by training on one dataset and testing on another (read TRAIN_TEST). The average CI achieved by *formNet* in cross-dataset testing (Fig. 69) is as follows: TREC_MSQA: 0.53, TREC_WSE: 0.44, MSQA_TREC: 0.68, MSQA_WSE: 0.58. We can observe that cross dataset testing provides best results when we train on MSQA and test on TREC. This is potentially due to the fact that the TREC dataset query structures are quite limited in construction; such constructions are contained within queries of MSQA. Performance is worst when we train on TREC and test on WSE. This is potentially due to the diverse and noisy queries in WSE, which are not captured during limited training over TREC. Nevertheless, for MSQA_WSE, our technique retrieves query subnets with CI > 0.5 in 73.1% cases and CI > 0.75 in 33% cases, suggesting robustness of *formNet* to web scale.

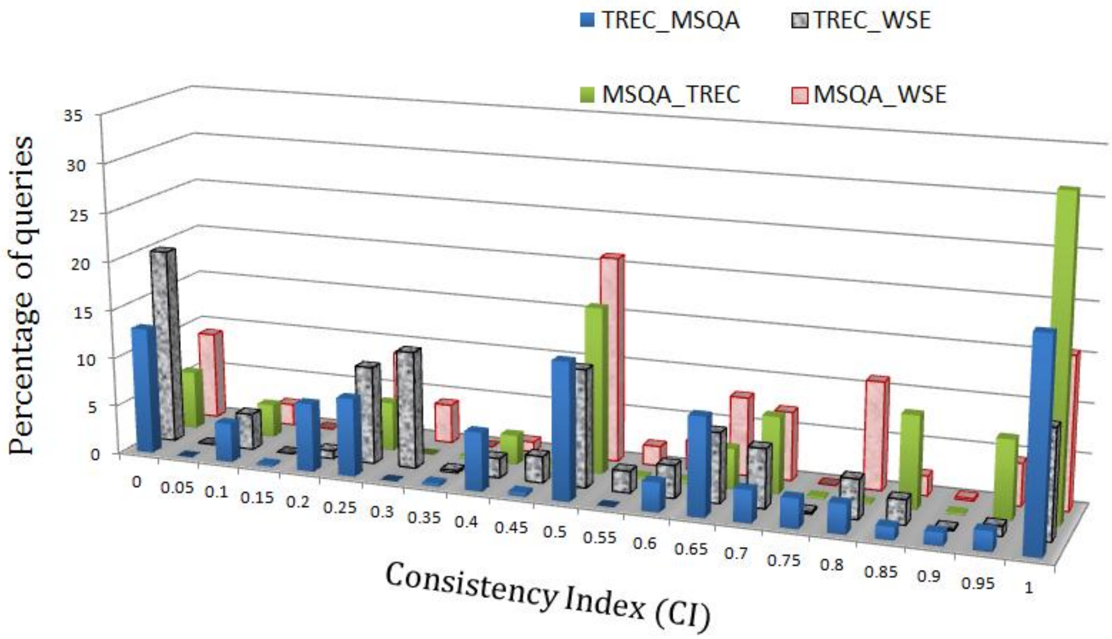


Figure 69: Cross-dataset testing on the 3 datasets

6.3.2 *Query Canonicalization*

In order to compare the cognitive canonicalized subnets to the SRL parse trees, we first need to generate the latter. This is achieved using a state-of-the-art NLP software described in [69], which processes an input query and generates a SRL parse tree. In Fig. 71, the two trees (SRL and cognitive canonical subnet) are shown for the query : "*Find reviews of the various TV and movie adaptations of The Secret Garden"* from the TREC dataset.

Our target is to determine which tree is more semantically coherent when they act as a canonical form for graph databases. The notion of semantic coherence refers to the semantic similarity between all pairs of nodes in the tree. This similarity can be obtained using an ontology, as is described in [70]. Given the nodes in the trees, we first detect if they exist as vertices in the ontology graph. For words that do not exist in ontology graph (e.g., 'of' for SRL parse tree), we eliminate them from analysis. For the set of nodes that do exist in the network, we calculate the closeness centrality among the nodes [71]. The reasons for choosing closeness centrality is described below.

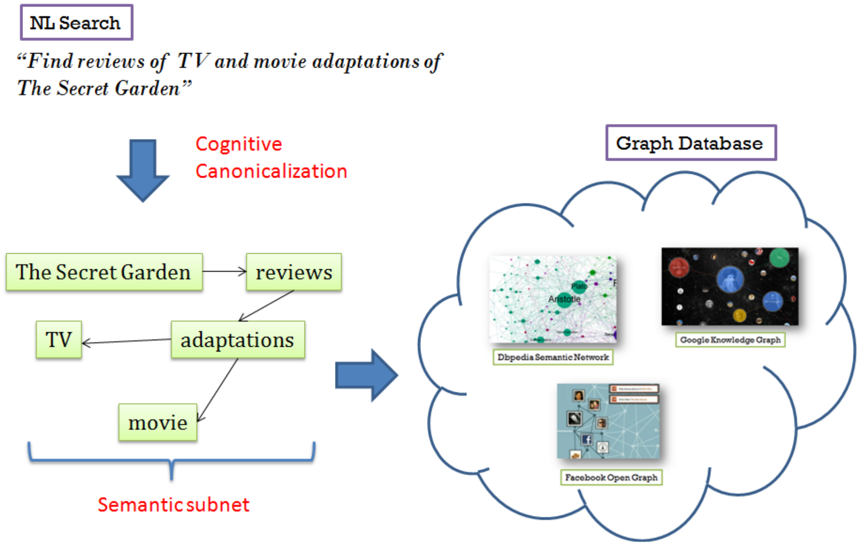


Figure 70: Idea of cognitive canonicalization of NL queries

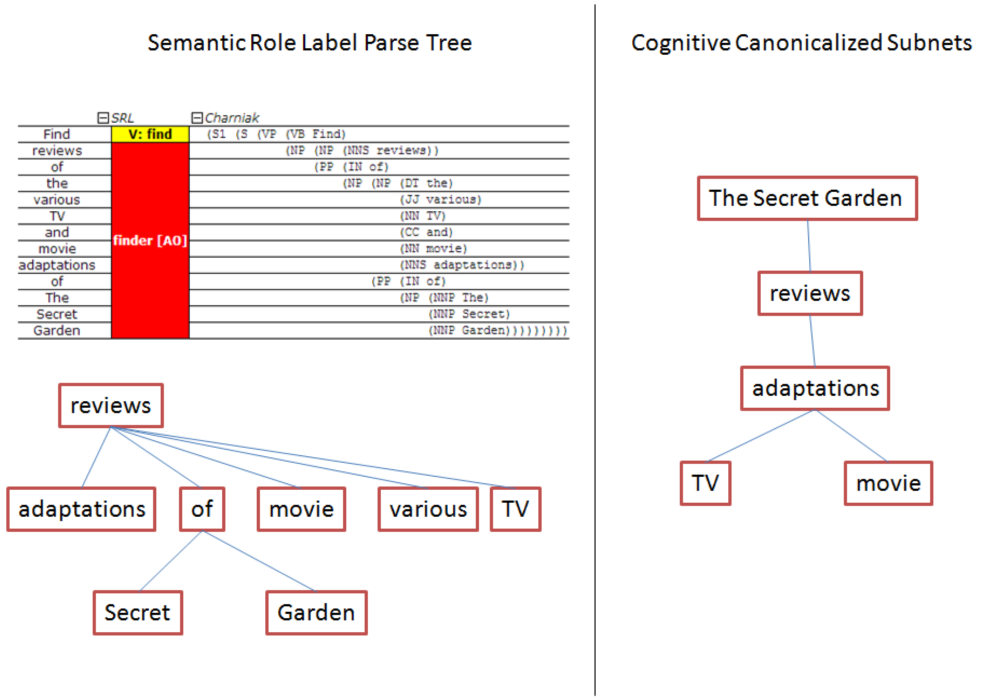
In a graph, the farness of a node is defined as the sum of its distances to all other nodes. Closeness is the inverse of farness. Consider that we spot all the tree nodes in the ontology graph. Then, we can calculate the farness of any tree node with respect to the other tree nodes. Note that an ontology is a graph where similar concepts are connected by edges. Therefore, the further two nodes in the ontology graph, the lower is their semantic similarity. In other words, the farness of the nodes determines their semantic dissimilarity. Conversely, the closeness of the nodes signifies their semantic similarity. As mentioned earlier, this closeness is a good measure of the semantic coherency of the tree.

Let us measure the closeness of a tree node i (call it focal node) with respect to other nodes of the tree. According to [71], the closeness of the focal node is given by :

$$closeness(i) = \frac{1}{\sum_{j \neq i} d(i,j)} \qquad (40)$$

where *d(i,j)* is the shortest distance between nodes *i* and *j,* where *j* is any node in the tree other than the focal node. Thus, we can calculate the semantic coherency of the subnet (*S*) with *n* nodes as :

$$semantic\ coherence(S) = \sum_{k=1}^{n} closeness(k) \qquad (41)$$



Semantic Role Label Parse Tree

	⊟SRL	⊟Charniak
Find	V: find	(S1 (S (VP (VB Find)
reviews	finder [A0]	(NP (NP (NNS reviews))
of		(PP (IN of)
the		(NP (NP (DT the)
various		(JJ various)
TV		(NN TV)
and		(CC and)
movie		(NN movie)
adaptations		(NNS adaptations))
of		(PP (IN of)
The		(NP (NNP The)
Secret		(NNP Secret)
Garden		(NNP Garden))))))))

Cognitive Canonicalized Subnets

Figure 71: SRL vs. Cognitive Canonicalized subnets

Eventually the purpose of a subnet is to serve as an NL interface to graph databases, converting the NL query into the internal graph representation. This requires experimentation on how well the canonicalized tree can 'fit' into a graph database compared to an SRL tree. A higher magnitude of semantic coherence of the subnet indicates a better fit. CI is not the appropriate metric in this scenario, because it compares to human annotation and not semantic similarity represented by graph database.

Results. In Table 14, we provide results of the semantic coherence averaged over all queries in each dataset for three methods, the POS-based parse tree, the SRL parse tree [69] and our proposed cognitive canonical subnet tree. We assume each edge of the ontology has weight 1. Thus, two nodes *i* and *j*, separated by at least three hops, will have *d(i,j)*= 3. For each query subnet/parse tree, we calculate the closeness of each node in tree using Eq. 40 and then the semantic coherence of the entire tree using Eq. 41. The semantic coherence is averaged over all the queries in the dataset

Tree	**TREC**	**MSQA**	**WSE**
POS Parse	0.041	0.054	0.027
SRL Parse	0.055	0.072	0.048
Canonical Subnet	*0.066*	*0.078*	*0.081S*

Table 14: Avg. Semantic Coherence for the three datasets using various methods

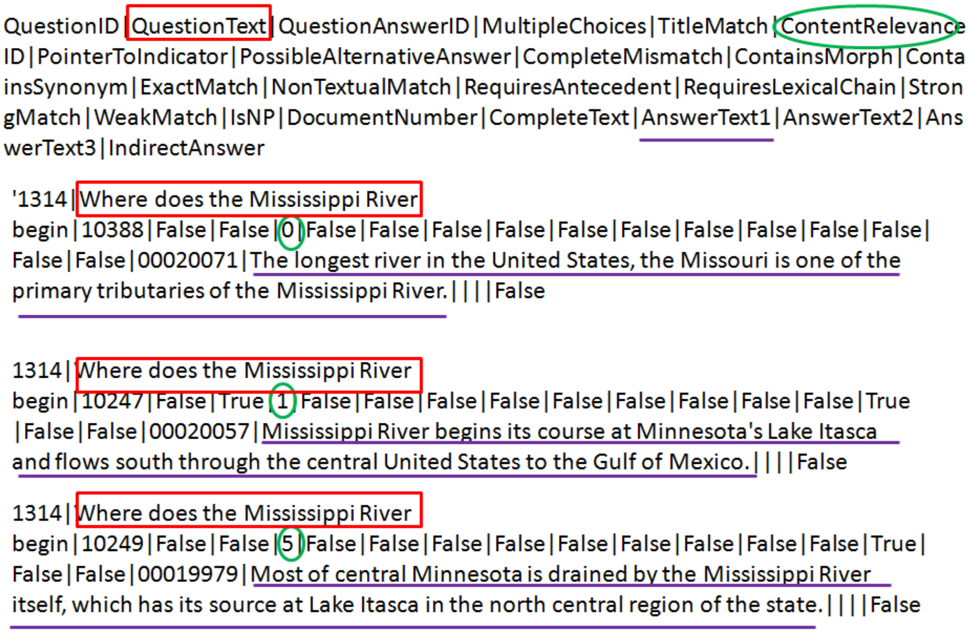
Some interesting observations can be made from Table 14. Firstly, the canonical subnet outperforms the other techniques for each dataset. Secondly, for both POS and SRL, performance deteriorates as : MSQA> TREC > WSE. However, for the canonical subnet, performance degrades as: WSE>MSQA>TREC. The second observation is potentially due to the nature of queries in the three datasets. MSQA queries have single frames [65], allowing each technique to perform better in MSQA because frame-relation modeling is not required. TREC has multi-frames with 'and' clauses, resulting in more complex natural language constructions. Recall that WSE has least number of words per query compared to the other datasets (Fig. 65). Thus, they are noisier than MSQA or TREC. POS tries to label the words in WSE but performs poorly due to the irregular grammar. Although SRL does better than POS, internet users often use keywords without grammar or predicate-action verbs, confusing the SRL parser. Cognitive subnets, on the other hand, focus on the granularity of the words when constructions are noisy in the WSE queries. Thus, cognitive subnets will not try to connect all the words unlike SRL or POS parse trees if the construction is noisy. This allows for far lesser words in the tree, reducing the closeness and boosting the semantic coherence.

Extrinsic Evaluation and Ranking Results. Documented in this section are the results of extrinsic evaluation and performance of the proposed approach in searching and ranking documents corresponding to the MSQA dataset. For these experiments, we took the subnet of each query (generated by the various benchmarks in addition to our approach) and ran a iterative deepening DFS search [72] on the semNet to retrieve relevant expanded nodes for the query sentence. Each expanded node is a concept node which was in path between two neighbor nodes of the subnet. Remember that neighbors in the subnet could be many hops away in semNet. Following this, we aim to retrieve documents with the expanded query words (obtained from semNet) in addition to the original query words.

The MSQA dataset has a document set of more than 37,000 documents from the Encarta 98 Encyclopedia in addition to 1300 queries mentioned in Section 6.2. On average, each query has approximately 7 related documents. Each document is given a relevancy scores from 0-5, based on how closely one or more sentences in the document accurately answers the query. Shown below is Fig. 72 is an example query and relevant documents for the MSQA dataset.

The expanded list of words is retrieved for each of the baseline cases: (1) subnet from *posNet,* (2) subnet from *noCRF*, (3) subnet from *noChnk*, (4) *SRL* parse tree and (5) Explicit Semantic Analysis (ESA) [73], which uses a vectorial representation of text and uses Wikipedia. However, ESA does not use network hierarchies like *strata* in parsing.

These methods are compared against our proposed canonical subnet based on semantic *strata*. In addition, we use another baseline search engine called Lucene, which is a popular open source search software and powers many web applications, including Twitter's real-time search. Lucene's ranking uses a combination of the Vector Space Model (VSM) and boolean model [74].



QuestionID|QuestionText|QuestionAnswerID|MultipleChoices|TitleMatch|ContentRelevanceID|PointerToIndicator|PossibleAlternativeAnswer|CompleteMismatch|ContainsMorph|ContainsSynonym|ExactMatch|NonTextualMatch|RequiresAntecedent|RequiresLexicalChain|StrongMatch|WeakMatch|IsNP|DocumentNumber|CompleteText|AnswerText1|AnswerText2|AnswerText3|IndirectAnswer

'1314|Where does the Mississippi River begin|10388|False|False|0|False|False|False|False|False|False|False|False|False|False|False|False|00020071|The longest river in the United States, the Missouri is one of the primary tributaries of the Mississippi River.||||False

1314|Where does the Mississippi River begin|10247|False|True|1|False|False|False|False|False|False|False|False|False|True|False|False|00020057|Mississippi River begins its course at Minnesota's Lake Itasca and flows south through the central United States to the Gulf of Mexico.||||False

1314|Where does the Mississippi River begin|10249|False|False|5|False|False|False|False|False|False|False|False|False|True|False|False|00019979|Most of central Minnesota is drained by the Mississippi River itself, which has its source at Lake Itasca in the north central region of the state.||||False

Figure 72: Example query and relevant documents for the MSQA dataset. Box marks the query. Underlines mark the answer sentence from the relevant document. Circles mark the context relevancy of the answer given the query. Content relevance of 0 represents 'no judgment made', 1 means 'exact answer', 3 means 'off topic', 4 means 'on topic, off target', and 5 means 'partial answer'. Document ranking proceeds as 1>5>4>3>0.

Since, we are not only concerned with document retrieval but also ranking of the retrieved documents, our evaluation metric is Normalized Discounted Cumulative Gain (NDCG) instead of F-Score. NDCG is a standard metric to evaluate search ranking results [75]. For a query q, $NDCG@K$ of a ranking of documents retrieved for query q is :

$$\frac{1}{N_K^{(q)}} \sum_{k=1}^{K} \frac{2^{r_k} - 1}{\log\,(k+1)} \tag{42}$$

where r_k is the relevance level of the kth ranked document, and $N_K^{(q)}$ is a normalization factor such that the best ranking gets $NDCG@K = 1$. For our experiments, we report the $NDCG@5$ and $NDCG@10$.

Interesting observations from results reported in Table 14 include: (a) Comparison between NDCG@5 vs. NDCG@10 tells us that over larger space of documents to be returned, the proposed approach will outperform ESA significantly. (b) Not using *strata* or vector model significantly deteriorates performance as reflected by *posNet*. (c) The *canonical subnet* will perform better compared to ESA with chunking, but performance will degrade to ESA level without it. (e) The *canonical subnet* will always perform better than SRL, even without chunking. (d) Lucene outperforms SRL parse when more documents are to be returned. (e) ESA and Lucene both use the vector space model, but ESA leverages Wikipedia which significantly improves its performance compared to Lucene. Overall, leveraging semantic *strata* allows the *canonical subnet* to edge out the existing techniques.

Method	**NDCG @ 5**	**NDCG @10**
posNet	0.635	0.591
noChnk	0.871	0.799
Lucene	0.790	0.760
ESA	0.872	0.788
SRL parse	0.812	0.730
Canonical Subnet	*0.956*	*0.940*

Table 15: Average NDCG results for MSQA dataset

Discussion. Several papers on computational cognitive psychology dwell on the fact that cognitive psychology models cannot be purely verified on the basis of behavioural experiments [55]. For researchers in the domain of NLP, a fascinating possibility is to model cognitive techniques computationally and test their robustness to noise in NL. Natural languages are undeniably imprecise, especially in the realm of semantics. The primary reason of this imprecision is the fuzziness of class boundaries [61]. Surprisingly, robustness to imprecision is often achieved by slightly relaxing the rigidity imposed by lexical grammar, by means of parsing at a higher abstraction than POS. In some ways, the case is analogous to robust, scalable image/video transmission in the face of a noisy channel, where lower-resolution data is usually transmitted if the connection is weak or the channel is noisy.

Essentially, there exists a hierarchy in every semantic network, exemplified by the network's degree distribution. This hierarchy identifies the generality of a concept node. Concepts are the essence of semantic information, and are granular in nature. Therefore, *strata,* which means levels, indicates this surreal hierarchy, which can be employed to understand the granularity of semantic information.

In this paper, we reproduce the structure-of-intellect model of cognitive psychology computationally. Exploring the various interactions among the semantic *strata* provides insights into the higher level abstract (conceptual) connection among the query words, which is subsequently exploited in generating the canonical subnet. The canonical form is consequently searched in a semantic ontology. Our proposed approach comprehensively outperforms existing techniques for query subnet extraction and produces more semantically coherent canonical parse trees compared to state-of-the-art NLP techniques like SRL.

CHAPTER 7: LEARNING COLLECTIVE ATTENTION MODELS FOR SOCIAL MEDIA

The vast quantity of information shared in social networked spaces has brought us to an age of attention scarcity, where getting users to be attentive to a message is not a given. It is a limiting factor in the consumption and spread of information. Understanding what captures the collective attention amongst a community of users is invaluable to applications such as product marketing, advertising and social or political campaign organization. Many scholars have analyzed how information spreads in social networked spaces, however few studies provide a quantitative method to model and predict attention over time within dynamic social networks.

In this chapter, we discuss the *Attention Automaton*, a probabilistic finite automata that can evaluate the collective amount of attention given to a topic by a community of users who are either grouped geographically or through common interests (followers of a given account) on Twitter. We identify two key factors that drive collective user attention: (1) the *volatility* of the community, i.e. frequency of change of posted topics, and (2) the selective categorical affinity of the group towards certain topics. Our results, which are based on a 6-month dataset of Twitter trending topics across 111 geographic regions and audience trends of approximately 50 accounts show that the *Attention Automaton* can predict audience reception of impending trends based on selective category types and the inherent properties of the community.

Background

Human attention is the mental 'spotlight' on a stage full of information. The idea that attention is a scarce commodity was first laid out by Herbert Simon [76]. However, it was Davenport et al. who first indicated that attention precedes activity on the web [77]. Given the overload of information in cyber space, search engines and recommendation systems attempt to learn from our interactions (click through data etc.) to identify and predict resources that users would be more attentive towards. Understanding the attention of online communities can be very useful for advertising leads, targeted advertising, marketing and understanding information diffusion in online social networks.

Attention is captured by the behavior of social network nodes in the face of competing choices of interaction [78]. It has been found that attention is the primary barrier for social contagion and information propagation in online social networks (Hodas 2012). The allocation of attention among a set of items in social news website Digg is log-normally distributed [80]. Lehmann et. al. found that the evolution of hash tags popularity in Twitter follows discrete classes, indicating user groups are attentive to selected categories of information [81]. The balance of attention dedicated to these categories is a relatively stable property over time [82]. Researchers have also shown that a combination of social network structure and finite attention is a sufficient condition for emergence of dynamics of social networks [83]. This makes attention modeling in social networks a vital prerequisite to predict future popularity and lifetime of trends.

Our model also makes use of probabilistic automatons, which are finite state machines. Finite state machines are fundamental to computer science. They are widely used as spelling checkers and Hidden Markov models. A probabilistic automata is a state transition system, consisting of a series of states, actions that can cause transition between states, and a probability attached to each potential transition from one state to another [79].

Since attention precedes online activity [77], it is pivotal to model attention of user communities in order to comprehend the fundamental differences in behavior between user groups, in other words, what makes them unique. There are two limitations in existing work in this domain: (1) Although social data mining reveals popularity and novelty of trends as a good indicator of attention patterns of users, it still does not help us quantify the collective attention shifts in communities or the categorical attention affinity that exists in users groups. Most importantly, it gives us few indications as to whether collective attention is at all *computable* (in terms of a model of computation) and whether we can predict the likelihood of a future trend to receive sustained attention. (2) Secondly, the dynamics of collective attention is substantially different from individual attention [83]. Our analysis shows that a collection of users bound together as followers of a given account or within close geographic proximity can play a big role in what becomes popular and receives attention.

Our research strives to address these limitations. We build a probabilistic automaton, called the *Attention Automaton*, showing that attention states in user communities of Twitter are computable in terms of a finite state machines. Moreover, unlike previous work, we focus on collective attention which is endorsed by the collective behavior of the inherent communities we are part of in the social network.

7.1 User Groups in Social Networks

Our research is based on two datasets containing Twitter trend data. The first data set includes Twitter trends based on geographical locations for approximately 7 months. The second data set contains trends for audiences who are subscribers/followers of some Twitter account for approximately 3 months. Let us describe the kind of user communities we see on Twitter and how they develop.

Twitter Trends

When a group of users on Twitter increasingly RT a message or tweet about some topic, then it is captured as a trend. We wrote a script that probes Twitter every 5 minutes and logs the TTL (Fig. 73) provided by Twitter for 111 geographical locations worldwide. This is essentially a time series, where each instance is of the form: {timestamp, location, [list of trends]}. Thus, each GT-TTL instance includes a list of 10 trends and resembles the top topics of discussion based on tweets coming from the specific geo-locations. We have this data from Nov. 2011 to June 2012.

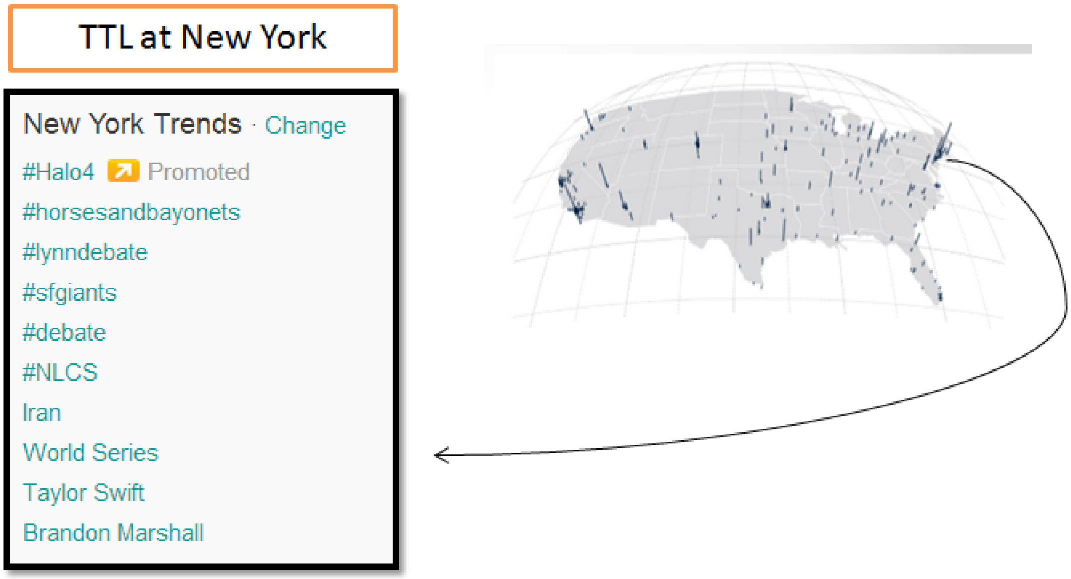


Figure 73: Example Trending Topic List (TTL) in New York on October 23rd at 8:15 AM - the morning after the 3rd presidential debate in 2012.

Audience Trends

We also collect tweets of followers for approximately 50 Twitter accounts (called brands hereafter). We maintain a diverse category list of brands, including *News* (@NYTimes), *Sports* (@ESPN), *Politics* (@CNNPolitics), *Gaming* (@IGN), *Entertainment* (@Miramax) etc. We follow a similar method to Twitter in detecting trends, i.e. based on how frequently a word appears in the collection of tweets. Thus, after pre-processing, we sort the most frequent words occurring in follower tweets for every brand and store it as BT. We update this every 2 minutes and therefore, our maximum granularity for BT is 2 minutes. The maximum granularity for GT is 5 minutes. Thus, the BT data set instances also resemble a time series in the brand audience world. Each instance is of the form: {timestamp, brand, [list of trends]}.

Our underlying assumption is that attention of a user group is characterized by the trends of that group, as they are derivative of the cumulative topics published by the group. Thus, the attention of user group NY is judged by the trends appearing in TTL of NY. Similarly, trends from audiences following @EA (Electronic Arts) account indicate behavior of user group of EA. We uncovered three key insights from these two datasets regarding collective attention in user groups. They are described as follows:

User Groups possess inherent attention shift tendencies. Different user groups have diverse (unequal) durations for which they can maintain attention on a particular topic. Rapid attention shifts are reflected by frequent changes to the TTL of the user group over consecutive time slots. We noticed that the TTL in some cities (e.g., St. Louis) remains fairly constant over multiple hours, whereas in other cities such as New York it changes every 5 minutes.

User Groups possess selective affinity to certain categories of trends. We also found that user groups in different cities and for different followers are disparately receptive to trends in various categories. For example, San Francisco has strong affinity to trends in *Gaming*, whereas Boston has strong affinity to trends in *Politics*. In a similar fashion, we show later how audiences of Pepsi are very attentive to *Entertainment* trends, especially Justin Bieber, whereas audiences of Burberry have near-zero affinity towards *Sports*.

User Groups react to real-world events based on a combination of their attention shift patterns and their selective categorical affinity. We chalked some of the major events over a period of 7 months synchronized with the data sets (Fig. 75). We found that it is possible to quantify the attention shift within user communities occurring in response to real world events. In other words and contrary to popular belief, it is not more difficult to force something to trend in bigger user groups such as New York compared to Tallahassee, Florida (comparatively smaller user group); provided we know what New

York user group has affinity towards and its attention shift tendencies i.e. given the right conditions, trends can break into New York TTL as easily as it does in smaller user groups.

Social networked spaces such as Facebook and Twitter emerged as platforms for connecting people who wanted to stay in touch, be heard, share information and track viewpoints. With an increasing number of users, brands and highly visible celebrities joining these services, there came an inevitable explosion in the amount of content readily available to users. As the threshold to publishing nears zero, getting users to be attentive is a limiting factor in our networked information ecosystem [80]. One cannot demand attention, or even expect it at a given point in time. It is a scarce commodity that must be earned [84].

We can quantify user attention within social networks by looking at the level of interest that a node (user) dedicates in managing its interaction with another node or group of nodes within the observed social network [82]. The interaction can be captured in different activities, such as 'liking' a Facebook status update, 'retweeting' a tweet or posting to a topic that is trending on Twitter [83]. For example, we can consider that when node *X* on Twitter retweets (RTs) a message *M* of another node *Y*, then *X* was attentive to *Y* or to the content of *M*. Similarly, if node *X* tweets about a topic that is trending, then we can claim that *X* was attentive towards that trend. By extension, when a group of users RT a certain tweet, they display *collective attention* [81]. This group of users could be geographically co-located, or followers of the same user or part of a networked community.

Motivation and Scope

Understanding the dynamics of collective attention can help content producers and intermediaries better manage information flows under the constraint of human attention. It also helps with the judgment of what, when and why some trend becomes popular, which has great relevance to monetization of online content. Social advertising utilizes a user and their community within social networked spaces, attempting to accurately target contextually relevant personalized ads. 'Promoted Content' on Twitter is a good example of targeted social ads. In news and media, it is key to judge which news topics will users be more receptive towards, potentially based on their location or the broadcast network they watch (follow). This necessitates prior knowledge of the facets that capture group attention.

Previous research has attempted to capture the dynamics of popularity and information diffusion in social networks to get a sense of what receives user attention. Interesting findings from these papers show that attention is the deciding factor in information spread [85], that there are specific categories which potentially receive more attention [81]and that these categories remain relatively consistent over long periods of time [82]. However, most of this work aims to understand individual user attention and misses the insights provided by the larger community. If the attention of communities of users is captured in a computing model (e.g., in terms of finite state machines) that can represent the dynamics of collective attention, we can attempt to predict future collective behavior.

This chapter discusses the development of a probabilistic automaton that aims to capture the dynamics of collective attention among user groups on Twitter, who are either geographically co-located or co-followers of the same Twitter account. Every state in the automaton is a list of trending topics from the user group. The trending topics list (TTL) of an user group, shown in Fig. 73, alters with time in response to user tweets (discussed further in related work). This phenomenon is captured by the *Attention Automaton* as it transitions from one state to another, mirroring the changes in the TTL over time (Fig.

75). The actions that cause the state transitions is a set of competing trends that are trying to break into the TTL at any given moment (impending trends). When a trend breaks into the TTL, it forces the automaton to jump to a new state, as the list changes. The probability of transition depends on two key factors: (1) the attention shift tendency of the group, and (2) the selective categorical affinity of the group towards certain trends. The attention shift tendency of the user group is modeled using a metric we call *Volatility*, which represents how frequently trending topics within the group change over time. We use a Levenshtein distance [86] based metric to formulate the volatility of the user group. The selective categorical affinity is obtained from the past history of topics that trended in the user group and their respective categories.

Our results reveal interesting information: (1) Collective attention is mainly driven by two key opposing forces: volatility vs. categorical affinity of attention. (2) The collective attention of user groups over time on Twitter can be modeled as a probabilistic automaton. This automaton has predictive power over future states given a time series of impending trends. (3) The *Attention Automaton* can capture Twitter community reactions to real world events.

7.2 The Attention Automaton

We first provide an overview of the automaton in terms of the Twitter ecosystem. We can then explain specifics of measuring attention shift, measuring categorical affinity and formulating the transition probability for the automaton.

A probabilistic automaton consists of a set of states, a series of actions and a transition probability attached to each potential jump from one state to another based on the action. Let $Distr(X)$ denote the set of all probability distributions over *X*. Then the Attention Automaton (*A*) consists of four components:

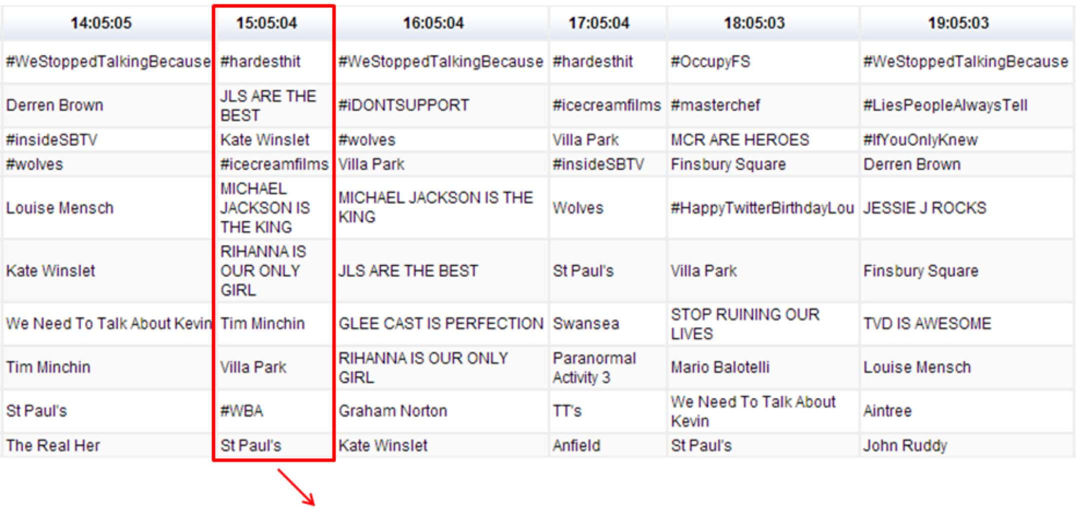
1.A set S_A of states.

2.A non-empty set S_A^0 of start states.

3.An action signature $sig_A = (E_A, I_A)$ consisting of external and internal actions respectively. We assume that E_A and I_A be mutually disjoint and the complete set of possible actions is $Act_A = E_A \cup I_A$.

4.A transition relation $\Delta_A \subseteq S_A \times Act_A \times Distr(X)$

Notice that for each user group, we possess a time series of TTLs. Fig. 74 shows one such time series TTL data for London on Oct. 22, 2011 between 2 PM and 6 PM. Each time-stamped TTL is a state in the automaton. As time passes, the automaton moves to another time-stamped TTL state, based on the new trends that replace some old trends in the latter TTL. We represent the TTL state at time *t* by γ_t. It is also important to note that time is deterministic. At a given instant, the automaton can be in only one state. The start state for all actions on or after time *(t+1)* is γ_t.



14:05:05	15:05:04	16:05:04	17:05:04	18:05:03	19:05:03
#WeStoppedTalkingBecause	#hardesthit	#WeStoppedTalkingBecause	#hardesthit	#OccupyFS	#WeStoppedTalkingBecause
Derren Brown	JLS ARE THE BEST	#iDONTSUPPORT	#icecreamfilms	#masterchef	#LiesPeopleAlwaysTell
#insideSBTV	Kate Winslet	#wolves	Villa Park	MCR ARE HEROES	#IfYouOnlyKnew
#wolves	#icecreamfilms	Villa Park	#insideSBTV	Finsbury Square	Derren Brown
Louise Mensch	MICHAEL JACKSON IS THE KING	MICHAEL JACKSON IS THE KING	Wolves	#HappyTwitterBirthdayLou	JESSIE J ROCKS
Kate Winslet	RIHANNA IS OUR ONLY GIRL	JLS ARE THE BEST	St Paul's	Villa Park	Finsbury Square
We Need To Talk About Kevin	Tim Minchin	GLEE CAST IS PERFECTION	Swansea	STOP RUINING OUR LIVES	TVD IS AWESOME
Tim Minchin	Villa Park	RIHANNA IS OUR ONLY GIRL	Paranormal Activity 3	Mario Balotelli	Louise Mensch
St Paul's	#WBA	Graham Norton	TT's	We Need To Talk About Kevin	Aintree
The Real Her	St Paul's	Kate Winslet	Anfield	St Paul's	John Ruddy

$\gamma_t = Trending\ topic\ list\ at\ time\ instant\ t$

Figure 74: GT-TTL in London on Oct. 22, 2011. Only hourly TTLs are shown here

A jump from one state to another defines a transition. Each transition is brought upon by an action. The action is a set of impending trends that are attempting to break into the TTL list at time (t+1) so that they can be part of γ_{t+1} (see Fig. 75). Thus, when trend r successfully breaks into the TTL, it fundamentally changes the content of the γ_{t+1} compared to γ_t. Changes in the TTL is represented by the automaton jumping among states based on the action and a transition probability.

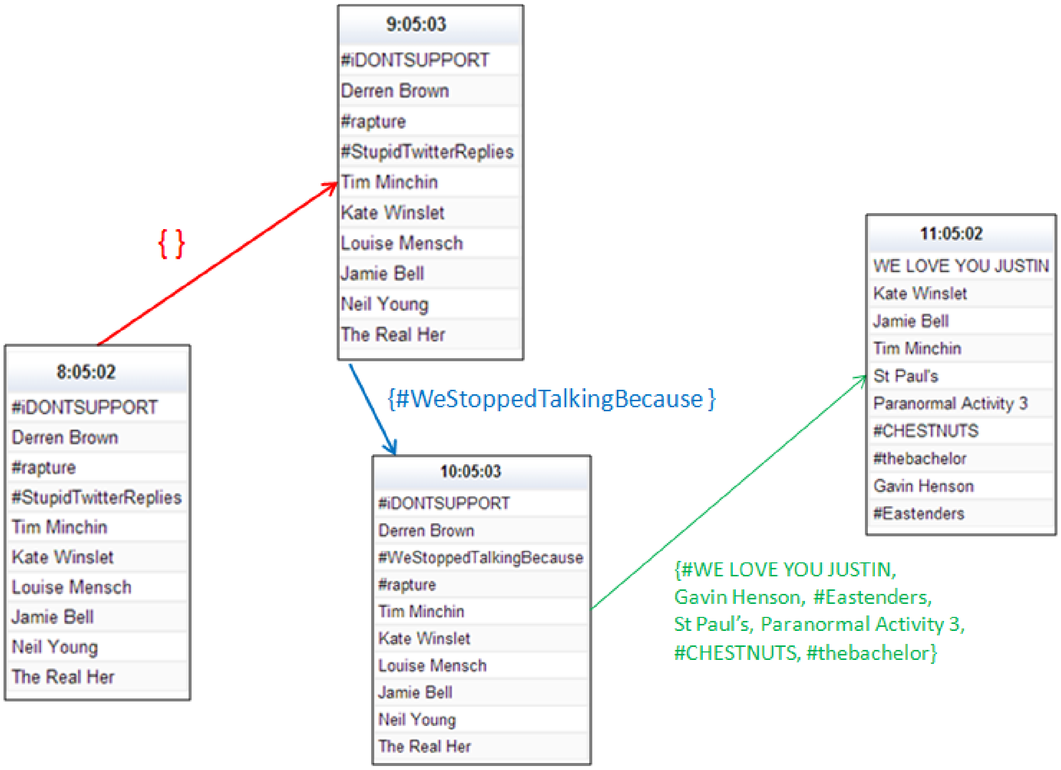


Figure 75: TTL changes in London on Oct. 22, 2011 between 8-11 AM. Only hourly TTLs are shown here

The final component of the automaton is the probability of the transition between two states. As mentioned earlier, we found that this probability depends on two factors: (1) the attention shift tendency of the user group, and (2) the categorical affinity of the user group. We first discuss our approach in modeling both these phenomena. Following that, we discuss combining the two attention factors to produce the transition probability.

Modeling Attention Shifts

We devise a metric called 'Volatility' to measure the tendency of attention shift over time for a user group. Since we represent content that is receiving attention in an user group based on the TTL (which is ranked) in one time slot, measuring the difference between the TTLs in consecutive time slots is an acceptable measure of attention shift.

Difference between consecutive TTLs.

The difference between consecutive time slot TTLs is basically the edit distance between the two TTLs. In other words, consider each TTL to be a string of trends. Then the difference of two TTLs can be visualized as string edit distance. We use the Levenshtein distance to measure the difference between two TTLs.

Mathematically, the Levenshtein distance between two strings a and b, of sizes i and j, can be expressed as:

$$L_{a,b}\ (i,j) = \begin{cases} 0 & ,i=j=0 \\ i & ,j=0\ and\ i>0 \\ j & ,i=0\ and\ j>0 \\ \min\begin{cases} L_{a,b}(i-1,j)+1 \\ L_{a,b}(i,j-1)+1 \\ L_{a,b}(i-1,j-1)+[a_i \neq b_j] \end{cases} & ,else \end{cases} \tag{43}$$

The above equation (Eq. 43) illustrates, $L_{a,b}\ (i,j)$ is the minimum number of edits required to convert string a to string b. In our scenario, we represent the Levenshtein distance defined as the minimum number of changes needed to convert TTL γ_t to *TTL* γ_{t+1} as $L_d(\gamma_{t+1}, \gamma_t)$. By using Eq. 43, the Levenshtein distance between TTLs at time 09:05:03 and 10:05:03 in London on Oct. 22, 2011 (shown in Fig. 75) is 2. On the same day, the Levenshtein distance between TTLs at time 17:05:04 and 18:05:03 in London (Fig. 75) is 10.

Volatility

Each pair of TTLs in consecutive time slots generate one L_d value. Thus, over a given range of time slots, we have a series of L_d values which is representative of how fast the TTL of the user group was changing over time. Let us consider *T* as the number of time slots for the duration of observation, i.e. if we want to calculate the volatility per day and each time slot is 5 minutes, then T= (24*60)/5 = 288.

Then, the volatility for user group *g* starting at time *st* can be defined as:

$$Volatility_{st,T} = \sum_{t=1}^{T-1} L_d(\gamma_{t+1}, \gamma_t) \qquad (44)$$

Note that the granularity of volatility measures can be adjusted. For example, if we measure the half-hour-wise change in the TTL for a day, then we can set T to (24*60)/30=48. In Fig. 77, we show the volatility time series over 5 months in some major US cities (with *T*=288). Peaks in the Fig. 77 refers to days when the TTL was changing rapidly. Peaks are not appealing to us, since rapid changes in TTL indicate that the attention of user groups is shifting rapidly. i.e., there is lack of persistent attention.

However, a minima in the volatility curve is of significant interest. Minima resemble days when the TTL was not changing significantly. In other words, attention is not shifting constantly, rather it is persisting. This can be caused due to two reasons, (1) Nothing is happening that is attention worthy, or (2) Something huge has captured user attention.

Especially, when a majority of the user groups display the same minima together on some day, like on March 8th, 2012 (see Fig. 76), then it indicates focused attention within all the user groups to some potentially big event. On March 8th 2012, every city in the

US (and most parts of the world) was trending #KONY2012, which was one of the largest online campaigns ever launched through social media[8]. The attention received by the event leads to the combined drop in volatility across all cities on March 8th (Fig. 77).Within user groups of brand audiences, we noticed a strong relation between volatility and the user groups' reaction to these physical world events, as shown in Fig. 76, where real-world events in 2012 were chosen for the visualization. Notice how Pepsi user group pays attention to anything involving Justin Bieber (*Entertainment*).

The categorical affinity within user groups, which we observed in Fig. 76, is triggered in this real world situation, causing audiences of Pepsi to strongly react to *Entertainment* events. Similarly, user group of Associated Press is very attentive to *Politics* and *News*. All user groups pay focused attention when there is an earthquake (emergency breaking news).

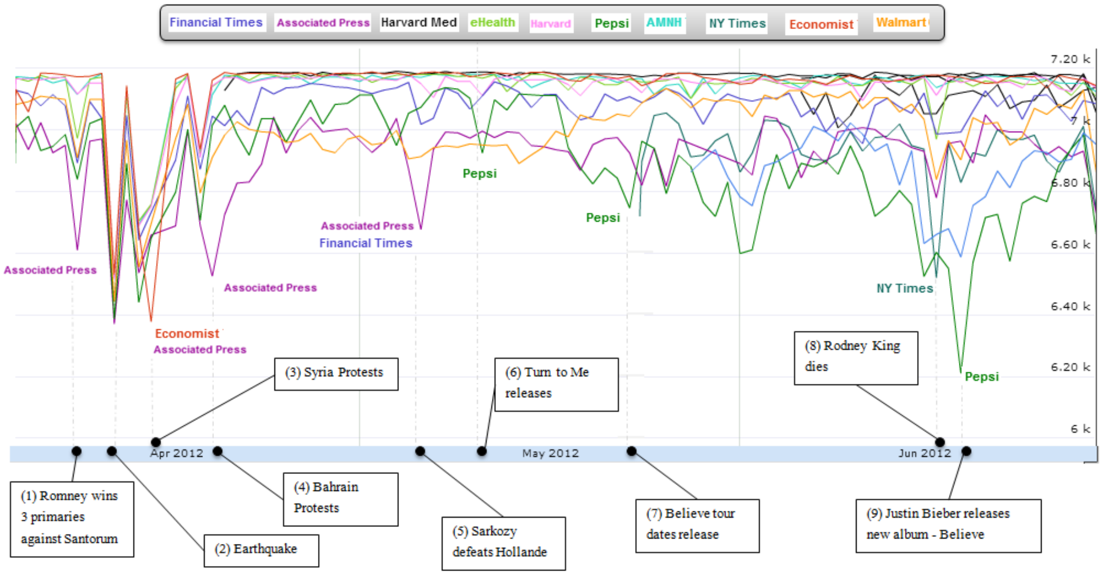


Figure 76: Variation of user group attention with real world events. The Attention Automaton uses the categorical affinity of user groups of brands with varying volatility to reflect how audiences react to real world events, such as earthquakes, election results etc

[8] Twitter trending #KONY2012 all day in every city significantly contributed to the campaign video receiving a record 60 million views in just 4 days!

Volatility Signal-to-Noise ratio

From the volatility time series of an user group, we can also infer that some user groups are always volatile (New York), while others are volatile only on few days (Salt Lake City). Volatility Signal-to-Noise ratio (VSNR) is a metric that captures how often an user group is volatile. Let ϑ_g represent the volatility time series of some user group *g*, i.e.,

$$\vartheta_g = \{(t1{:}\, Volatility_{t1,T}), (t2{:}\, Volatility_{t2,T}), \dots\}$$

where *t1* is a time instant and *T* is the number of slots over which the volatility was calculated (for a day *T=288*). Then, VSNR can be defined as:

$$VSNR = \delta_g = \frac{Mean(\vartheta_g)}{Std.Dev.(\vartheta_g)} \qquad (45)$$

It is evident why we call this signal-to-noise ratio, since it is basically the ratio of the mean to the standard deviation of the volatility signal. VSNR gives us a single number representing the attention shift tendency of the user group. appendix axxdepicts VSNR across cities worldwide. We notice Tokyo, New York, Djakarta, London, Los Angeles have high VSNR. In comparison, Montreal, Glasgow, Johannesburg and Mumbai have low VSNR. There can be two explanations of this observation, (1) cities with high VSNR have greater diversity in tweeter profiles - thus lots of topics capture attention and/or (2) cities with higher VSNR are strongly linked to other user groups, allowing for much larger exposure to diverse information forcing high attention shifts.

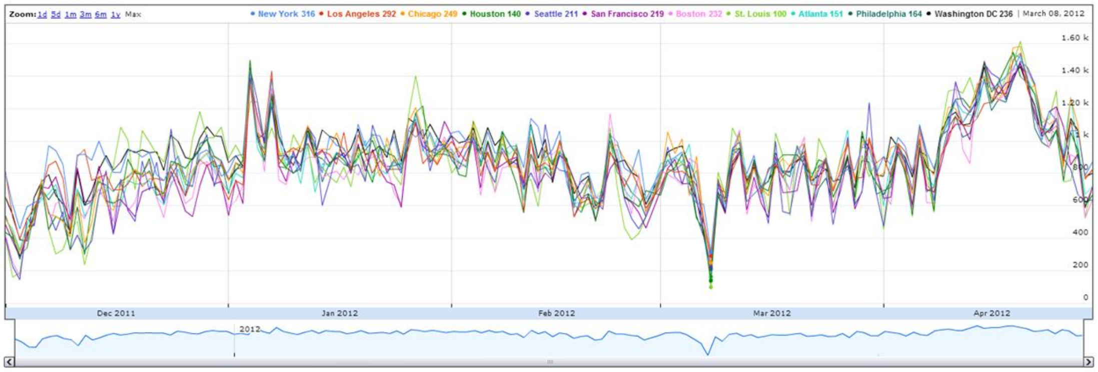


Figure 77: The volatility per day in some major US cities over a period of 5 months. Note the combined minima on March 8th, 2012 across all the cities, which is attributed to the #KONY2012 campaign

Attention Shift Tendencies

Consider the complex ecosystem where a set of user groups (agents) are consuming information. Each user group has some VSNR (δ), indicative of the perturbation of the user group caused by information flow in the underlying social network. Perturbance dynamics in complex networks suggest that there exists a feedback pattern created by the sub-structural network, such that the perturbation of each agent is directly or indirectly affected by another [87]. To put it simply, since the underlying social network governs information flow, the potential of information consumption (attention) of some user group depends on its connections to other user groups through which information reaches it[9]. Assuming all information produced is consumed within the social network, the attention shift tendency of an user group is the probability of consuming information (using attention) by the user group relative to the entire system. It can be defined as a simple ratio:

$$P(\vartheta_{g'}) = \frac{\delta_g{}'}{\sum_{g=1}^{G} \delta_g} \qquad (46)$$

where g' is some user group and G is the set of all user groups. Equation 46 gives us the probability of attention shift for some user group g' existing in a world of G groups. A

[9] Observe from A that locations such as NYC, Los Angeles and London have high VSNR, potentially needing to consume the increased information flow attributed to the numerous social network links between users in these locations.

higher probability indicates the user group is potentially more likely to transition to a new state every time.

Category Affinity. User groups also behave differently to trends in different categories. For example, the audience of Pepsi is highly attentive to any trend about *Entertainment*, especially Justin Bieber, whereas user group of San Francisco is more attentive to trends in *Gaming*, such as the trend '#halo4'. We categorize trends over 15 categories, based on whatthetrend.com and the category of the trend word in Wikipedia [112]. These categories are $\mathbb{C} =$ {*entertainment, gaming, lifestyle, science, sports, technology, business, spam, meme, conference or event, news, place or location, holiday or date* and *charity or cause* }.

Follower Affinity. A similar selective affinity to trends is demonstrated in BT-TTL. Followers of specific accounts have selective congeniality to certain trend categories. Fig. 78 shows the categorical distribution of trends that was observed in 3-months worth BT data for followers of four brands, namely Harvard, Burberry, Pepsi and Economist.

It is very interesting to notice how followers are receptive to certain category trends (larger bubbles in Fig. 78) and not so much to others. For example, Pepsi's followers are predominantly sensitive to trends in *Entertainment* whereas Burberry's followers do not care much about *Sports*. Moreover, notice that Harvard followers have a versatile set of categories they are interested in (many same sized bubbles in Fig. 78). Due to lack of space, we cannot provide all the charts. The main indication from this data is that user groups have selective categories they are attentive towards. Therefore, *whether an user group allows an impending trend to enter its TTL is partially dependent on the category of the impending trend*.

Transition Probability

As mentioned earlier, the transition probability determines the probability of a state transition in response to an action stimulus. In the previous sub-sections we describe two probabilistic random variables: attention shift $P(\vartheta_{g'})$ and categorical affinity $P(C_{g'})$ for the user group g'. Note that although $P(\vartheta_{g'})$ has no concern for the action stimulus, $P(C_{g'})$ is fundamentally determined by the action. We know that the joint probability of two mutually independent random variables X and Y is given by:

$$P(X,Y) = P(Y).P(X)$$

In our scenario, the categorical affinity is assumed to be independent of the attention shift tendency, and thus, the transition probability can be considered a joint distribution, written as:

$$Distr(S_A) = P(C_{g'}).\ P(\vartheta_{g'}) \qquad (47)$$

which completes the transition relation mentioned earlier.

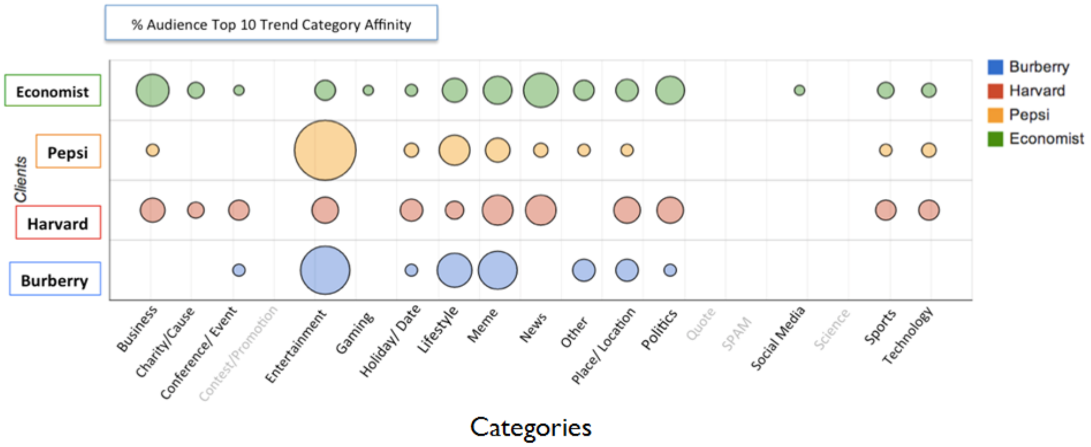


Figure 78: Distribution of categories for trends among followers of some brands. Size of bubble shows percentage of trends in the user group that belonged to a particular category

7.3 Applications

7.3.1 Modeling Collective Attention in Geographical Communities

Experimental Settings. To test the model, we first need to prepare a dataset of action trends. This is shortlisted from the trends data. Let all user groups be denoted by *U*. Let $\overline{\gamma_{g',(t)}}$ represent the trends in the TTL of user group g' at time *t*. To test a particular user group g', we need to choose actions strings consisting of trends not currently in the user group. Note that these trends are competing simultaneously to be part of $\overline{\gamma_{g',(t+1)}}$. For this purpose, we collect all the unique trends across all user groups *U* at time *t* that are not in $\overline{\gamma_{g',(t)}}$. We also record the number of times they have occurred in other TTLs. Thus, this gives us a set *D* of potential action trends:

$$D = \{d_1, d_2, \ldots., d_m\}$$

where $d_j = (x_j,\ y_j)$ represents a trend along with the number of other TTLs it occurs in at *t*, i.e. $x \in \{\overline{\gamma_{U,t}} - \overline{\gamma_{g',t}}\},\ y \geq 1, 1 \leq j \leq m.$ We collect the top-*k* trends in *D* and choose action strings of different sizes to feed to the automaton. The various action strings encompass the set of actions (mentioned in Section 4.1), which can be written as,

$$Act_A = K_{C_{m*}}, where\ 1 \leq\ m^* \leq 10, K = top_k(D)$$

and n_{C_r} represents the standard notation of number of combinations of *n* items taking *r* at a time. The above equation selects combinations of $m*$ items from the set *K* as action string. Empirically, we found that *k* =500 and $1 \leq m^* \leq 10$ are good parameters for experiments.

The overall purpose of the automaton is to predict most probable future states. The future state depends on the new trends introduced in the next TTL state $\gamma_{g',(t+1)}$. At each time instant *t*, $Act_{A,t}$ defines the trends that are competing to make it to $\overline{\gamma_{g',(t+1)}}$. However, only q new trends will eventually be in $\overline{\gamma_{g',(t+1)}}$. In other words, $\overline{\gamma_{g',(t+1)}} - \overline{\gamma_{g',(t)}} = q$. The task of our evaluation is to correctly detect the *q* trends that will cause the automaton to jump from state $\gamma_{g',(t)}$ to state $\gamma_{g',(t+1)}$ forced by the action string of q trends. Said alternatively, we need to detect the q trends which the automaton will accept out of all the competing trends; that essentially mirrors the actual TTL shift in the Twitter world at that time instant.

Benchmarks. Lack of exact comparative work limits our options in selecting benchmarks. However, since this is a time series prediction scenario, we use the traditional *Auto-Regressive Integrated Moving Average (ARIMA)* model which is widely used in statistical analysis of time series with drift [113]. Given a time series, ARIMA can predict future values in the series. The model is generally referred to as an ARIMA(a,i,v) model where a, i, and v are non-negative integers that refer to the order of the autoregressive, integrated, and moving average parts of the model respectively. We use ARIMA(1, 2,1) to predict trends for future TTLs. The 'statmodels' python package was employed to implement ARIMA in our scenario (http://pypi.python.org).

Additionally, we use a *random selection scheme*, where the predictor randomly chooses trends to appear in the next TTL. This benchmark is chosen to study if the trend shifts resemble random jumps.

Metrics. Since the task is detecting a set of correct trends that mirrors the actual Twitter world TTL state transition, we can simply utilize the precision and recall metrics which are popular in information retrieval. Precision measures how many of the identified trends were actually in q. Recall measures how many of the q trends were retrieved. The harmonic mean of precision and recall is called *F-Score*, which is $2 * precision * recall/(precision + recall)$ and serves as our evaluation metric. A higher F-Score suggests higher accuracy of test results.

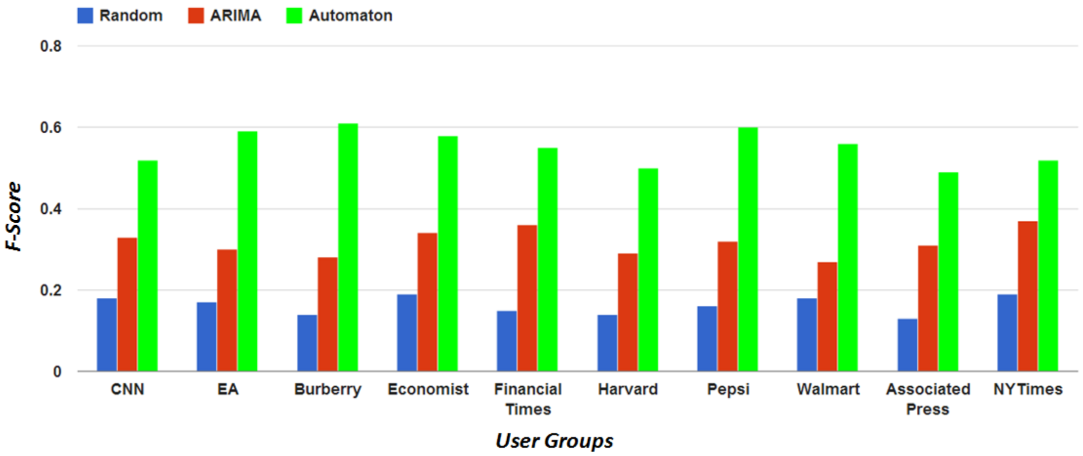


Figure 79: F-scores obtained in testing different models on user groups based on brand following

Results. We select 30 user groups of brands to perform similar experiments as the GT user groups. For each BT user group, half the time series is used for training and the other half for testing.

Results are reported in Fig. 79 for ten of these brand user groups. The average F-Score achieved using the random method, the ARIMA model and the Attention Automaton is 0.163. 0.317 and 0.552 respectively. Overall, the improvement using the Attention Automaton with respect to F-score was 238% over the random scheme and 74% over the ARIMA model.

One interesting observation is that the F-score improvement of the Automaton over ARIMA is different for different user groups. More precisely, Automaton performs 61% better for user groups of EA, Pepsi, Burberry and Walmart compared to Harvard, Associated Press or CNN. We contribute this nature to the distribution of categorical affinity of user groups. User groups of Pepsi, EA and Bur-berry have small number of categories they have affinity towards, effectively reducing the decision space for prediction. As shown in Fig. 78, Pepsi has high affinity to 'Entertainment' trends. In contrast, user groups of CNN/Harvard have a large number of categories they have affinity towards. ARIMA lacks understanding of categorical affinity, as it is driven by the statistical variation in the time series. Therefore, for user groups that have very high affinity to very few categories, the Attention Automaton performs significantly better than ARIMA/

7.3.2 Modeling Collective Attention in Geographical Communities

Geographical Trend Initiation. For GT-TTL, a simplistic way to experiment categorical affinity of user groups is to find the location where a trend originated (first trended in the Twitter world) and note the category of that trend. We call this the trend initiation of an user group. Given trends in a category, we can observe the proportion of these trends that originated in some city, and normalize it by initiation in other cities worldwide. This provides us with a Normalized Initiation Score (NIS) between 0 and 1.

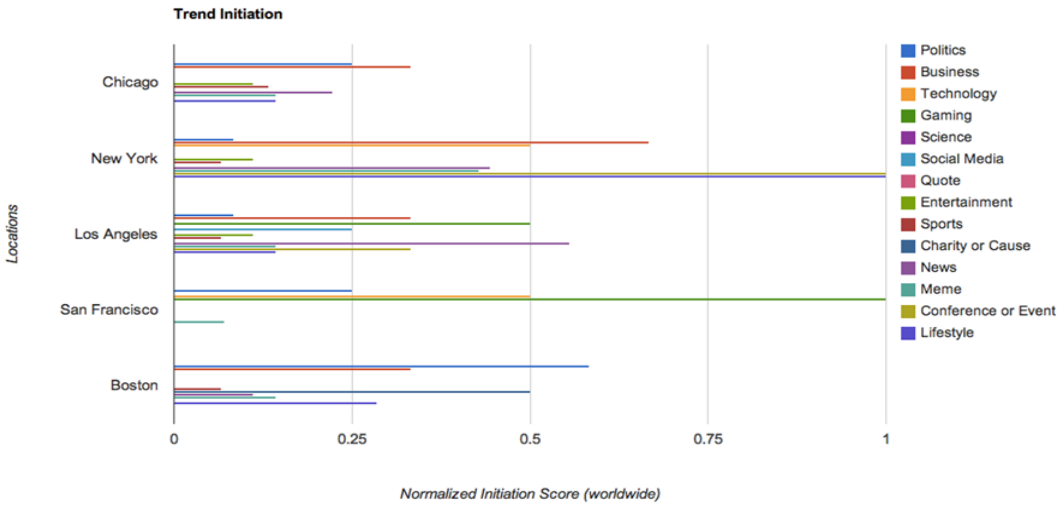


Figure 80: Initiation scores for trends in different categories across US cities

In Fig. 80, we show the NIS for five major US cities. Notice most *Gaming* trends in the Twitter world originate in San Francisco whereas a significant portion of *Business* trends originate in New York. Somewhat surprisingly, Boston leads all these cities in generating political trends.

User Group	***Random***	***ARIMA***	***Automaton***
New York	0.18	0.33	*0.45*
Los Angeles	0.17	0.30	*0.49*
Baton Rouge	0.14	0.28	*0.42*
Boston	0.19	0.34	*0.48*
Paris	0.15	0.36	*0.51*
London	0.14	0.27	*0.43*
Dublin	0.16	0.32	*0.54*
Atlanta	0.18	0.35	*0.55*
San Francisco	0.13	0.31	*0.50*
Glasgow	0.19	0.37	*0.44*

Table 16. F-scores obtained in testing different models on user groups in various geographical locations

We randomly select 30 locations worldwide to perform these tests. The average F-score obtained using the Attention Automaton, ARIMA and Random models was 0.49, 0.34 and 0.18 respectively. Overall, the *F-score performance using the Attention Automaton was 44% better than ARIMA and 171% better than random selection*. The F-score of user groups in 10 out of the 30 locations chosen for testing is provided in Table 16, which was generated using 3 months of the user group data for training and 3 months for testing.

CHAPTER 8: CONCLUSION AND FUTURE WORK

The age of Big Data is upon us; irrespective of how varied are its definitions by both camps - the ones who think it will be pivotal to our progress in understanding nature and humanity and the others who claim it is merely another technology bubble. However, we cannot deny its impact on our lives. Media hails Facebook as the world's biggest friend circle, Twitter as the world's biggest cocktail party, IBM's Watson as the breakthrough expert system and Google as the world's most intelligent search engine. Most major companies leverage huge amount data which they did not foresee two decade ago. Whether these titles will be held in the next decade is another story. Moreover, long standing technologies are being disrupted due to this data. SQL databases are no longer capable of efficiently handling this streaming data. Real time algorithms are winning the battle in almost every field ranging from finance to biology.

More importantly though, I feel the urgent need of algorithms that can understand this data across different platforms, and in one word - make sense of it all. This grand challenge is non-trivial due to a many factors, including the fact that platforms do not want to share data freely. Different platforms also have different noise levels in data, vastly disparate features and dissimilar rates of data generation. To approach this problem, the first task would be to observe two key facts: (1) the data across domains have a semantic connection and might symbiotically assist the understanding of one another, and (2) data in some domains have traces of social touches by online users, giving us the power to leverage crowd-sourced information. The second task is to detect data domains that can work in synergy with one another. This thesis identifies such domains and enables machine learning algorithms to use data of one domain to augment computational performance in another. Over the next paragraphs, we summarize how this is achieved in various chapters of this work.

The various domains in which data can be used synergistically - that is the data of one domain can enrich the data in another is identified in Chapter 1. It also discusses the role of the multimedia with respect to this data, i.e. some of this data is natural language, some are stored as images, some videos and others in micro-texts format. Learning from data in any domain is possible through machine learning algorithms, which are described in Chapter 2. This chapter also mentions the different formats of data and the challenges that algorithms must overcome to recover intelligent features from disparate data formats. Following this, each chapter picks up data from two domains and explains how a machine learning algorithm's performance can be improved by the cross-domain data usage.

In Chapter 3, we propose a way to learn from social streaming data. Our Online Streaming LDA is able to scale with the bursty nature of Twitter while making sense of the noisy tweets. It creates an intermediate topic space, which can allow for bidirectional information exchange between Twitter and any other data domain. In Chapter 4, we use YouTube videos as the other domain and explain how this cross-domain information transfer actually happens. This chapter is critical to the thesis, it describes the very engine that can handle the fast Twitter stream, learn information in real-time and use it to augment video domain application performance. Specifically, three different video domain applications are shown to be boosted by this cross-domain information transfer and transformed into socially-aware media applications, namely, social video recommendation, social video popularity prediction and social query suggestion.

Moving away from social and video data, Chapter 5 focuses on the Semantic Web, and ways to leverage semantic data in various diverse applications. We show how semantic network data can be used to augment web page data, categorizing and predicting social trending topics and forecasting popularity of movies. Thus, here the cross-domain data involves semantic network data, social trends, movie data etc. In Chapter 6, we show how the semantic web data can be used to better understanding natural language data of search queries. We also computationally reanimate a cognitive psychology model in this chapter and use it for improved natural language processing. Finally in Chapter 7, we argue that all this data is creating an essential change in the world - ushering the age of attention economy, where there is too much information to pay attention to. As a result, users pay attention to few things, shifting between categories of information. We demonstrate how to build an automaton that can model these attention shifts in online social networks.

8.1 The Cross-Domain Issue

The Cross-Domain issue is deeply tied with the structure and evolution of the world wide web and major organizations dealing with huge amounts of data. All this data cannot potentially be located in a single platform, for business and technological reasons. Thus, data is spread across various domains on the Internet (Fig. 81). These different domains bear data which have dissimilar rates of generation, disparate feature space, formats, types and may reflect the physical world in different scopes.

However, some of these domains possess data which have semantic connections and social undertones. This is observed in the way the data instances behave (e.g., in terms of life length, popularity etc.). Thus, data from two domains can be sometimes used synergistically - allowing performance of algorithms in one domain to be boosted by using some data from another domain.

The usage of this cross-domain data requires intelligent understanding of the feature space, in addition to the combinatorial alignment of features that results in performance improvement and avoids negative information transfer.

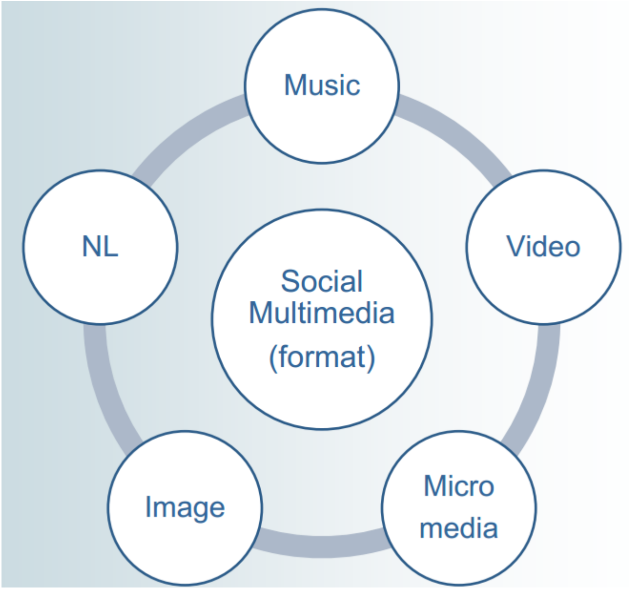


Figure 81: Multimedia is available in different formats, and domains often specialize in generating data of one format.

8.2 The Information Transfer Issue

Once data domains are detected, we need to build a technique to efficiently transfer this information across domains. In most situations, this is dependent on (1) the source domain, (2) the target domain and (3) the algorithm being used in the target domain to improve performance. In other words, different machine learning algorithms handle data differently, and seamlessly associating cross-domain data into them might be tricky.

The need for information transfer is critical to the progress of computing in general, where algorithms are intelligent to analyze disparate data streams efficiently. It is important we learn to semantically connect a tweet, an image, a video, a Wikipedia article, a gif and match it to a search query by some user, in real-time, taking into account social network of the user. This is how collective intelligence will become reality, where the algorithms are not bound by same data distributions and features (which is a current problem with most machine learning algorithms).

Finally, we must understand that if the human brain is an organic computer, then it has a very efficient information transfer technique. We learn the basics and school and at an young age, which we use and re-use as modules in several tasks of growing complexity and difficulty. The human mind is exceptional at detecting data domains that can be used together to solve a task, and how information in this cross-domain data can be seamless combined in a learning process. If computational intelligence is ever to reach the levels of human intelligence, it must make sense of cross-domain data in real-time and find efficient techniques of transferring information.

8.3 Future Work

I believe this work will open several avenues of future research. (1) Firstly, the intermediate topic space we built in Chapter 3 can be made richer by connecting topics using semantic information from semantic networks. Questions still remain as to how we can archive older topics to make space for new ones. (2) Secondly, I leave it to future researchers to find applications were social information inclusion will boost performance, beyond the domain of video relevancy and popularity prediction. (3) Visualizing data across domains and how information transfer takes place will be very important in edutainment. (4) During my research with social trends, I found traces of proof that trends originate and initially appear across various geographical locations in a near-chaotic order (before they are global trends), and then reach a tipping point after which, they explode into global trends. An example of this is shown in Fig. 82, where a social trend breaks the tipping points and starts trending globally, instead of just in a few cities. What makes a trend reach the tipping point? Is there a critical order to reach the critical point?

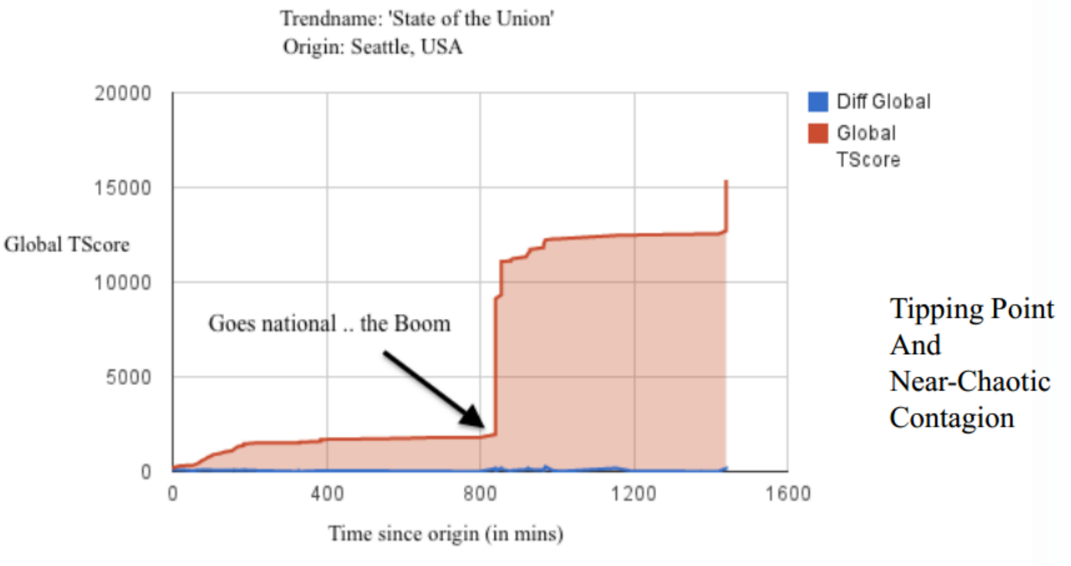


Figure 82: The tipping point after origin, after which a trend becomes global/national and starts trending in all cities

(5) Finally, while analysis social trend signals in Chapter 4 and 7, we came to the conclusion that some properties of the network might be encoded in the trend signal. Thus analyzing the trend signal can give us information about the dynamic information transfer within the network itself. This is a fascinating perspective, wherein a network

can then be analyzed without relying solely on graph theory. Instead, we could perform signal analysis on the trend signal to attain information about the network.

8.4 Last Impressions

As a collection of last impressions, I would like to elucidate the broader impact of my research. Here we discuss some deeper questions from three perspectives: (1) the value of this research beyond computer science, in other scientific and non-scientific fields of education, (2) the socio-cultural connotations of the thesis, and (3) the philosophical essence of the thesis.

Value of the research beyond Computer Science, in other scientific and humanities domains:

--*In Digital Media (detecting popular content)*: One of the key metrics used in evaluating the quality of digital media is the popularity of the corresponding published work among the circulation masses. Popularity is a result of widespread exposure, collective attention and information cascades. In Chapter 3, we demonstrate how to detect popular topics in Twitter using OSLDA. Many popularity signals with reference to Twitter trending topics are transferred across domains, like YouTube, where content related to trending topics gains popularity (such as more view counts) - as shown in Chapter 4. Most of Chapter 7 is also devoted to measuring collective attention of users towards various digital media (and other) brands boosting popularity of such content.

--*In Journalism (spotting news stories quickly using OSLDA)* : One of the most desired applications of news editors is a tool that allows them to quickly figure out what's 'news', so they can share the story with the world. Sadly, most editors still need to spend long hours reading through posts on Reddit/Digg or Twitter to figure out what news they should feature on their publishing sites. In Chapter 3, we describe the OSLDA algorithm which can learn and score trending topics from Twitter. The corresponding software can recommend news articles/tweets that are being shared/RT-ed a lot, and is being used by editors (news and social media) to decide what content is worth featuring on their news sites.

-- *In Cultural Anthropology (evolution of TIME magazine topics)* : Culture encompasses the range of human phenomena which are not manifested by genetic inheritance. Fundamentally, cultural anthropology studies the cultural variation of humans, in various communities and over time. In Chapter 5 (pages 86-99), we discuss the evolution of topics as featured on the cover page of the TIME magazine. We claim these topics have enough socio-cultural importance, given that the magazine has the largest circulation for a weekly news magazine.

-- *In Sociology (social attention, persistence etc.)* : Sociology is the systematic study of human social actions. A significant portion of this thesis is devoted to understanding the concept of 'social attention' caused by interaction among social agents on some multimedia content shared in social media. Most prominently in Chapter 5 (pages 100-105), we explore the spatio-temporal evolution of social trends, which involves developing a model that can answer some key questions about social attention - what persists, what is likely to reappear, what spreads furthest and what user groups are most receptive to changing trends.

-- *In Linguistics* (*computational cognitive model*) : The scientific study of language has an extremely wide range of impressive work. One existing challenge involves an intuitive explanation of what makes humans highly robust (can easily understand/comprehend) to noisy sentence constructions (which deviate from grammatical rules), whereas machines fail to make sense of such sentences. In Chapter 6, we propose a cognitive model to reason and provide empirical proof that adaptation and robustness to noise in sentences constructions can be achieved by adjusting the resolution at which semantics needs to be extracted from the sentence. This conceptual abstraction was computationally modeled and proves to be significantly better than some state-of-the-art techniques in natural language processing (NLP).

-- *In performing arts (movie genome communities)* : The movie genome community detection described in Chapter 5 (pages 106-122) shows a novel way in predicting the ingredients that comprise a likable movie. It could be an interesting tool for movie studio/network producers and actors to gauge the potential of a movie based on its genetic composition of the movie before they commit to making the movie.

Socio-Cultural Connotations:

In this part we focus on the socio-cultural effects of this work and how it improves our understanding of the human condition. The *Time* magazine analysis gives us a comprehensible depiction of what topics have affected human lives over decades. For example, it is remarkable to notice how less we were concerned about the environment before 1950s or the somewhat sad but decreasing interest in theater over time (Appendix A2, pg. 205).

Further, remember that 'memes' are often called the unit of culture. Memes are very commonly found on Twitter as trending topics, and thus measures like 'persistence', 'recurrence' described in Chapter 5 give us information about their growth-decay patterns - explaining the human condition around ideas that quickly capture our attention. This also concerns the concept of social contagion theory, which was mentioned as one of the future works in this thesis.

Another factor that led to socio-cultural evolution was language, which helped us to communicate. One continual topic of scholarly discussions for centuries has been the question on the origin of languages. Language is important for a key human ability - communication and thinking. The minimalist theory of Chomsky proposes that there must be a minimal generative grammar that lexically assures fixed sentence structure for comprehensible meaning. However, humans can understand mis-constructed sentences surprisingly well. Thus arose the field of cognitive linguistics, which argues that principal linguistic phenomena such as syntax is essentially conceptual in nature - i.e. humans recognize grammar in terms of conceptualization and not purely lexical constructs. One computational implementation of cognitive linguistics is described in Chapter 6, where we re-create the 'Structure-of-Intellect' model computationally to extract semantics from natural language search queries.

Finally, in Chapter 7, we show empirical proof of evolution of social dynamics as users respond to real-world events by tweeting about them on Twitter. Social dynamics is a critical component of the human condition - our behavior governed by not only the physical reality but also the digital activity of individuals. Exploring the aggregate behavior of a group of users based on their location or their preference for some brand/company can answer interesting sociological questions about the global yet dispersed attention spans of such communities within the set of human population which are digitally connected.

Philosophical Essence:

Let us discuss the philosophical spirit of this thesis in terms of its impact to the scientific way, humanity and existence. I think my research addresses a fundamental question of epistemological philosophy - the question of *'knowledge that, knowledge how and knowledge acquaintance*'. For example, in mathematics we possess the *'knowledge that*' 3+3 =6, then the *'knowledge how'* the + operator works and the 'knowledge acquaintance' that 3 dogs + 3 cats does not make 6 dog-cats.

The *'knowledge that'* data from one domain can improve performance in another domain is covered in most chapters of this thesis, for example from Twitter to Video domain, or from Semantic Web to Natural Language domain. Then comes the question of *'knowledge how'* , i.e. how do we transfer this knowledge. Recall that *SocialTransfer* shows us how to transfer the knowledge. Finally, the *'knowledge acquaintance'* topic always rests in the background, which is why we do not claim social media data can help understand brain FMRI (functional magnetic resonance imaging) data. The final claim might be proved wrong, since 3 dogs + 3 cats does make 6 animals - similarly someday social media data might help us understand brain signals better.

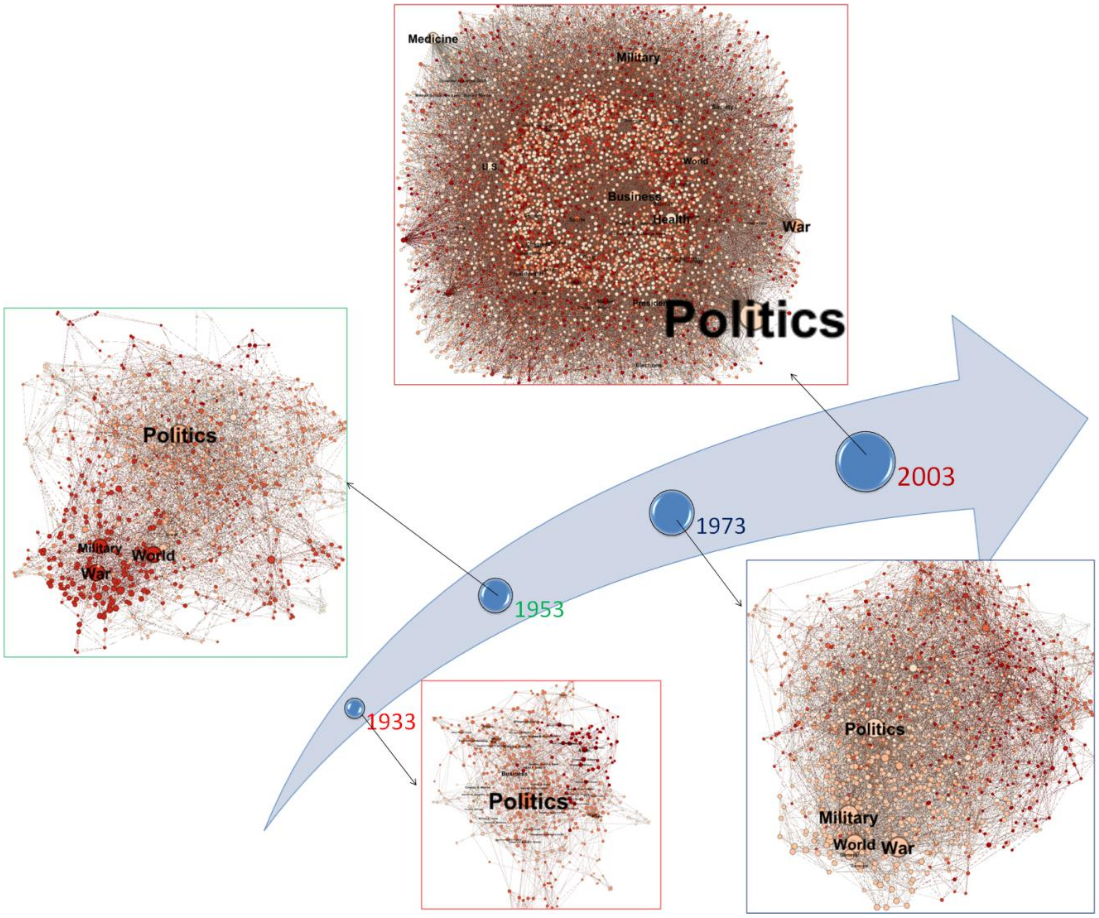
In truth, the reality of human existence is strongly dependent on our perception through senses. The sensory information has different features (image/sound etc.), however, it is in our potential of combining these disparate data and making sense of it is what makes us a strong species. Making sense of what we perceive is the task of the knowledge gathered from our experiences. This knowledge must be transferred across various layers, starting in detecting which two pockets can be combined, how to combine them and how to store it and utilize in future applicable scenarios.

Returning from broader discussions to the Computer Science, I strongly believe that understanding of the philosophy of Computer Science is more important than ever in the modern world, given its ubiquitous applicability in our present existence. Our world is clandestinely governed by data and algorithms, ensuring we have a safe flight between cities, alleviating our surprise when it snows, assuring us of safely performing online banking transactions and guiding the Curiosity mars rover 140 million miles away from Earth. Thus, it is extremely important to study different types of data and understand how algorithms can be modified or reinvented to deal with them using both social and semantic structures.

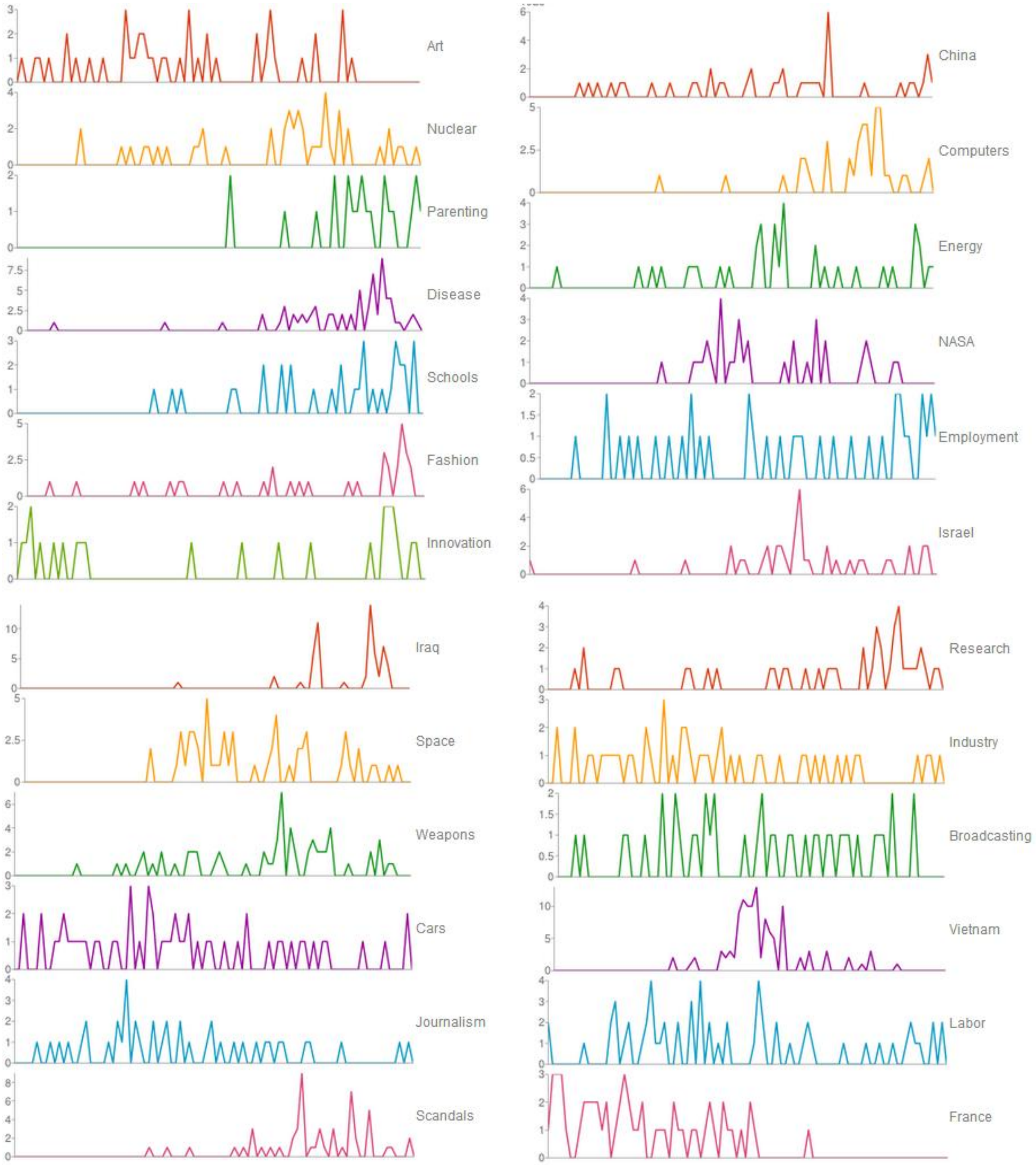
This page is intentionally left blank

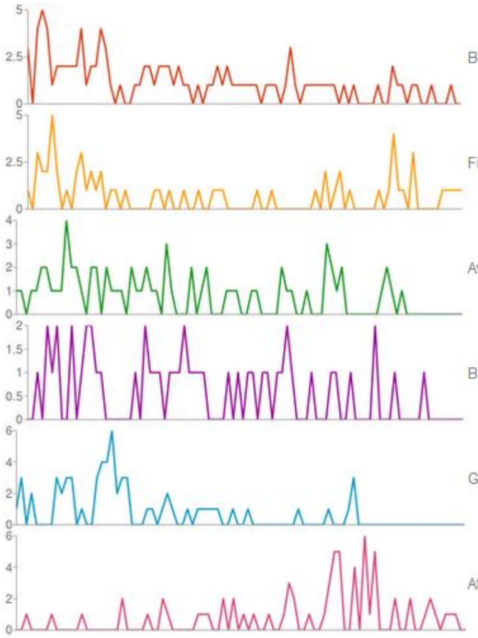
APPENDIX A: VISUALIZATIONS

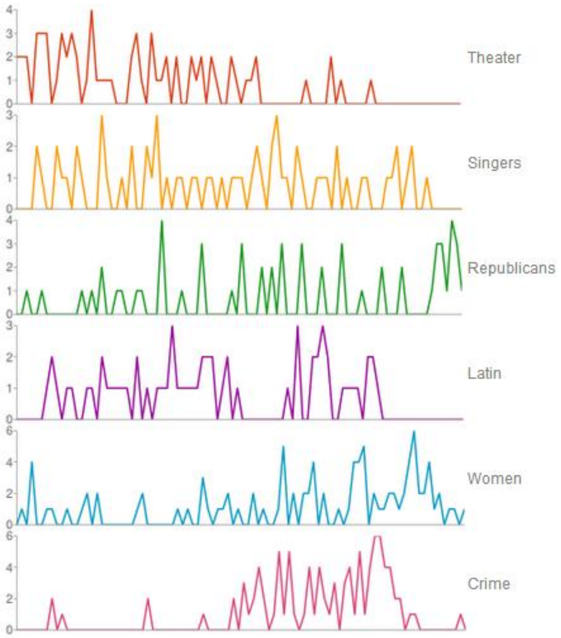
A1. The dynamic evolution of the topic network for the *Time* magazine articles.

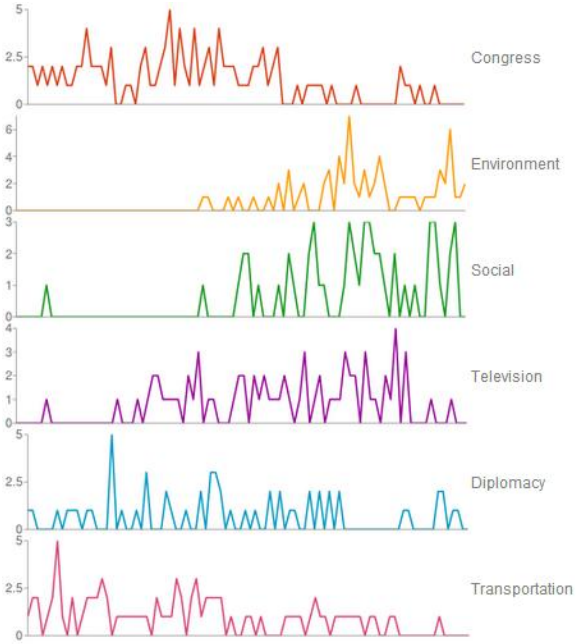


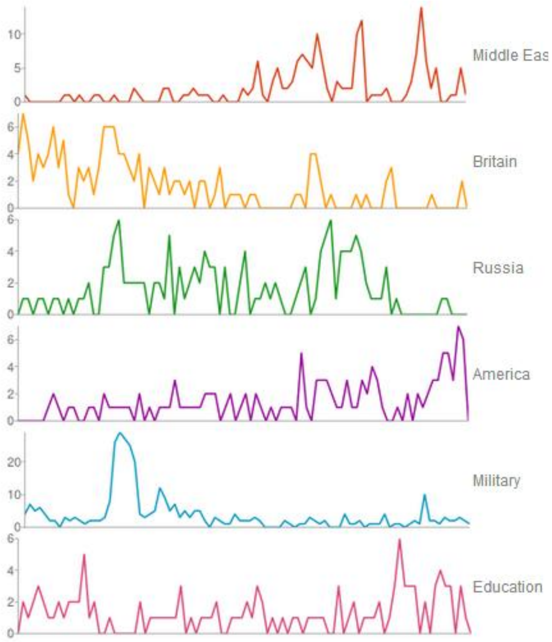
A2. The topic signals for various categories in the *Time* magazine articles.

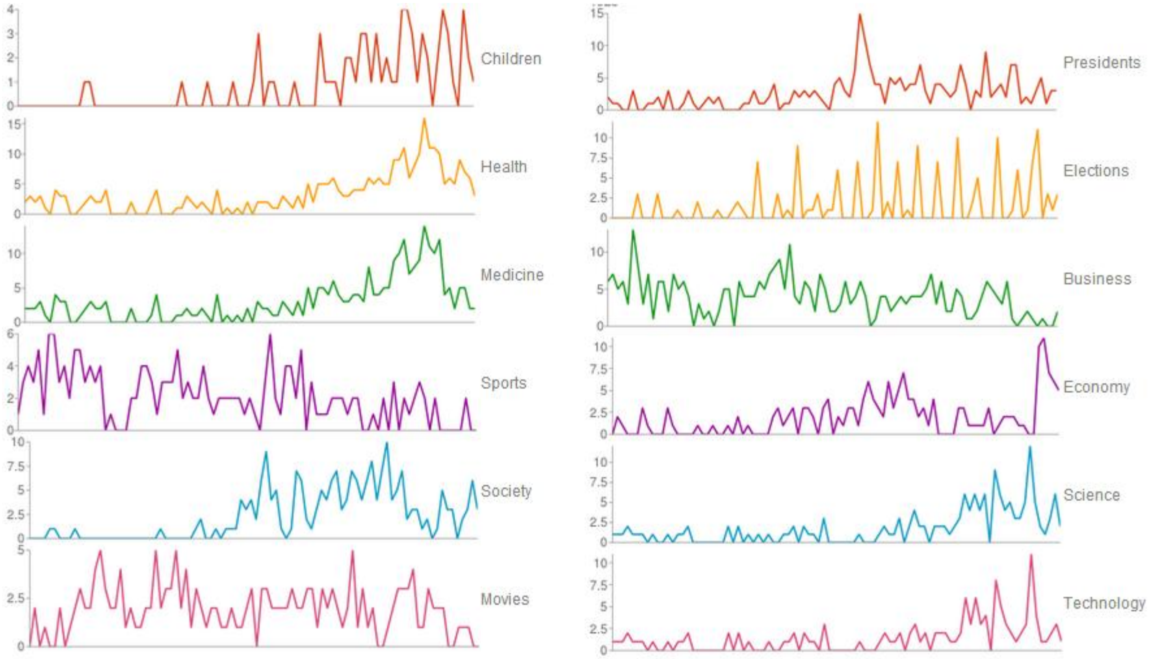



5
2.5
0
Books
5
2.5
0
Finance
4
3
2
1
0
Aviation
2
1.5
1
0.5
0
Baseball
6
4
2
0
Germany
6
4
2
0
Africa

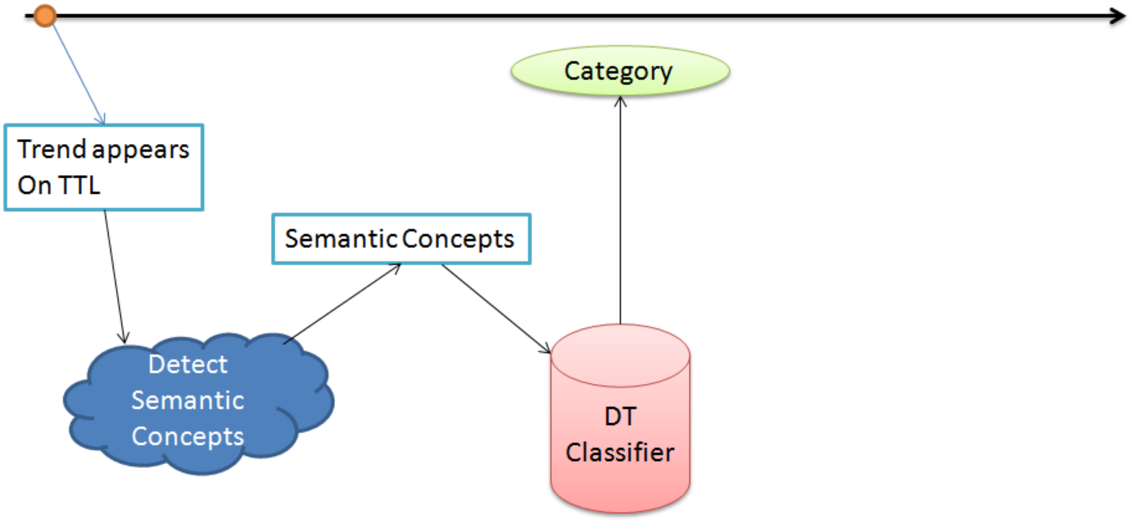

4
3
2
1
0
Theater
3
2
1
0
Singers
4
3
2
1
0
Republicans
3
2
1
0
Latin
6
4
2
0
Women
6
4
2
0
Crime


5
2.5
0
Congress
6
4
2
0
Environment
3
2
1
0
Social
4
3
2
1
0
Television
5
2.5
0
Diplomacy
5
2.5
0
Transportation

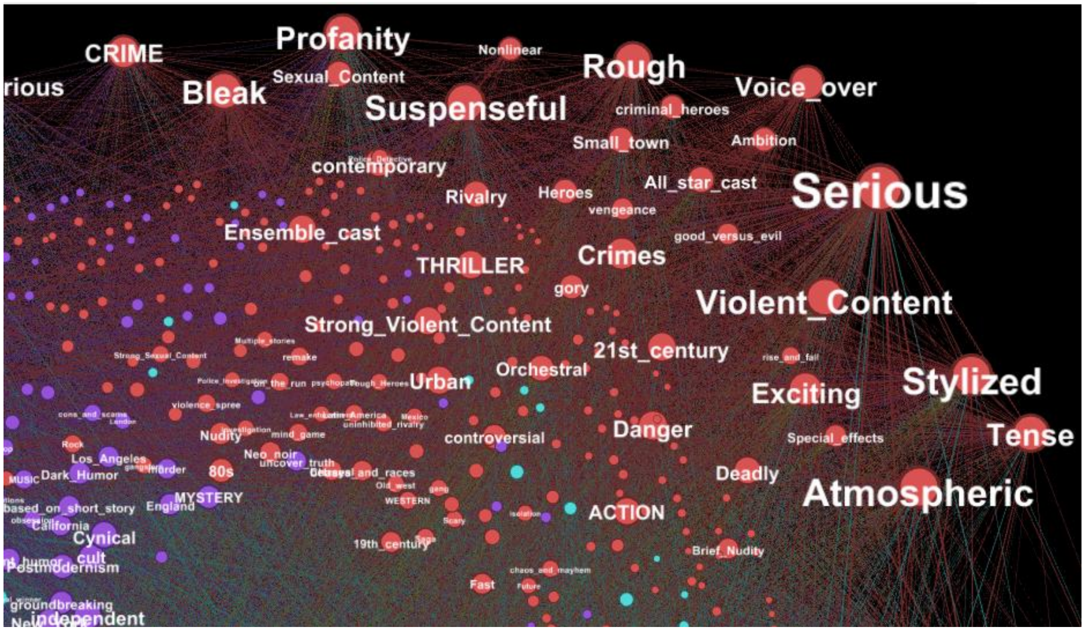

10
5
0
Middle Eas
6
4
2
0
Britain
6
4
2
0
Russia
6
4
2
0
America
20
10
0
Military
6
4
2
0
Education



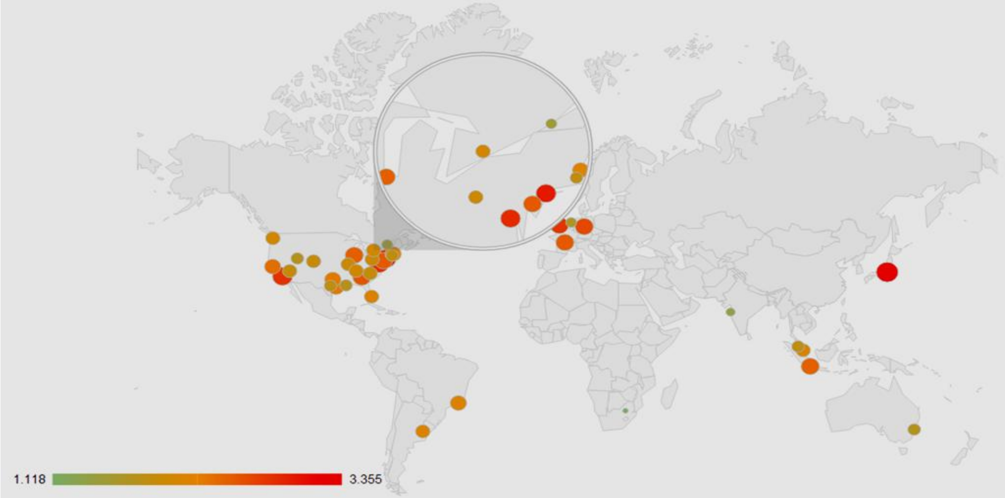
A3. The timeline of trend categorization using semantic web and decision trees described in Chapter 5



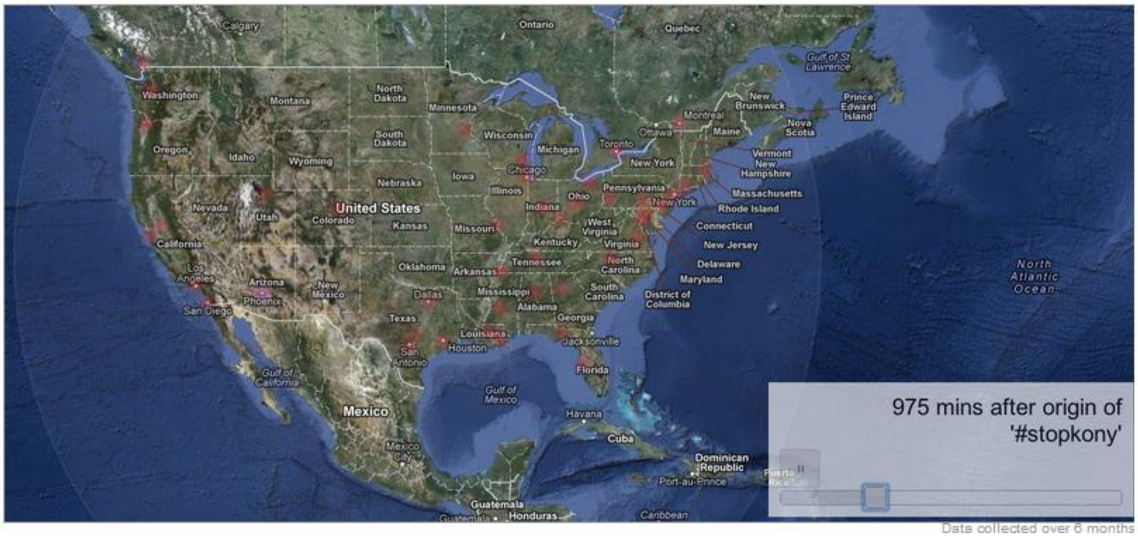
A4. Community in the movie genome co-occurrence network



A5. The Volatility-signal-to-noise ratio at various geographical locations



A5. Spread of trend '#stopkony'



Data collected over 6 months

BIBLIOGRAPHY

[1] Sadjadi, Farzad., "Comparison of fitness scaling functions in genetic algorithms with applications to optical processing," Optical Science and Technology, the SPIE 49th Annual Meeting. International Society for Optics and Photonics, (2004)

[2] Newman, Mark EJ., "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences* 103.23 (2006).

[3] Westergren, Tim., "The music genome project," Online: pandora. com/mgp.shtml Accessed 4.25, (2007).

[4] S. D. Roy, T. Mei, W. Zeng and S. Li., "Empowering Cross-Domain Internet Media With Real-Time Topic Learning From Social Streams." Proceedings of *IEEE International Conference on Multimedia and Expo*, (2012).

[5] http://www.jinni.com/movie-genome.html

[6] Girvan, Michelle, and Mark EJ Newman, "Community structure in social and biological networks*," Proceedings of the National Academy of Sciences* 99.12 (2002): 7821-7826.

[7] Goldberg, David E., and John H. Holland, "Genetic algorithms and machine learning," *Machine learning* 3, no. 2 (1988): 95-99.

[8] X. Cheng, C. Dale, and J. Liu. "Statistics and social network of youtube videos," In *Proc. of Inter. Workshop on Quality of Service*, pages 229–238, (2008).

[9] R. Zhou, S. Khemmarat, and L. Gao. "The impact of youtube recommendation system on video views.," In Proc. of International Conference on Internet Measurement, pages 404–410, (2010).

[10] Kocarev, Ljupco, and Visarath In, "Network science: A new paradigm shift," *Network, IEEE* 24.6 (2010): 6-9

[11] Kim, Sang Ho, Namkee Park, and Seung Hyun Park, "Exploring the Effects of Online Word of Mouth and Expert Reviews on Theatrical Movies' Box Office Success," *Journal of Media Economics* 26.2 (2013): 98-114

[12] Ishii, Akira, Hisashi Arakaki, Naoya Matsuda, Sanae Umemura, Tamiko Urushidani, Naoya Yamagata, and Narihiko Yoshida, "The 'hit' phenomenon: a mathematical model of human dynamics interactions as a stochastic process," *New Journal of Physics* 14.6 (2012): 063018.

[13] Mestyán, Márton, Taha Yasseri, and János Kertész, "Early Prediction of Movie Box Office Success based on Wikipedia Activity Big Data," *arXiv preprint* arXiv:1211.0970 (2012).

[14] Wong, Felix Ming Fai, Soumya Sen, and Mung Chiang, "Why watching movie tweets won't tell the whole story?," *Proceedings of the 2012 ACM workshop on Workshop on online social networks*. *ACM*, (2012).

[15] Cohen, A. M., Hersh, W. R., Dubay, C., & Spackman, K., "Using co-occurrence network structure to extract synonymous gene and protein names from MEDLINE abstracts," *BMC bioinformatics* 6.1 (2005).

[16] Mirizzi, R, Tommaso Di Noia, Azzurra Ragone, Vito Claudio Ostuni, and Eugenio Di Sciascio, "Movie Recommendation with DBpedia," In *IIR*, pp. 101-112. (2012).

[17] X. Jin, A. C. Gallagher, L. Cao, J. Luo, and J. Han. "The wisdom of social multimedia: using flickr for prediction and forecast," In Proc. of ACM Multimedia, pages 1235–1244, (2010).

[18] http://trec.nist.gov/data/tweets/

[19] K. Filippova and K. B. Hall. "Improved video categorization from text metadata and user comments," In Proc. of SIGIR, (2011).

[20] Namaan. M.. "Social multimedia: highlighting opportunities for search and mining of multimedia data in social media applications." Multimedia Tools and Applications, 56(1):9, (2012).

[21] D. Mavroeidis." Mind the eigen-gap, or how to accelerate semi-supervised spectral learning algorithms," *In Proc. of IJCAI*, pages 2692–2697, (2011).

[22] H. Luo, J. Fan, D. A. Keim, and S. Satoh. "Personalized news video recommendation," *In Proc. of MMM*, pages 459–471,(2009).

[23] W. Dai, O. Jin, G.-R. Xue, Q. Yang, and Y. Yu. "Eigentransfer: a unified framework for transfer learning,". *In Proc. of ICML*, page 25, (2009).

[24] F. R. Bach and M. I. Jordan, "Learning spectral clustering, with application to speech separation," In *Jour. of Machine Learning Research*, 7:1963-2001, (2006).

[25] J. Davidson, B. Liebald, Junning Liu, P. Nandy and T. V. Vleet. "The YouTube Video Recommendation System," *In Proceedings of Conference on Recommender System*. (2010)

[26] S. Baluja, R. Seth, D. Sivakumar, Y. Jing, J. Yagnik, S. Kumar, D. Ravichandran and M. Aly. "Video Suggestion and Discovery for YouTube: Taking Random Walks Through the View Graph," In *World Wide Web Conference*. (2008)

[27] B. E. Boser, I. Guyon, and V. Vapnik. "A training algorithm forcoptimal margin classifiers," *In Proc. of Workshop on Computational Learning Theory*, (1992).

[28] Z. Chen, J. Cao, Y. Song, J. Guo, Y. Zhang and J. Li.. "Context-oriented web video tag recommendation," In *World Wide Web Conference*. (2010)

[29] K. Filippova and K. B. Hall. "Improved video categorization from text metadata and user comments," In *34th ACM SIGIR Conference on Research and development in Information Retrieval* (2011)

[30] G. Chatzopoulou, C. Sheng and M. Faloutos, "A first step towards understanding popularity in YouTube," *IEEE INFOCOM,* (2010).

[31] J. W. Eerkens and C. P. Lipo. "Cultural Transmission Theory and the Archaeological Record: Providing Context to Understanding Variation and Temporal Changes in Material Culture," *Journal of Archaeological Research*. Vol. 15. Issue 3, pp 239-274. (2007).

[32] Nau, D., & Wilkenfeld, J. "Computational cultural dynamics," *Intelligent Systems, IEEE*, *23*(4), 18-19. (2008)

[33] M. A. Porter, J. P. Onnela and P. J. Mucha. "Communities in Networks," *North American Mathematical Society* 56: 1082–1097, 1164–1166. (2009)

[34] Carley, K. M. "Dynamic network analysis," In *Dynamic social network modeling and analysis: Workshop summary and papers* (pp. 133-145). Committee on Human Factors, National Research Council. (2003.)

[35] Snijders TA. "Models for longitudinal network data," *Models and methods in social network analysis*. Cambridge University Press, New York, pp 148–161. (1997)

[36] M. Friendly. "Corrgrams: Exploratory Displays for Correlation Matrices," *The American Statistician*, Vol. 56, No. 4, pp. 316-324. (2002)

[37] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak and Z. G. Ives, "DBpedia: A Nulclues of a Web of Open Data," in *Inter. Semantic Web Conference*, (2007).

[38] Mirizzi, R, Tommaso Di Noia, Azzurra Ragone, Vito Claudio Ostuni, and Eugenio Di Sciascio, "Movie Recommendation with DBpedia," In *IIR*, pp. 101-112. (2012).

[39] J. Lehmann, J. Schuppel and S. Auer, "Discovering Unknown Connections – the DBpedia Relationship Finder," in *Proc. of the 1st SABRE Conference on Social Semantic Web*, (2007).

[40] I. Horrocks, "Semantic web: the story so far," *Inter. Cross disciplinary conference on web applications*,(2007).

[41] E. Gabrilovich and S. Markovitch, "Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis," in *Inter. Joint Conference on Artificial Intelligence*, (2007)

[42] T. Peterson, S. Patwardhan and J. Michelizzi, "Wordnet::Similarity – Measuring the relatedness of Concepts," in *AAAI* (2004).

[43] R. Cilibrasi and P. Vitanyi, "The Google Similarity Distance," *IEEE Trans. On Knowledge and Data Engineering*, 19:3, pp. 370, (2007).

[44] Banfield, R. E., Hall, L. O., Bowyer, K. W., & Kegelmeyer, W. P. "A comparison of decision tree ensemble creation techniques," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *29*(1), 173-180. (2007).

[45] Herdagdelen, A., Ciaramita, M., Mahler, D., Holmqvist, M., Hall, K., Riezler, S., & Alfonseca, E. "Generalized syntactic and semantic models of query reformulation," In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval* (pp. 283-290). ACM. (2010)

[46] Booth, Joel, Barbara Di Eugenio, Isabel F. Cruz, and Ouri Wolfson. "Query Sentences as Semantic (Sub) Networks." In *Semantic Computing, 2009. ICSC'09. IEEE International Conference on*, pp. 89-94. IEEE, 2009.

[47] Croft, William, and David Alan Cruse. "Cognitive linguistics," *Cambridge University Press*, (2004).

[48] Popescu, Ana-Maria, Oren Etzioni, and Henry Kautz. "Towards a theory of natural language interfaces to databases." *Proceedings of the 8th international conference on Intelligent user interfaces*. ACM, 2003.

[49] Kaufmann, Esther, Abraham Bernstein, and Lorenz Fischer. "NLP-Reduce: A "naıve" but Domain-independent Natural Language Interface for Querying Ontologies." ESWC, 2007.

[50] Hu, Jian, Gang Wang, Fred Lochovsky, Jian-tao Sun, and Zheng Chen. "Understanding user's query intent with wikipedia." In *Proceedings of the 18th international conference on World wide web*, pp. 471-480. ACM, 2009.

[51] McClosky, David, Mihai Surdeanu, and Christopher D. Manning. "Event extraction as dependency parsing for bionlp 2011." *Proceedings of the BioNLP Shared Task 2011 Workshop*. Association for Computational Linguistics, 2011.

[52] Huang, Minhua, and Robert M. Haralick. "Identifying Patterns in Texts."*Semantic Computing, 2009. ICSC'09. IEEE International Conference on*. IEEE, 2009.

[53] Collobert, Ronan, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. "Natural language processing (almost) from scratch." *The Journal of Machine Learning Research* 12 (2011): 2493-2537.

[54] Finkel, Jenny Rose, and Christopher D. Manning. "Joint parsing and named entity recognition." *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics, 2009.

[55] Chater, Nick, Joshua B. Tenenbaum, and Alan Yuille. "Probabilistic models of cognition: Conceptual foundations." *Trends in cognitive sciences* 10.7 (2006): 287-291.

[56] Joy P. Guilford. "Way beyond the IQ," *Creative Education Foundation, NY.* (1977).

[57] John B. Carroll. "Human Cognitive Abilities," *Cambridge University Press, Cambridge* (1993).

[58] Kino Coursey and Rada Mihalcea. "Topic identification using Wikipedia graph centrality," In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers* (2009).

[59] M. Steyvers and J. B. Tenenbaum. "The Large-Scale Structure of Semantic Networks: Statistical Analyses and a Model of Semantic Growth," *Cognitive science*, *29*(1), 41-78. (2010).

[60] Andrew McCallum. "Efficiently inducing features of conditional random fields," In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence* (2002).

[61] Lotfi A. Zadeh. "Some reflections on soft computing, granular computing and their roles in the conception, design and utilization of information/intelligent systems," In *Soft Computing* 2(1): 23-25. (1998).

[62] Fei Sha and Fernando Pereira. "Shallow parsing with conditional random fields," In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1* (NAACL '03), Vol. 1. 134-141. (2003).

[63] Christopher D. Manning and Hinrich Schuetze. "Foundations of Statistical Natural Language Processing," *MIT Press.* (1999)

[64] Silviu Cucerzan. "Large-Scale Named Entity Disambiguation Based on Wikipedia Data," In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning,* pp 708-716, (2007)

[65] D. Croce, C. Giannone, P. Annesi, and R. Basili. "Towards open-domain semantic role labeling," In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* (pp. 237-246). (2011)

[66] MSQA. (2008). http://research.microsoft.com/en-us/downloads/88c0021c-328a-4148-a158-a42d7331c6cf/

[67] Charles A. Clarke, Nick Craswell, Ian Soboroff and Ellen M. Voorhees. Overview of the TREC 2011 Web Track (2011)

[68] Gabriel Cardona, Francesc Rossello, and Gabriel Valiente. "Comparison of Tree-Child Phylogenetic Networks," *IEEE/ACM Trans. Comput. Biol. Bioinformatics* 6, 4 (2009), 552-569

[69] V. Punyakanok, D. Roth and W. Yih. "The Importance of Syntactic Parsing and Inference in Semantic Role Labeling," *Computational Linguistics*. (2008)

[70] I. Gurevych, Rainer Malaka, Robert Porzel, and Hans-Péter Zorn. "Semantic coherence scoring using an ontology. "In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1* (NAACL '03), Vol. 1. 6. (2003).

[71] Dangalchev Ch. "Residual Closeness in Networks, " *Phisica A* **365**, 556 (2006)

[72] Korf, Richard E. "Depth-first iterative-deepening: An optimal admissible tree search," *Artificial intelligence* 27.1 (1985): 97-109

[73] E. Gabrilovich and S. Markovitch, “Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis,” in *Inter. Joint Conference on Artificial Intelligence*, (2007)

[74] Hatcher, E., Gospodnetic, O., & McCandless, M. " Lucene in action," *Manning Publications Co.*(2004)

[75] Yining Wang, Liwei Wang, Yuanzhi Li, Di He, Wei Chen, Tie-Yan Liu., "A Theoretical Analysis of NDCG Ranking Measures," *In Proceedings of the 26th Annual Conference on Learning Theory*. (2013)

[76] H. A. Simon. "The Sciences of the Artificial (3rd ed.)," Cambridge, MA: *The MIT Press*. (1996)

[77] T. H. Davenport and J. C. Beck. "The Attention Economy: Understanding the New Currency of Business," *Harvard Business School Press*. (2011)

[78] Arpita Ghosh and Preston McAfee, "Incentivizing high-quality user-generated content," In *Proceedings of the 20th international conference on World wide web* (WWW '11). ACM, New York, NY, USA, 137-146 (2011).

[79] Azaria Paz. "Introduction to Probabilistic Automata (Computer Science and Applied Mathematics). *Academic Press, Inc.,* Orlando, FL, USA. (1971)

[80] B. A. Huberman, D. A. Romero and F. Wu. 2008. "Social networks that matter: Twitter under the microscope," In *Computing Research Repository - CORR*, vol. abs/0812.1, no. 1, (2008)

[81] Janette Lehmann, Bruno Gonçalves, José J. Ramasco, and Ciro Cattuto. "Dynamical classes of collective attention in twitter," In *Proceedings of the 21st international conference on World Wide Web* (2012)

[82] L. Backstrom, E. Bakshy, J. Klienberg, T. M. Lento and I. Rosenn. "Center of attention: How facebook users allocate attention across friends," In *Inter. Conf. on Weblogs and Social Media (*2011)

[83] L. Weng, A. Flammini, A. Vespignani and F. Menczer. "Competition among memes in a world with limited attention," In *Nature*, Scientific Reports 2, Article No. 335, (2012).

[84] C. Wagner, M. Rowe, M. Strohmaier and H. Alani. "What Catches Your Attention? An Empirical Study of Attention Patterns in Community Forums," In *Inter. Conf. on Weblogs and Social Media (*2012)

[85] N. O. Hodas and K. Lerman. "How Visibility and Divided Attention Constrain Social Contagion," In *SocialCom* (2012).

[86] Li Yujian and Liu Bo.." A Normalized Levenshtein Distance Metric, " *IEEE Trans. Pattern Anal. Mach. Intell.* 29, 6 (June 2007), 1091-1095

[87] Marshall A. Kuypers, Walter E. Beyeler, Robert J. Glass, Matthew Antognoli, and Michael D. Mitchell. "The impact of network structure on the perturbation dynamics of a multi-agent economic model," In *Proceedings of the 5th Inter. Conf. on Social Computing, Behavioral-Cultural Modeling and Prediction* (2012)

[88] Griffiths, T., et al. "Hierarchical topic models and the nested Chinese restaurant process." *Advances in neural information processing systems* 16 (2004): 106-114.

[89] Blei, David M. "Probabilistic topic models." *Communications of the ACM* 55.4 (2012): 77-84.

[90] Dumais, Susan T. "Latent semantic analysis." *Annual review of information science and technology* 38.1 (2004): 188-230.

[91] Rosario, Barbara. "Latent semantic indexing: An overview." *Techn. rep. INFOSYS* 240 (2000).

[92] Moon, Todd K. "The expectation-maximization algorithm." *Signal processing magazine, IEEE* 13.6 (1996): 47-60.

[93] Szpektor, Idan, Aristides Gionis, and Yoelle Maarek. "Improving recommendation for long-tail queries via templates." *Proceedings of the 20th international conference on World wide web*. ACM, (2011).

[94] Bollacker, Kurt, et al. "Freebase: a collaboratively created graph database for structuring human knowledge." *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. ACM, (2008).

[95] Wallach, Hanna M. "Topic modeling: beyond bag-of-words." *Proceedings of the 23rd international conference on Machine learning*. ACM, (2006).

[96] Li, Ching Chun. "Path analysis-a primer," *The Boxwood Press*., (1975).

[97] Sakaki, Takeshi, Makoto Okazaki, and Yutaka Matsuo. "Earthquake shakes Twitter users: real-time event detection by social sensors." *Proceedings of the 19th international conference on World wide web*. ACM, (2010).

[98] Achrekar, Harshavardhan, Avinash Gandhe, Ross Lazarus, Ssu-Hsin Yu, and Benyuan Liu. "Predicting flu trends using twitter data." In *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on*, pp. 702-707. IEEE, (2011) .

[99] Biever, Celeste. "Twitter mood maps reveal emotional states of America." *New Scientist* 207.2771 (2010): 14.

[100] Bishop, Christopher M., and Nasser M. Nasrabadi. *Pattern recognition and machine learning*. Vol. 1. New York: springer, (2006).

[101] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." *the Journal of machine Learning research* 3 (2003): 993-1022.

[102] Golub, Gene H., and Christian Reinsch. "Singular value decomposition and least squares solutions." *Numerische Mathematik* 14.5 (1970): 403-420.

[103] Hoffman, Matthew, Francis R. Bach, and David M. Blei. "Online learning for latent dirichlet allocation." *advances in neural information processing systems*. (2010).

[104] Roy, Suman Deb, Wenjun Zeng, Tao Mei and Shiping Li "SocialTransfer: cross-domain transfer learning from social streams for media applications." *Proceedings of the 20th ACM international conference on Multimedia*. ACM, (2012).

[105] Roy, Suman Deb, Wenjun Zeng, Tao Mei and Shiping Li "Towards Cross-Domain Learning for Social Video Popularity Prediction." *IEEE Transactions on Multimedia* (2013).

[106] Lin, Frank, and William W. Cohen. "Power iteration clustering." *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*. (2010).

[107] D. Ramage, S. T. Dumais and D. J. Liebling. "Characterizing microblogs with topic models," In *4th Inter. AAAI Conf. on Weblogs and Social Media*. (2010)

[108] Chang, Chih-Chung, and Chih-Jen Lin. "LIBSVM: a library for support vector machines." *ACM Transactions on Intelligent Systems and Technology (TIST)*2.3 (2011): 27.

[109] Mislove, A., Marcon, M., Gummadi, K. P., Druschel, P., & Bhattacharjee, B. "Measurement and analysis of online social networks." In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*(pp. 29-42). ACM. (2007).

[110] D. Newman, S. Karimi and L. Cavedon, "External evaluation of topic models," *Proc. of ADCS 2009*, pp. 11-18, (2009).

[111] D. Newman, J. H. Lau, K. Grieser and T. Baldwin, "Automatic Evaluation of Topic Coherence," *Annual Conf. of the North American Chapter of the ACL*, (2010).

[112] Kathy Lee, Diana Palsetia, Ramanathan Narayanan, Md. Mostofa Ali Patwary, Ankit Agrawal, and Alok Choudhary. "Twitter Trending Topic Classification," In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining*. (2011)

[113] Ojo, I. F. "Autoregressive Integrated Moving Average." *Asian Journal of Mathematics and Statistics* 3.4 (2010): 225-236.

[114] S. J. Pan and Q. Yang. "A survey on transfer learning" IEEE Trans. on Knowledge and Data Engineering, 22:1345–1359, (2010).

[115] Wang, Chong, John W. Paisley, and David M. Blei. "Online variational inference for the hierarchical Dirichlet process." *International Conference on Artificial Intelligence and Statistics*.(2011).

VITA

Suman Deb Roy is a Data Scientist at Betaworks in NYC. His job involves building the popularity ranking framework and architecture of Instapaper and Digg, data driven approaches to scale media applications, detect profitable seed investments and develop novel models for understanding user behavior patterns on the social web. Being a combination of a scientist, developer and hacker, Suman loves large-scale machine learning, data visualization, network analysis and predictive modeling that helps in interpreting the patterns and relationships mined from data to people in product development and marketing. His current research interests include social media computing, targeted social advertising, semantic analysis of natural languages and micro-blogs, transfer learning and the applications of game theory in multi-agent and cross-domain systems uniting data from several media platforms for collaborative intelligence.

Over the past few summers, Suman has interned with Huawei- NJ, Microsoft Research and SocialFlow Inc., NY developing various algorithms for multimedia encryption, topic modeling from social streams (Twitter) and predictive modeling of various characteristics of geo-spatial social trends (e.g., how they will last within a user community). He is the bulletin editor for IEEE Special Technical Community on Social Networking.

Suman is also a PhD Candidate in the Dept. of Computer Science at the University of Missouri and a former Fellow of the Reynolds Journalism Institute (RJI) at the Missouri School of Journalism. He was recently awarded the Missouri Honor Award for Outstanding PhD student in 2013. He also holds a Bachelor of Technology in Computer Science from National Institute of Technology, Durgapur, India.