Qualifier Exam for Debjyoti Paul

Problem 1

Social media data is enormous, but semi-private. List relevant social media data sources, and explain what is known about their sizes (in terms of storage space, and number of records), including both what is (probably) privately controlled by companies, and what is available for sufficiently-motivated and -resourced academic researchers.

Explain the state-of-the-art (with references to research papers) for scraping such semi-publicly accessible data sets, and what are the largest bottlenecks for such tasks.

Predict (using a machine learning / data mining techniques on the data above) what the total number of social media records available to researchers will be in 2022.

Social Media Data and its availability in Research

Debjyoti Paul
University of Utah
deb@cs.utah.edu

OVERVIEW

The term *social media* gained attention with the advent of Web 2.0 in the first decade of 20th century [14]. Web 2.0 is also known as *Participative* or *Social Web* that emphasize on user interaction and user generated content encouraging participatory culture. Before we jump into more details of social media it would be wiser to define it. Though ever evolving social media services makes it hard to define them, most of the research work define it as follows.

Definition 1 (Social Media). Social media are interactive computer-mediated technologies that facilitate the creation and sharing of information, ideas, career interests and other forms of expression via virtual communities and networks [15].

In contrast to the *traditional media* which operates under a monologic transmission model i.e. one source to many receivers, such as a television, newspaper or a radio station which broadcasts the same programs to an entire city; *social media* are dialogic transmission system which brings interaction, usability and a notion of individual entity in digital world.

1 PART A

Social Media Data in Numbers

Marketing and social media experts broadly agrees to classify social media with respect to media type and its usage i.e *blogs, social networks, private messaging, microblogs, photo sharing, video sharing, professional networks, enterprise social networks, forums, products/services review, social bookmarking, social gaming, collaborative projects and virtual worlds* [1]. We now present a list of relevant social media according to the classification stated in Table 1.

Table 1: List of Relevant Social Media

Category	Social media sites with link
Social Networks	Facebook, Snapchat, WeChat, Quora
Private Messaging	Messenger, Whatsapp, QQ, WeChat, Skype
Microblogs	Twitter, Sina Weibo, Tumblr
Photo Sharing	Instagram, Photobucket, Flickr
Video Sharing	Youtube, Vimeo, Dailymotion
Professional Networks	LinkedIn, AngelList, Meetup
Enterprise Social Networks	Workday
Blogs	Wordpress, Medium, Buffer Blog
Forums	Reddit, Hacker News, Quora
Products/Services Review	Yelp, Foursquare, Google Places
Social Bookmarking	Pinterest, Digg, Stumble Upon Mix
Social Gaming	Pokemon Go, IGN, Gamespot [20]
Collaborative Projects	Slack, Invision, Trello, Github, Bitbucket
Social Gaming	Friendster

The popularity of a social media site is primarily determined by the total number of users or monthly active users. Table 2 presents

Table 2: Social media sites and number of users (in millions).

		Years						
Category	**Site**	**2013**	**2014**	**2015**	**2016**	**2017**	**2018**	**Type**
Social Networks	Facebook	1228	1393	1591	1860	2129	2271	Total
	WeChat	355	500	697	889	989	1082	Total
Microblogs	Twitter	241	284	305	318	330	332	Active
	Weibo	140	175	237	310	340	392	Active
	Tumblr	175	–	–	–	460	550	Total
Photo Sharing	Instagram	150	300	460	600	870	1000	Active
	Snapchat	33	100	180	301	–	400	Total
Video	Youtube	700	1100	1431	1618	1767	1900	Active
Professional	LinkedIn	277	347	414	467	530	576	Total
Services	Yelp	96	135	150	158	170	178	Active
	Foursquare	33	30	50	–	–	55	Total
	Ridesharing	–	–	208	272	338	400	Active
Bookmarking	Pinterest	–	–	110	160	220	250	Total

Table 3: Social media sites and media units created per day (in millions).

		Years						
Category	**Site**	**2013**	**2014**	**2015**	**2016**	**2017**	**2018**	**Unit**
Social	Facebook	3600	–	–	4320	–	–	posts
Microblogs	Twitter	245	399	500	–	657	682	tweets
	Tumblr	120*	205*	270*	315*	380*	448*	total blogs*
Photo Sharing	Instagram	5	31	–	–	–	67	photos
	Snapchat	–	–	–	–	760	3000	photos
Video	Youtube	69,120†	103,680†	432,000†	576,000†	720,000†	–	hours video †
Services	Yelp	40‡	55‡	75‡	95‡	135‡	171‡	total reviews‡
	Foursquare	33	–	–	–	–	12000§	total checkins§
Bookmarking	Pinterest	–	5	13	–	–	–	pins

facts about social media sites user base which gives some sense of its popularity [7, 8, 18, 21]. The attribute *type* with values *(a) Total (b) Active* represents whether the statistic is of total users or active monthly users respectively.

Other than the social media sites mentioned in Table 2 there are some significant sites where only the current user statistics are available. For example Flickr, the photo sharing platform has 90 million users. Quora, a question answer social platform has 300 million users worldwide. Reddit, a social forum has 330 million active users.

Number of users is not just important to measure the popularity of a social media site but also to estimate the amount of data storage it maintains. Another feature that will help us to estimate data storage is the amount of media units (e.g. posts, photos, microblogs, videos etc.) ingested per day. Table 3 presents all the statistics of relevant social media from open internet [7, 8, 21, 24]. The statistics for social media sites missing in Table 3 but mentioned in Table 2 are almost impossible to find in open internet.

Social Media Storage Estimate:

Social media sites seldom reveals the amount of data they store or ingest on daily basis. Also the ever growing social media makes it hard to estimate their storage capacity. I present few methods in the following section to estimate social media storage.

University of Utah, Salt Lake City, USA.

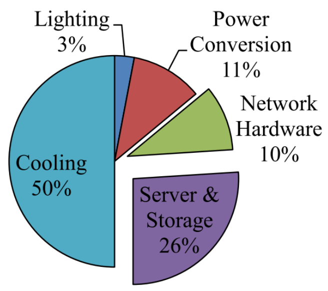


Figure 1: A typical breakdown of energy usage among components in data center [12].

1. Storage space estimate from media units: This method works for all the social media sites metioned in Table 3 where the approximate storage space required by media unit is known.

Youtube: Lets take an example of Youtube video data. From the Table 3 we find by year 2017 users upload 720,000 hours of video in Youtube. First, assuming the fact that Youtube pretty much stores almost video in 1080p and it stores video in multiple resolution such as 240p, 360p, 720p, 1080p and format e.g. Webm, flv, mp4, 3gp, mp3. We can determine the amount of storage space needed for a 1 minute video [13].

$$\begin{aligned}&27.71\text{ MB (Webm)} + 17.00\text{ MB (flv)} + 554.43\text{ KB (3gp)}\\&+ 45.80\text{ MB (mp4)} + 2.81\text{ MB (mp3)}\\&= 93.8614355\text{ MB}\end{aligned} \tag{1}$$

From the above we find that $720,000 \times 60 \times 93.8614355 \approx 4.055$ petabytes (PB) of storage space is required by Youtube everyday. We can also calculate the total amount of storage space ingested during the period of 2013 to 2017 from Table 3 by utilizing area under the curve method with interpolation. The above method reckons 3096.17 PB or 3.096 exabytes (EB) of storage. Considering videos before 2013 and new 4K video which takes more space it can be easily assumed that Youtube use 10-15 EB storage space.

Twitter: Similar to the method above we can find the space required to store a tweet. A tweet is stored in Twitter as UTF-8 format. This takes 140 characters tweets atmost 560 bytes of space. However the metadata attached with a tweet is much more than the tweet itself. I personally did a random sample experiment of 100K tweets stord in our databases to find the average storage space for tweet json object obtained from streaming api. I find one json tweet object takes 3247 bytes of space in average. 682 million tweets per day will require around 2.2145 terabytes of data per day. Using the interpolation method for area under the curve we can find that Twitter use 3.13 petabyte of space for storing the tweet alone. It is also worth noting that 42% of tweets contains images [23]. If we assume the average image size be 100 KB then we will see $(100 * 1024)/3247 * 42\% \approx 13.2$ times increase in storage space requirement.

2. Storage space estimate from data center power usage: This section presents an approximate method to estimate space capacity of large social media companies like Facebook and Google. A typical breakdown of energy consumption by data center given in Figure 1. The largest energy consuming component is cooling infrastructure with 50% of total energy. Rest of the energy is used by power conversion, lighting, network and server components [6, 12]. Facebook data centers use efficient data center architecture and hardware tweaks saves 8-12% of energy spent in cooling, 13-25% in power conversion, 10% in motherboard [11]. That implies atmost 11% more efficient than typical data centers. Hence, it can be claimed that Facebook servers use 37% of energy. Considering Facebook's 138 MW Altoona data center equipped with 200 Watts servers each with 6×4 TB of HDD as used in their experiment for [11]. Assuming the datacenter is running at peak energy $(138 \times 0.37 \times 24)/200 = 6127200$ TB $= 6.1272$ exabytes (EB). Taking all the data centers in consideration and diving them with replication factor we can estimate the storage capacity of Facebook. The analysis provided above supports news *Facebook Builds Exabyte Data Centers for Cold Storage* in 2013 [5].

Social Media Data for Researchers:

Regardless of the vast data in social media sites. the dataset availables for researchers in public domain is very very limited. Also these datasets size are miniscule in comparison to what we mean by bigdata. The only exception is Twitter. Twitter provides $\tilde{1}$% of sample tweets through its streaming API. By utilizing multiple resources and some other APIs such as keyword search researchers can obtain more than 1% sample data. Also it is noteworthy that researchers looking for geotagged data face greater challenge as only 0.85% of tweets in Twitter are geotagged [19]. A study on the sample tweets and orginial stream (firehose) reveals that the research on sample and original can differ unless proper coverage is taken care of during data collection strategy [16]. From the previous analysis and checking our twitter streaming collection, we can estimate that $\tilde{1}$% sample collects 25-30 GB of uncompressed data daily.

Facebook has tighten the security and restricted access to many of its data for public research after Cambridge Analytical Scandal [4, 10]. However, Facebook launched an initiative to make a dataset available to *The Social Science Research Council* for assessment on impact of social data on election [9]. That means only affiliated researchers with certain agencies will be able to access Facebook's data. I believe we will continue to see restrict access behavior from similar social media sites in future which can affect public researchers.

To sum up, I present some of the most relevant social media dataset available for public research in Table 4. From the Table 4 it is clear that there is no relationship between the amount of data social media sites possesses and the data available for researchers in public domain.

Many social media sites expose APIs for developers to access data. The free APIs of all the relevant social media sides are very restrictive. For example, facebook allows 200 api requests per hours/user. Instagram earlier had 5000 requests/hour which has been reduced to 200 request/hour. Geolocation service like foursquare 500 requests/hour on premium api end points. Hence, it is clear that availability of social media data in public domain is not only subject to effort we invest in collecting it but also restrictive policies

Table 4: Most relevant social media dataset.

Site	Dataset	Size	Link
Network Repository	Frienster	8 GB	link
	Twitter (1)	6 GB	link
	Twitter (2)	6 GB	link
	Twitter (3)	960 MB	link
	Orkut (1)	388 MB	link
	Orkut (2)	422 MB	link
	Sina Weibo	960 MB	link
Stanford SNAP	Facebook (ego)	4,039 nodes	link
	Google Plus	107,614 nodes	link
	Twitter Social	81,306 nodes	link
	Expinion	75,879 nodes	link
	Youtube	1,134,890 nodes	link
	Amazon Product	334,863 nodes	link
	Reddit	132,308 submissions	link
	Flickr	2,316,948 images	link
	BrightKite (Location)	58,228 Nodes	link
	Gowalla (Location)	196,591 Nodes	link
	Movies	196,591 Nodes	link
Social Computing ASU	Youtube (1)	1,138,499 nodes	link
	Youtube (2)	15088 nodes	link
	Last FM	108,493 nodes	link
	Twitter	11,316,811 tweets	link
	Flickr	80,513 nodes	link
	Foursquare	106,218 nodes	link
	Digg	116,893 nodes	link
	Delicious	103,144 nodes	link
Sentiment 140	Twitter Sentiment	160,000 tweets	link
Reddit	Reddit	1.7 billion comments	link
Yahoo	Flickr	100 million images	link
Awesome Data Github	Google Scholar	Unknown	link
	Indie Map	Unknown	link

from companies. We will revisit about scraping challanges in next section.

2 PART B.

Scraping Social Media:

Social media data is broadly divided into [3] :
1. Historic datasets: Previously accumulated and stored social/news, financial and economic data.
2. Realtime feeds: Live data feeds from streamed social media, news services, financial exchanges, telecom services, GPS devices and speech.

Historical datasets are relatively easy to collect than real-time feeds because of API limitation and limitation of scraping via crawling webpages. Social media data is mainly collected via two procedure API based or web crawling based approach. API crawling methods are easy to maintain and modifiable. Web crawling based approach can extract more information which might not be available via APIs. Also web crawling can avoid API rate limit and can crawl more data. But it needs data clean up and high maintenance because the web interface can change quickly which result in change of code and crawling procedure.

Open source projects on API libraries available in Github and other collaborative platforms enables researchers to collect data e.g. tweepy for Twitter, pyFacebook, python-flickr, foursquare-api, uberpy, python-vimeo, youtube-api etc. (all available in Github.) A comprehensive list of api wrappers can be obtained from here [2].

Due to the imposed restriction of APIs crawling technolgies have evolved rapidly. Few state-of-the-art crawling libraries and utilities are scrapy [1], beatifulSoup4 [2], selenium [3], Graphene [4].

Difficulties in Scraping:

In this section, we present comprehensive list of the challenges researchers face while scraping social media data.

Lack of free APIs: Many social media sites does not expose relevant APIs for free. They are mostly paid APIs and are costly.

API call limit: API call limit is the most discouraging element for scraping social media data. Sometimes they even show erratic behavior even if the scraper have not reached documented rate limit e.g. foursquare APIs.

Javascript enabled data: Researchers resorted to web crawling face this challenge when the data is fetch via javascript calls and simple web crawling will not work. This kind of scenario needs expertise and technologies. Using libraries like scrapy with plugin splash (scriptable browser) [5] is the only best method known to researchers. But it slows down scraping process.

IP block: Web crawling often result in IP block. Only a well resourced researcher can avoid it by charging IP addresses often through proxy services. But sometimes social media sites can even detect proxies and limit access.

Legal Terms and Conditions: Web crawling sometimes violate the legal terms and conditions from social media sites. Even if a researcher gets the data by crawling he/she might not be able to publish the data or share with other researchers.

Missing link metadata: It is often found that data exposed either in API or in Webpages miss link metadata. Link metadata are information like followers which reduce usability of such data.

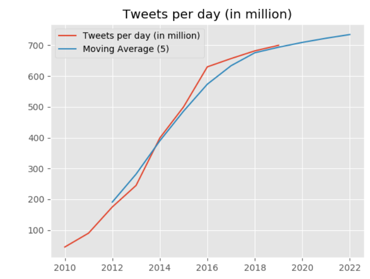
3 PART C.

Predicting future:

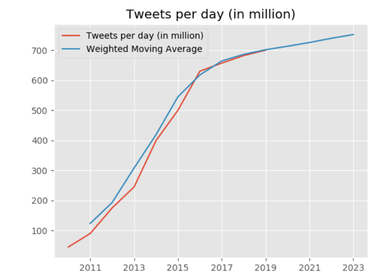
As mentioned in Section 1 that there is no true evident relation between the amount of social media data with companies and data available for public research except for Twitter. We will only attempt to predict the amount of data will be available for Twitter in 2022. Also using the same technique we will try to predict how much social media data will be available in 2022 within the companies. Also the fact that I have limited amount of information from Table 2 and Table 3 it will be wise to start with a simple forecasting methods. Also it is very common for extremely simple methods like forecast with historical average to outperform more complex methods [22]. This statement is even more likely true for short time series.

To start with the basic forecasting model we might consider is a historical *mean model* which assumes that the time series consists of independently and identically distributed ("i.i.d.") values, as if

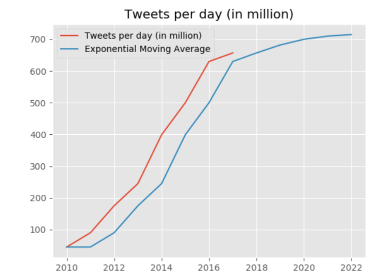
[1]github.com/scrapy/scrapy
[2]github.com/il-vladislav/BeautifulSoup4
[3]pypi.org/project/selenium/
[4]graphene-python.org/
[5]github.com/scrapy-plugins/scrapy-splash



(a) Moving Average with window 5.

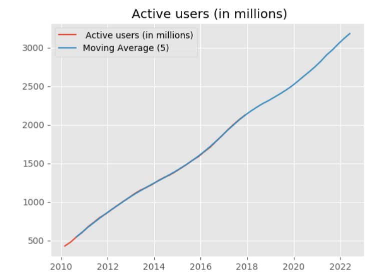


(b) Weighted Moving Average with window 3 (0.5, 0.3, 0.2).

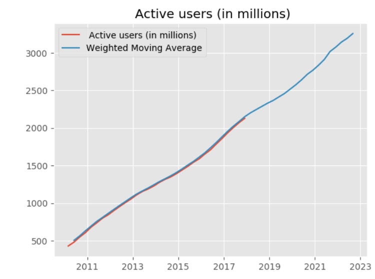


(c) Exponential Moving Average with $\alpha = 0.9$.

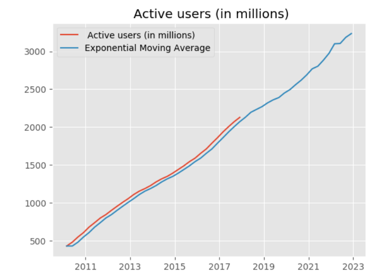
Figure 2: Predicting tweets per day from 2019-2022.



(a) Moving Average with window 5.

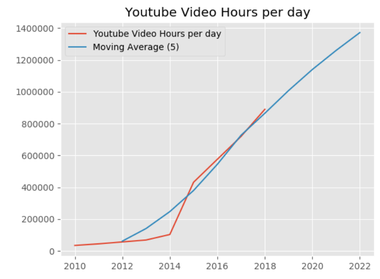


(b) Weighted Moving Average with window 3 (0.5, 0.3, 0.2).

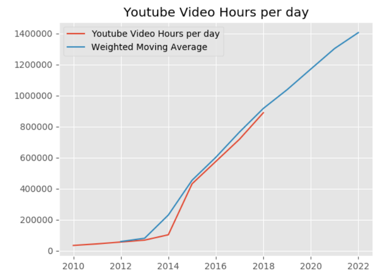


(c) Exponential Moving Average with $\alpha = 0.9$.

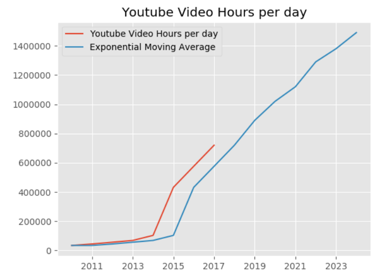
Figure 3: Predicting Facebook monthly average users from 2019-2022.



(a) Moving Average with window 5.



(b) Weighted Moving Average with window 4 (0.5, 0.35, 0.05,0.05).



(c) Exponential Moving Average with $\alpha = 1.0$.

Figure 4: Predicting Youtube video upload hours from 2019-2022.

each observation is randomly drawn from the same population. Under this assumption, the next value should be predicted to be equal to the historical sample mean if the goal is to minimize mean squared error. I tried a bunch of experiment with *mean model* and it appeared to be working well with relatively moderate mean squared error.

I avoided linear trend model as it is not a very *robust* model for time-series forcasting [17]. Since I have the priori knowledge of the series has a positive trend or zero trend I can use a moving average model that puts more weight on the most recent values than to use a linear trend model with a *not very significant* trend estimate. With this notion I tried the *moving average model* to forecast the tweets per day metric for 2022. I tried more forecasting methods with *weighted moving average* and *exponential moving average model* . Figure 2 presents all the forecasts for tweets per day metric till 2022 [6].

Complex models like *ARMA (AutoRegressive Moving Average)* and *ARIMA (AutoRegressive Integrated Moving Average)* models did not do well (validating with small train test data).

From the prediction in Figure 2 we find that twitter will generate around 730 million tweets every day in average. That is almost 2.4 terabytes of tweet data withput media.

[6]github.com/debjyoti385/MovingAverageForecasting

Similar forecasting applied on Facebook monthly active users in Figure 3 reveals that almost 3000 million people will be active in Facebook by the end in 2022. From Youtube's hourly video upload data predicts that in 2022 users will upload 1.4 million hours of video everyday 4.

REFERENCES

[1] T. Aichner and F. Jacob. Measuring the degree of corporate social media use. *International Journal of Market Research*, 57(2):257–276, 2015.
[2] APIWrappers. List of api wrappers, 2018. https://github.com/realpython/list-of-python-api-wrappers, [Online; accessed 27-Nov-2018].
[3] B. Batrinca and P. C. Treleaven. Social media analytics: a survey of techniques, tools and platforms. *Ai & Society*, 30(1):89–116, 2015.
[4] BBC. Cambridge analytica facebook scandal, 2018. https://www.nytimes.com/2018/04/04/us/politics/cambridge-analytica-scandal-fallout.html, [Online; accessed 27-Nov-2018].
[5] DataCenterKnowledge. Data center knowledge, facebook builds new exabyte data center, 2018. https://www.datacenterknowledge.com/archives/2013/01/18/facebook-builds-new-data-centers-for-cold-storage , Online; accessed 27-Nov-2018.
[6] M. Dayarathna, Y. Wen, and R. Fan. Data center energy consumption modeling: A survey. *IEEE Communications Surveys & Tutorials*, 18(1):732–794, 2016.
[7] Domo. Domo, 2018. [Online; accessed 27-Nov-2018].
[8] ExpandedRamblings. Expanded ramblings, 2018. https://expandedramblings.com,[Online; accessed 27-Nov-2018].
[9] FacebookNewsroom. Facebook launches new initiative to help scholars assess social mediafis impact on elections, 2018. https://newsroom.fb.com/news/2018/04/new-elections-initiative/, [Online; accessed 27-Nov-2018].
[10] FacebookNewsroom. An update on our plans to restrict data access on facebook, 2018. https://newsroom.fb.com/news/2018/04/restricting-data-access/, [Online; accessed 27-Nov-2018].
[11] E. Frachtenberg, A. Heydari, H. Li, A. Michael, J. Na, A. Nisbet, and P. Sarti. High-efficiency server design. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, page 27. ACM, 2011.
[12] I.-T. R. Group et al. Top 10 energy-saving tips for a greener data center. *Info-Tech Research Group. fi...*, 2007.
[13] jdownloader. Youtube jdownloader, 2018. http://i.imgur.com/CgX5t.png, Online; accessed 27-Nov-2018.
[14] A. M. Kaplan and M. Haenlein. Users of the world, unite! the challenges and opportunities of social media. *Business horizons*, 53(1):59–68, 2010.
[15] J. H. Kietzmann, K. Hermkens, I. P. McCarthy, and B. S. Silvestre. Social media? get serious! understanding the functional building blocks of social media. *Business horizons*, 54(3):241–251, 2011.
[16] F. Morstatter, J. Pfeffer, H. Liu, and K. M. Carley. Is the sample good enough? comparing data from twitter's streaming api with twitter's firehose. In *ICWSM*, 2013.
[17] R. Nau. Review of basic statistics and the simplest forecasting model: the sample mean. *Fuqua School of Business, Duke University*, 2014.
[18] NetImperative. Sina weibo overtakes twitter user numbers, 2018. http://www.netimperative.com/2017/05/chinas-weibo-overtakes-twitter-user-numbers/, [Online; accessed 27-Nov-2018].
[19] L. Sloan, J. Morgan, W. Housley, M. Williams, A. Edwards, P. Burnap, and O. Rana. Knowing the tweeters: Deriving sociologically relevant demographics from twitter. *Sociological research online*, 18(3):1–11, 2013.
[20] Statista. Most popular games in us, 2018. https://www.statista.com/statistics/580150/most-popular-us-gaming-apps-ranked-by-audience/, [Online; accessed 27-Nov-2018].
[21] Statista. Statista, 2018. https://www.statista.com,[Online; accessed 27-Nov-2018].
[22] StatsStackExchange. Is it unusual for the mean to outperform arima, 2018. https://stats.stackexchange.com/questions/124955/is-it-unusual-for-the-mean-to-outperform-arima, [Online; accessed 27-Nov-2018].
[23] Thenextweb. Next web, 2018. [Online; accessed 27-Nov-2018].
[24] Zephoria. Facebook statistics, 2018. [Online; accessed 27-Nov-2018].

Problem 2

Deep reinforcement learning is a useful technique for realizing goal-oriented algorithms. It combines the idea of reinforcement algorithm with deep learning and has been shown to be very effective in many different application scenarios. Answer the following questions:

1) Please provide a detailed review of deep reinforcement learning.

2) How would you apply deep reinforcement learning for tuning the performance of a DB system? (i.e., like what OtterTune does, but using deep reinforcement learning). Please outline your problem formulation, overview of your approach, and a sketch analysis.

Deep Reinforcement Learning and Self Instructing Database

Debjyoti Paul
University of Utah
deb@cs.utah.edu

INTRODUCTION

Reinforcement learning (RL) is the area of machine learning that deals with *sequential decision making*. Sequential decision-making is a task of deciding, the sequence of actions to perform in an uncertain environment in order to achieve some goals. Sequential decision-making tasks cover a wide range of possible applications with the potential to impact many domains, such as robotics, healthcare, smart grids, finance, self-driving cars, and many more.

Inspired by behavioral psychology (e.g. Sutton [21]) reinforcement learning (RL) proposes a formal framework to this problem. The main idea is that an artificial agent may learn by interacting with its environment, similarly to a biological agent. I will dive more into the details how a general RL problem is formally represented and what are key components involved in the framework in Section 1.

1 PART A

In reinforcement learning there is no supervisor but a *reward* signal to guide the learning process. Even the feddback is delayed and not instantenous which sets it as a different paradigm from other machine learning methods. In a sequential decision making process an/the *agent* get to pick an action at every time step which tries to optimize the future cumulative *reward* signal. Traditional Reinforcement Learning requires explicit design of state space and action space, while the mapping from state space to action space is learned [22]. That makes the problem spaces and the possible states in an environment very limited, only with fully observable state of environment for agent. Neural networks enhances RL with the capability to make the agent learn a state abstraction and a policy approximation directly from its input data in a partially observable environment. Deep RL is the fruitful outcome of this attempt of RL enhancement.

1.1 Formal RL Framework:

The general RL problem is formalized as a discrete time stochastic control process where an agent interacts with its environment in the following way: the agent starts, in a given state within its environment $s_0 \in \mathcal{S}$, by gathering an initial observation $\omega_0 \in \Omega$. At each time step t, the agent has to take an action $a_t \in \mathcal{A}$. As illustrated in Figure 1 it follows three consequences: (i) the agent obtains a reward $r_t \in \mathcal{R}$, (ii) the state transitions to $s_{t+1} \in \mathcal{S}$, and (iii) the agent obtains an observation $\omega_{t+1} \in \Omega$. This control setting was first proposed by [3] and later extended to learning by Barto [2]. The goal of RL is to *select actions to maximise total future reward.*

Definition 1. Reward: A reward r_t is a scalar feedback signal that indicates how well *agent* is doing at step t.

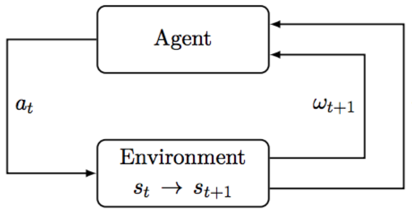


Figure 1: Agent-environment interaction in RL.

An example of *reward* in the case of flying toy robot helicopters are : (a) +ve reward for following desired trajectory, (b) fk??ve reward for crashing.

Definition 2. History: The history $\mathcal{H}_t$ is the sequence of observations, actions, rewards, i.e. all observable variables up to time t. Mathematically, $\mathcal{H}_t = \omega_1, r_1, a_1, \ldots, a_{t-1}, \omega_t, r_t$.

Formally, state $s_t \in \mathcal{S}$ is a function of the history $\mathcal{H}_t$.

$$s_t = f(\mathcal{H}_t)$$

Environment and agent both can have threr own state which we define as s_t^e and s_t^a respectively. The environment state is usually not visible to the agent.

In a fully observable system, the agent directly observes environment state i.e. $\omega_t = s_t^a = s_t^e$. Formally this is called Markov Decision Process (MDP). The Markov property means that the future of the process only depends on the current observation, and the agent has no interest in looking at the full history.

Markov Decision Process (MDP):
An MDP is a 5-tuple $(\mathcal{S}, \mathcal{A}, T, \mathcal{R}, \gamma)$ where

$\mathcal{S}$ is the state space,
$\mathcal{A}$ is the action space,
$T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the transition function (set of conditional transition probabilities between states),
R: $\mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathcal{R}$ is the reward function, where R is a continuous set of possible rewards in a range $R_{max} \in \mathbb{R}^+$ (e.g., $[0, R_{max}]$),
$\gamma \in [0, 1)$ is the discount factor.

However, in the real world not many sytems are fully observable, an *agent* indirectly observes the environment without knowing it's internal information state. In this scenario $s_t^a \neq s_t^e$.

Formally this is a partially observable Markov decision process (POMDP) illustrated in Figure 2.

Partially Observable Markov Decision Process (POMDP): A POMDP is a 7-tuple $(\mathcal{S}, \mathcal{A}, T, \mathcal{R}, \Omega, O, \gamma)$ where

$\mathcal{S}$ is the state space,
$\mathcal{A}$ is the action space,
$T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the transition function (set of conditional transition probabilities between states),

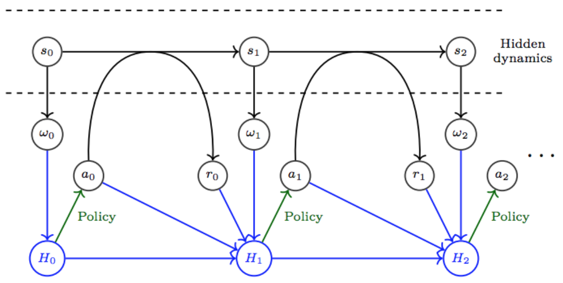


Figure 2: Partial Observed Markov Decision Process (POMDP).

R: $\mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathcal{R}$ is the reward function, where R is a continuous set of possible rewards in a range $R_{max} \in \mathbb{R}^+$ (e.g., $[0, R_{max}]$),
Ω is a finite set of observations $1, ..., N_\Omega$,
$O : \mathcal{S} \times \Omega \rightarrow [0, 1]$ is a set of conditional observation probabilities,
$\gamma \in [0, 1)$ is the discount factor.

In a POMDP *agent* must construct its own state representation s^a_t. For example,
(i) **Complete history:** $s^a_t = \mathcal{H}_t$.
(ii) ***Beliefs* of environment state:** $s^a_t = \mathbb{P}[s^e_t = s^1], \ldots, \mathbb{P}[s^e_t = s^n]$ where s^k represents a state in environment.
(iii) **Recurrent Neural Network (RNN)**: $s^a_t = \sigma(s^a_{t-1}W_s + \omega_t W_o)$ where W_s and W_o represents weight vectors and σ some non linear function. [11, 12, 23].

This RNN approach to solving POMDPs is related to other problems using dynamical systems and state space models, where the true state can only be estimated [5].

1.2 RL Agent Components:

So far, we have introduced the key formalism used in RL, the MDP and the POMDP. Now I will talk about what is inside an RL agent and the components it uses for learning. An RL agent may include one or more of these component for learning.

Value functions: Value function attempts to measure goodness/badness of each state and/or action. In other words, value function methods are based on estimating the value (expected return) of being in a given state i.e. future reward. The state-value function $V_\pi(s)$ is the expected return when starting in state s and following π henceforth:

$$V_\pi(s) = \mathbb{E}_\pi[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots | s_t = s]$$

where $\gamma \in [0, 1]$ is discount parameter, that governs the future sightedness of the model.

Policy Search: A policy is the *agent*'s behavior. It is a map from state to action. Examples of some policies are:
(i) *Deterministic Policy:* This is the simplest policy, here an action a is mapped to a state s i.e. $a = \pi(s)$.
(ii) *Stochastic Policy:* A policy can be stochastic based on probability of state observation, i.e. $\pi(a|s) = \mathbb{P}[\mathcal{A} = a|\mathcal{S} = s]$.
(iii) *Parameterized Policy:* A parameterised policy π_θ is chosen, whose parameters are updated to maximise the expected return $E[R|\theta]$ using either gradient-based or gradient-free optimisation [7].

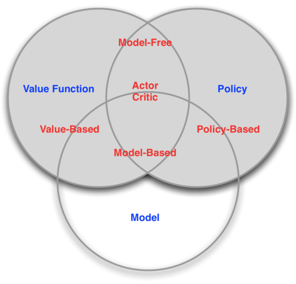


Figure 3: RL Agent Taxonomy.

Neural networks that encode policies have been successfully trained using both gradient-free [6, 9, 15] and gradient based [13, 16, 24] methods.

Model: A model tries to learn the behavior of the environment. First, it tries to predict what the environment will do next, that is predicting next state and is called Transitions ($\mathcal{P}$). Secondly, the model tries to predict the expected next reward and is called Rewards model ($\mathcal{R}$).

$$\mathcal{P}^a_{ss'} = \mathbb{P}[\mathcal{S}' = s'|\mathcal{S} = s, \mathcal{A} = a]$$

where s is the state prior state and s' is the resultant state on action a.

$$\mathcal{R}^a_s = \mathbb{E}[r|\mathcal{S} = s, \mathcal{A} = a]$$

where s is the state prior state and r is the reward for next step on action a.

1.3 RL Agent Taxonomy:

Model methods are optional in RL systems. In fact there are many model free based methods applied for real world problems. To understand a comprehensive taxonomies of categorizing RL Agent we present it in Figure 3.

RL focus on learning without access to the underlying model of the environment. However, interactions with the environment could be used to learn value functions, policies, and also a model. Model-free RL methods learn directly from interactions with the environment, but model-based RL methods can simulate transitions using the learned model, resulting in increased sample efficiency [1]. This is particularly important in domains where each interaction with the environment is expensive. However, learning a model introduces extra complexities, and there is always the danger of suffering from model errors, which in turn affects the learned policy; a common but partial solution in this latter scenario is to use model predictive control, where planning is repeated after small sequences of actions in the real environment [5]. Although deep neural networks can potentially produce very complex and rich models [8, 18], sometimes simpler, more data efficient methods are preferable [10].

1.4 Deep Reinforcement Learning (DRL):

Traditional RL works was mainly on low dimentional problems (i.e. few state space, limited actions etc.). Integration of deep neural network with RL framework bolster the framework, by converting higher dimentional problems into low dimentional representations. Initial DRL works was mainly involved on scaling up prior work

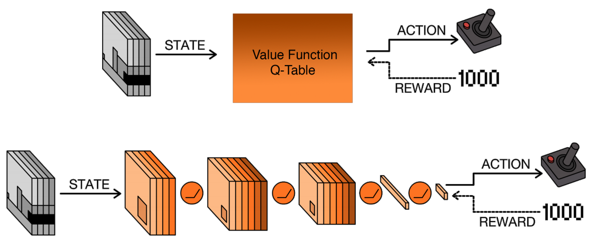


Figure 4: Q-learning RL (above) and DQN (below) [17].

in RL to high dimensional problems. DRL can deal efficiently with the curse of dimensionality, unlike tabular and traditional non-parametric methods [4].

In DRL, deep neural network is trained to model/predict one or more of (a) the optimal value functions $V^{\pi}(s)$ (b) optimal policy $\pi(s)$ (c) optimal quality function Q^* (d) optimal actions $\mathcal{A}^*$.

Many works on DRL is based on gradient based backpropagation algorithm [13, 19, 20] which models the optimisation of the expected return as the optimisation of a stochastic function.

This stochastic function can have one or more models, policies and value functions combined in various ways. Each individual component might not directly optimise the expected reward but can incorporate useful information to optimize reward as a whole. For example, a DRL using a differentiable model and policy, it is possible to forward propagate and backpropagate through entire episodes; and policy component can learn the information over the history. Both can be summarized with a value functions for optimizing reward [13].

To present a concrete example of how a traditional RL network is extended to a DRL, I will present a value-function-based DRL algorithms with the DQN in Figure 4. Q-functions learns action-value function. In a traditional RL problem, a Q-learning function create and update a Q-table to find the maximum expected future reward of an action, given a current state. The model takes greyscale images as state from the video game; with the input current state Q-table returns with actions. It is a good strategy but it is not scalable.

On the other hand, the DQN takes the state-a stack of greyscale frames from the video game-and processes it with convolutional and fully connected layers, with ReLU nonlinearities in between each layer. At the final layer, the network outputs a discrete action, which corresponds to one of the possible control inputs for the game. Given the current state and chosen action, the game returns a new score. The DQN uses the reward-the difference between the new score and the previous one-to learn from its decision. More precisely, the reward is used to update its estimate of Q, and the error between its previous estimate and its new estimate is backpropagated through the network.

1.5 Current Research Challenges:

In this section, I present some of the research challenges in Deep Reinforcement Learning process.

Exploration vs Exploitation: Online decision making involves a fundamental choice *(i) Exploitation* or *(ii) Exploration.* Exploitation refers to make the best decision with the current information gathered by agent. Whereas exploration involves attempting new action for more information. This is also known as *exploitation-exploration* dilemma. The best long-term strategy may involve lot of exploration and short term sacrifices. On the other hand if the agent have already found best strategy exploration may reduced the rewards.

Transfer Learning: Transfer learning is about efficiently using previous knowledge from a source environment to achieve new (slightly) different tasks in a target environment. To achieve this agent must develop generalization capabilities such as *(i) feature selection (ii) removing asymptotic bias (iii) reduce overfitting and function approximator (iv) Optimizing horizon* (length of observations history involved in decision making process) etc.

Learning without explicit reward function: In reinforcement learning, the reward function defines the goals to be achieved by the agent. Due to the complexity of the environments in practical applications, defining a reward function can turn out to be rather complicated. Approaches like *imitation learning* (supervised learning) and *inverse reinforcement learning* where agent determines possible reward functions given observations of optimal behavior, are research challenges for next decade.

2 PART B. SELF INSTRUCTING DATABASE:

2.1 Overview:

Any standard database has considerable large number of configuration knobs. Databases needs to be tweaked with proper configuration for running efficiently with different workloads and hardware resources. Tweaking database configuration knowbs needs a great level of expertise from database administrator (DBA) because optimal configuration varies with the type of worloads. Beside that even finding optimal knob configurations might involve trial and error process for a DBA (which restrits the search spaces for knobs). An auto-configuring database system or self intructing database is a desirable feature to demand from database companies.

Human database administrators rely on experience and intuition to configure it. DRL process mimick same learning strategies i.e. learn from mistakes and correctness to maximize future rewards. Keeping this in mind we can explore how DRL can provide a solution to automatic database tuning.

2.2 Problem Formulation:

Given a workload $\mathcal{W}$ (which is a serialized query profile DAG of execution task), with a knobs setting $C = \{c_1, c_2, \ldots, c_{|C|}\}$ a typical database outputs a log of metrics $\mathcal{M} = \{m_1, m_2, m_3, \ldots, m_{|\mathcal{M}|}\}$ as shown in Figure 5. The database keeps receiving workloads at some discrete interval of time and it runs with some configuration and outputs a metric log, which is also called a discrete time stochastic control process.

We can apply deep reinforcement learning to train a neural network in taking over the database tuning process by optimizing configuration for observable database workload. Essentially, we have to define a RL problem environment consisting of the four following components to perform the learning as shown in Figure 6:

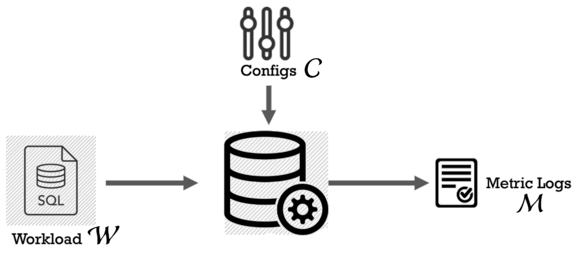


Figure 5: A typical database system.

1) Observable State: This is also input to the neural network. This is typically the current workload in form of query characteristics, for which the system should be optimised as well as the current state of the configuration. Figure 6 illustrates workload $\mathcal{W}_t$ is mapped to observation/state $\omega_t \in \Omega$ or $s_t \in \mathcal{S}$ for time stamp t. (Note: $\omega_t = s_t$ in MPD)
(Note: These notations are defined in Section 1.1).

2) Actions: An action is a bounded set of configuration C_t where each knobs can have a range of permissible values. Changing the size of a database buffer is an example of action. Now we can represent $C_t \to a_t \in \mathcal{A}$.

3) Reward: We map the metric $\mathcal{M}_t$ to rewards $r_t \in \mathcal{R}$. Since reward is a scalar function and metric $\mathcal{M}_t$ is a set of values representing the goodness and badness of database execution on workload $\mathcal{W}_t$, we need to apply some function $r_t = f(\mathcal{M}_t)$ to keep it simple. Later we will see an alternative approach where function f is not needed with multi-agent DRL.

4) Hyperparamters for Neural Network: This includes properties of the neural network (e.g. number of hidden layers, number of nodes per layer) as well as properties of the learning process like the number of iterations.

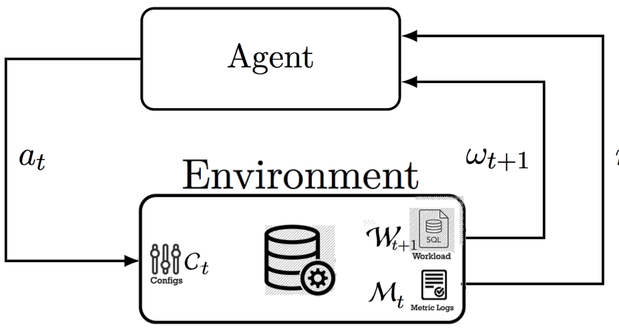


Figure 6: RL Agent-Database Environment Interaction.

Training and Prediction:

With the high level components defined, now we will go through the workflow of learner. Assuming the *neural network agent (NNA)* is configured with required hyperparamters, the learning process starts with time $t = 0$. First, a workload $\mathcal{W}_t = s_t$ is feed into *neural network agent* (NNA). Then, the NNA explore action set $\mathcal{A}$ to produce an action (configuration) $a_t = C_t$. The database environment on receiving action configuration C_t executes workload $\mathcal{W}_t$ and returns with metrics $\mathcal{M}_t$. Some function $f : \mathcal{M}_t \to r_t \in \mathbb{R}$ converts metric to scalar reward. This process is repeated many times and agents either explore action set or exploit learned optimal action to predict next best configuration. The desired goal is to optimize maximum cumulative positive rewards.

An intuitive solution for choosing function $f : \mathcal{M}_t \to r_t \in \mathbb{R}$ is through these steps:

- negate all the metrics whose desired objective is to minimize, such that we now optimize for maximization.
- normalize each metric $m_i \in \mathcal{M}_t$ with their satisfied range of operation.
- rewards is sum of normalized metrics.

To avoid the problem of choosing a function f, we can transform the problem to a *Multi-task Deep RL* problem. Some hierarchical RL techniques also decompose tasks into subtasks, these methods then solve the subtasks in a locally optimal way and then global optimality can be achieved by aggregating back together. Another approach is to try *Linear Temporal Logic* specification that enables an interleaving of subtasks to support global optimization [14]. These techniques can help in avoiding selection of f by considering specific set of metrics and optimize actions for it.

Model based/Model-free Agent: Model-free and model based both type of agent can be frutiful in this type of scenario. However model based approach needs more effort in design. The advantage of model based approach is that it can learn strategies to trade-off exploration and eploitation to learn quickly. But it is non-arguably plausible to choose gradient based methods because configurations knobs tend to have convex properties.

REFERENCES

[1] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath. A brief survey of deep reinforcement learning. *arXiv preprint arXiv:1708.05866*, 2017.
[2] A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983.
[3] R. Bellman. Dynamic programming. *Princeton, USA: Princeton University Press*, 1(2):3, 1957.
[4] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.
[5] D. P. Bertsekas. Dynamic programming and suboptimal control: A survey from adp to mpc. *European Journal of Control*, 11(4-5):310–334, 2005.
[6] G. Cuccu, M. Luciw, J. Schmidhuber, and F. Gomez. Intrinsically motivated neuroevolution for vision-based reinforcement learning. In *Development and Learning (ICDL), 2011 IEEE International Conference on*, volume 2, pages 1–7. IEEE, 2011.
[7] M. P. Deisenroth, G. Neumann, J. Peters, et al. A survey on policy search for robotics. *Foundations and Trends® in Robotics*, 2(1–2):1–142, 2013.
[8] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 512–519. IEEE, 2016.
[9] F. Gomez and J. Schmidhuber. Evolving modular fast-weight networks for control. In *International Conference on Artificial Neural Networks*, pages 383–389. Springer, 2005.
[10] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine. Continuous deep q-learning with model-based acceleration. In *International Conference on Machine Learning*, pages 2829–2838, 2016.
[11] M. Hausknecht and P. Stone. Deep recurrent q-learning for partially observable mdps. *CoRR, abs/1507.06527*, 7(1), 2015.
[12] N. Heess, J. J. Hunt, T. P. Lillicrap, and D. Silver. Memory-based control with recurrent neural networks. *arXiv preprint arXiv:1512.04455*, 2015.
[13] N. Heess, G. Wayne, D. Silver, T. Lillicrap, T. Erez, and Y. Tassa. Learning continuous control policies by stochastic value gradients. In *Advances in Neural Information Processing Systems*, pages 2944–2952, 2015.
[14] R. T. Icarte, T. Q. Klassen, R. Valenzano, and S. A. McIlraith. Teaching multiple tasks to an rl agent using ltl. 2018.
[15] J. Koutník, G. Cuccu, J. Schmidhuber, and F. Gomez. Evolving large-scale neural networks for vision-based reinforcement learning. In *Proceedings of the 15th*

annual conference on Genetic and evolutionary computation, pages 1061–1068. ACM, 2013.

[16] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[17] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[18] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh. Action-conditional video prediction using deep networks in atari games. In *Advances in neural information processing systems*, pages 2863–2871, 2015.

[19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[20] J. Schulman, N. Heess, T. Weber, and P. Abbeel. Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing Systems*, pages 3528–3536, 2015.

[21] R. S. Sutton. Temporal credit assignment in reinforcement learning. 1984.

[22] R. S. Sutton and A. G. Barto. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

[23] D. Wierstra, A. Förster, J. Peters, and J. Schmidhuber. Recurrent policy gradients. *Logic Journal of the IGPL*, 18(5):620–634, 2010.

[24] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Problem 3

1) Let's say you now work at twitter and are part of a team that builds dashboards of geotagged tweets. You're able to monitor these for indicators of happiness, food, and physical activity. You're also able to build associations with various desirable and undesirable health conditions. BlueCross Blueshield comes to you with a project: they want to use your predictive tools along with their own databases of individuals to augment risk factor analysis with data from individual's social media streams.

a) How would you go about building such a tool? What are the main technical and data challenges that you foresee? b) A member of your team feels uncomfortable about doing this for the insurance company on ethical grounds, and seeks to void the contract. What is your position on this and why?

2) Consider the same scenario above, but now instead of BlueCross BlueShield, it's the Center for Medicare and Medicaid Services (a federal government entity). Does your answer in b) above change?