Deep Learning for Adverse Event Detection From Web Search

Faizan Ahmad, *Member, IEEE*, Ahmed Abbasi, *Senior Member, IEEE*, Brent Kitchens, Donald Adjeroh, *Member, IEEE*, and Daniel Zeng, *Fellow, IEEE*

Abstract—Adverse event detection is critical for many real-world applications including timely identification of product defects, disasters, and major socio-political incidents. In the health context, adverse drug events account for countless hospitalizations and deaths annually. Since users often begin their information seeking and reporting with online searches, examination of search query logs has emerged as an important detection channel. However, search context - including query intent and heterogeneity in user behaviors – is extremely important for extracting information from search queries, and yet the challenge of measuring and analyzing these aspects has precluded their use in prior studies. We propose DeepSAVE, a novel deep learning framework for detecting adverse events based on user search query logs. DeepSAVE uses an enriched variational autoencoder encompassing a novel query embedding and user modeling module that work in concert to address the context challenge associated with search-based detection of adverse events. Evaluation results on three large real-world event datasets show that DeepSAVE outperforms existing detection methods as well as comparison deep learning auto encoders. Ablation analysis reveals that each component of DeepSAVE significantly contributes to its overall performance. Collectively, the results demonstrate the viability of the proposed architecture for detecting adverse events from search query logs.

Index Terms—Adverse event detection, search queries, deep learning, auto encoders, query embeddings, user modeling

1 Introduction

Adverse event detection has become a critical component of post-marketing surveillance in many contexts including pharmaceutical drugs, children's toys, and the automotive industry [2]. For instance, adverse reactions to pharmaceutical drugs are responsible for over 10 percent of all hospital admissions [41], resulting in millions of hospitalizations and over 100,000 deaths annually [44]. The pharmaceutical drug Pradaxa alone has caused 9,000 hospitalizations, 1,000 deaths, and $650 million in lawsuit settlements over the past five years [51]. Similarly, in the automotive industry, Toyota recently settled lawsuits totaling nearly $6 billion for inadequate rust protection on their trucks, and the unintended acceleration "sticky pedal" fiasco [2]. Such surveillance also has implications for other types of events, including socio-political incidents and natural disasters [27], [45]

Detection entails use of signal or anomaly detection methods capable of accurately identifying such events in a timely manner (i.e., earlier). In recent years, there has been greater focus on employing user-generated content channels to detect adverse events [2], with user search query logs serving as a major channel [3], [56]. The importance and viability of search is largely due to the increased volume and timeliness of search data – users often begin information seeking and reporting with online searches [2], [27]. Consequently, the ability to detect events using search query log-based signals in an accurate and timely manner has important implications for many real-world problems. Given its immense potential for garnering situational awareness and listening to the voice of the customer, in 2009, Google chief economist Hal Varian noted that search trends could help "predict the present" [12]. However, search-based event detection has been somewhat maligned in recent years. Recent studies have shed light on a major challenge – the context problem [9], [29]. A high-profile example where lack of proper contextualization might have been partly responsible was Google shutting down their search-based flu trend prediction service after it over-estimated flu levels by nearly 100 percent one year [10], [11]. The lower salience of search, due to reliance on queries that are typically 3-5 words in length or shorter, makes it difficult to properly infer query intent [1], [27] – people seeking information on flu treatment versus those wondering if they should get a flu shot this year [10]. Further, users' internet behaviors are diverse – yet existing detection methods rarely consider user heterogeneity [7].

We propose a novel deep learning framework called DeepSAVE (deep learning for search-based adverse event detection) for detecting adverse events based on user search query logs. DeepSAVE employs an enriched variational autoencoder that incorporates specific provisions to address the context challenge related to detection of adverse events via search, including a query embedding for better representation

- *Faizan Ahmad is with Computer Science, University of Virginia, Charlottesville, VA 22904. E-mail: faizann288@gmail.com.*
- *Ahmed Abbasi is with IT, Analytics, and Operations, University of Notre Dame, Notre Dame, IN 46556. E-mail: aabbasi@nd.edu.*
- *Brent Kitchens is with Information Technology, University of Virginia, Charlottesville, VA 22904. E-mail: bmk2a@comm.virginia.edu.*
- *Donald Adjeroh is with the Lane Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown, WV 26506. E-mail: Donald.Adjeroh@mail.wvu.edu.*
- *Daniel Zeng is with the Institute for Automation, Chinese Academy of Sciences, China. E-mail: dajun.zeng@ia.ac.cn.*

Manuscript received 23 July 2019; revised 12 July 2020; accepted 10 Aug. 2020. Date of publication 0 . 0000; date of current version 0 . 0000.
(Corresponding author: Ahmed Abbasi.)
Recommended for acceptance by E. Chen.
Digital Object Identifier no. 10.1109/TKDE.2020.3017786

TABLE 1
2x2 Contingency Table

	Outcome of Interest	Other Outcomes
Entity of Interest	a	b
Other Entities	c	d

a, b, c, and d represent the frequency of occurrence.

of search intent, and user-level modeling to account for heterogeneity.

DeepSAVE was evaluated on a rich test bed encompassing 104 million user search queries spanning a six year period, coupled with three event databases containing over 800 events related to the health and automotive industries. The results reveal that the proposed framework is able to garner enhanced recall and f-measures relative to existing baseline and benchmark methods. Ablation analysis shows that each component of DeepSAVE significantly contributes to its performance, underscoring the efficacy of the proposed framework.

2 Background and Related Work

2.1 Disproportionality Analysis

Disproportionality Analysis (DA) methods [36] find potential associations between entities and adverse outcomes. Examples of entities include the drug Pradaxa or Toyota Prius, whereas associated outcomes might be stomach bleeding (in the case of Pradaxa) or the accelerator pedal sticking (in the case of Prius). DA methods are computed based on a 2x2 contingency table encompassing entity and outcome occurrences (see Table 1). DA has been widely used in the past for adverse event detection from search and spontaneous reporting databases [2]. Specific examples of DA measures proposed in the literature are Reporting Odds Ratio (ROR), Relative Reporting Ratio (RRR), Proportional Reporting Ratio (PRR) and Information Component (IC) [8], [36]. As noted, these measures are based on values in Table 1. For instance, ROR is computed as (a*d)/(b*c). Szarfman *et al.* [46] proposed a multi gamma poisson shrinker (MGPS) method that adopts a relatively more involved Bayesian approach. For most DA methods, values above a certain threshold are deemed potential adverse events (i.e., "positives") [13]. Many DA methods suffer from high variability due to simplified "mention" model-based detection that ignores search context [2]. Consequently, DA methods have typically yielded low precision and recall for adverse event detection [2], [3].

2.2 Association Rule Mining

Association rule mining (ARM) methods follow a similar intuition to DA methods by attempting to find associations between entities and related potential adverse outcomes. Several measures, such as support and confidence [48] have been proposed to mine association strength between two objects [4], [53]. Given such measures for all entity-outcome tuples, only tuples with measures above a certain threshold are deemed potential events ("positives"). Most of these methods are well-suited for pervasive adverse events (i.e., ones with high support), but do not work well for events with a weaker signal [48]. To address this problem, some studies have focused on more robust adverse event detection methods [32], [48], [53], but these still suffer from high false positive rates and high computational complexity [22]. Ji*et al.* [22] proposed two association measures based on a fuzzy recognition-primed decision model [21] for mining causal relations between drugs and adverse reactions, called causal leverage (CL) and exclusive causal leverage (ECL). Further, Jin *et al.* [23] proposed an interestingness measure and mining algorithm (EXCLEV) for highlighting unexpected events. These measures have demonstrated strong results on electronic databases, but have not been applied in the context of search data.

2.3 Event Mention Classification

Event mention classification methods use a classifier to categorize potential adverse event mentions such as a tweet or search query [2], [34]. The filtered mentions (i.e., those categorized as relevant) are then input into DA or ARM methods. For example, the classifier results may create a refined subset of a,b,c,d in Table 1 which can then be used for DA-based adverse event detection, thereby potentially alleviating false positives and enhancing precision. Numerous approaches for classifying event mentions in text have been proposed. Sarker *et al.* [44] trained a Support Vector Machine (SVM) classifier to detect whether a tweet contains an ADR. They applied their method on a dataset encompassing 6K manually annotated tweets. Lee *et al.* [31] proposed a method for ADR detection in tweets using a Convolutional Neural Network (CNN) and region embeddings. Huynh *et al.* [19] applied CNNs followed by Recurrent Neural Networks (RNN) with and without attention mechanism to two labeled datasets. Event mention classification methods attempt to better contextualize and refine entity-outcome mention tuples, thereby implicitly examining search/query intent [27]. However, these approaches do not consider user-level characteristics [7] such as heterogeneous search and nuanced querying patterns. Moreover, they still rely on DA methods for the final event detection signals. As we later demonstrate empirically in our evaluation section, these limitations make event mention classification techniques less ideal for adverse event detection.

2.4 Data Mining Techniques for Twitter Event Detection

Twitter is a major channel for social-media based detection of real-world events. Hashtags have made it easier to find and extract tweets related to a specific event, upon which data mining techniques can be applied. Several such methods have been proposed in recent years that consider event detection as a temporal stochastic process. Here we discuss a few exemplars selected based on their performance as reported in prior studies [3], [17]. pyMABED [39] uses anomaly detection to detect spikes in event mentions which can be visualized in a system for manual inspection. SEDTWik [37] examines tweet hashtags to find bursty segments which are clustered to find important events. Precision is increased by making use of an external data source (Wikipedia) to verify events. TwitterTopics [20] aggressively filters tweets based on length and content. It then applies hierarchical clustering on the refined set of tweets and finally prunes results by weighting. PeakLabel [3] uses a spike detection heuristic to identify events from Twitter mentions. Despite empirically performing well on social media-based event detection tasks, it remains unclear how well these techniques can perform on search data. For instance, while these aforementioned methods perform analysis (e.g., anomaly detection or cluster analysis) at the word or tag level, important factors such as word sense and context are omitted from the analysis pipeline. Hence, some of the same intent and user heterogeneity limitations mentioned earlier may apply. Moreover, many of these methods are based on sophisticated pipelines that rely heavily on the manual feature/model engineering paradigm.

2.5 Auto Encoders and Dimensionality Reduction

Dimensionality reduction methods are a family of unsupervised methods that deal with learning an efficient compressed representation of the data that is well-suited for easy reconstruction of the input. Principal component analysis (PCA) [57] and Singular Value Decomposition (SVD) [16] are two seminal methods that have been successfully used for data compression, anomaly, and event detection [40]. In recent years, Auto encoders (AE) [18], which are a type of unsupervised neural networks, have been proposed for this task. They consist of two components, an encoder that converts the input into a compressed representation, and a decoder that converts the compressed representation back to the original input. Reconstruction loss is used to back propagate the error and enable learning. Initially, auto encoders were used for dimensionality reduction [18], but recently, they have been applied to anomaly detection tasks [43], [58]. After training an auto encoder, if a test instance gives high reconstruction loss (i.e., the network cannot reconstruct the test input accurately relative to some threshold), it is considered an anomaly. Denoising auto encoders (DAE) [50] are a special type of auto encoder where a small amount of noise is intentionally added to the input as a regularization strategy. By learning on noisy input, the goal is to train models robust to small perturbations. Variational auto encoders (VAE) [28] constrain the compressed representation to follow a prior distribution (e.g., Gaussian). The encoder compresses the input into two compressed representations: mean and variance which are joined to get a single representation. Kullback-Leibler divergence is used to constrain the compressed representation to match the prior distribution. Adding this distribution constraint can act as a regularization strategy. It also allows use of more principled anomaly thresholds based on prior distributions. Despite encompassing several attractive properties, to the best of our knowledge, auto encoders have not previously been used for adverse event detection.

2.6 Deep Learning for Search

There has been increased research interest in applying neural networks to word (word2vec) [35] and sentence embeddings [30]. These embeddings represent words and sentences in a high dimension such that there are semantic relationships between them. A few extensions of word2vec [26] have also been proposed for modeling search queries. Zamani *et al.* [60] and Le *et al*. [30] proposed a method of averaging word embeddings to create embeddings for short pieces of text such as queries and sentences. Query2vec [26] uses ideas from word2vec and skipgram modeling to propose several different schemes for creating query embeddings, including querygram, clickgram, and sessiongram. Query embeddings could help better account for search context factors such as query intent, yet have not been employed in prior adverse event detection studies.

2.7 Research Gaps

Based on our review of prior work, we have identified two important research gaps:

- *Lack of Attention to Search Context*–Salience is critical, yet often elusive with user-generated content channels such as search query logs [1]. Failure to contextualize user-generated content can have dire consequences for event detection [56]. Nevertheless, effective contextualization methods for adverse event detection remain elusive. The intention behind queries is one critical consideration [60]. Further, users may internalize and vocalize adverse experiences in diverse ways, depending on various factors. Prior work examining user-generated channels has mostly not considered such heterogeneity.
- *Dearth of Parsimonious Models for Detecting Adverse Events*–Previous studies have largely relied on aggregate-level DA or ARM methods applied atop either basic or machine-learning classifier-based mention models [10]. As we later demonstrate empirically, such methods, which are applied uniformly across entities in a sequential “pipeline” manner, are unable to learn the nuanced characteristics of specific potential adverse events since they fail to consider the interplay between mention instances and aggregate-level events.

3 Proposed Framework: DeepSAVE

3.1 Generic Definition of Problem and Solution Space

Suppose we have externally defined sets $e_1, e_2, \ldots, e_m \in E$ entities and $o_1, o_2, \ldots, o_n \in O$ outcomes for a given problem domain. For instance, if attempting to detect adverse drug events, entities would be all relevant drugs and outcomes the set of all possible adverse reactions. For automotive events, entities would be vehicle makes and models, whereas outcomes would include vehicular defects that could manifest after purchase. This results in a set of possible entity-outcome tuples $e_i o_j \in E \times O$. The objective is to examine potential event signals to identify, as timely as possible, each tuple $e_i o_j \in A \subset E \times O$, where A is the set of all true adverse events, which is defined by external criteria and ex ante unobservable aside from a known subset $A' \subset A$ that may be used for training.

In each time period t, users $u_1, u_2, \ldots u_k$ each perform queries $q_{ut1}, q_{ut2}, \ldots q_{utl}$, which may contain a potential event signal $e_i o_j$ by either individually or collectively mentioning both entity e_i and outcome o_j. Following the standard approach adopted in the disproportionality analysis literature [55], [56], evidence for an actual event may be provided by entity-outcome tuples that are anomalous (i.e., disproportionate) in their occurrence relative to other tuples, both within

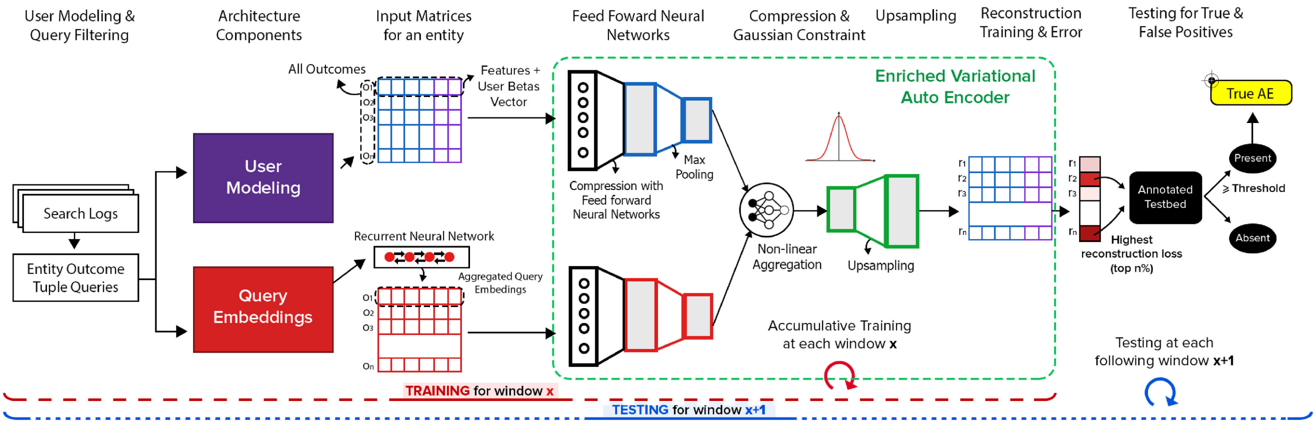


Fig. 1. Proposed deep learning framework for adverse event detection.

and across timeframes, or relative to themselves from prior time periods [2]. At time t, some metric of strength for each potential signal $e_{it}o_{jt}$ is compared against its past occurrences $e_{is}o_{js}$ $\forall s < t$ as well as with other event signals at the current time $e_{xt}o_{yt}$ $\forall e_{x\in E}o_{y\in O}$. Based on this comparison, an anomaly score $a_{te_io_j}$ is assigned to each signal e_io_j $\forall e_{i\in E}o_{j\in O}$ that quantifies the abnormality of the signal. All signals are ranked based on anomaly scores, and the top $p\%$ adverse event signals (highest anomaly scores) are considered as our potential true positives. These are measured against a ground truth set of known true events from future times $t_{k+1}, t_{k+2}, \ldots, t_n$ to gauge performance.

3.2 DeepSAVE Components

Fig. 1 depicts our DeepSAVE deep learning framework for adverse event detection. It consists of multiple components designed to address the aforementioned gaps in the literature:

- *Query Embedding* for the query intent aspect of search context. This module combines search queries with clickstream data to create query embeddings for better intent inference. In particular, the query embedding attempts to disentangle relevant from irrelevant queries, and further categorize relevant queries into different types.
- *User Modeling* to account for the user heterogeneity aspect of search context. This component of the framework uses hierarchical Bayesian modeling to generate a novel user embedding to identify and account for diversity in how users seek information via search. Collectively, the user modeling accounts for diverse user content generation and consumption patterns.
- *Enriched Variational Auto Encoder* for parsimonious modeling of adverse events. The auto encoder takes compressed representations of the aforementioned components and attempts to reconstruct them with a prior distribution constraint. To enhance performance, the decoder is enriched with query and user embeddings to better align reconstruction loss with valid adverse event signals.

3.3 Overview of DeepSAVE

We begin with a high-level overview of DeepSAVE before diving further into the three components: enriched variational autoencoder, query embeddings, and user modeling. As noted, adverse event detection is about detecting an event signal. As shown in Fig. 1, DeepSAVE embodies this core intuition. Longitudinal search data is divided into n sliding windows of size s. For each window w_i, queries with entity-outcome tuples are extracted. Since a user's search intent might manifest across multiple queries, we let the entity and outcome terms occur in different queries within a time period T. Each query is associated with its text, a user id, and a set of URLs visited within t seconds after entering the query. Relevant queries are passed through two different components of DeepSAVE that extract information for tuples and create input matrices for an entity with outcomes as row vectors.

The query embedding and user modeling modules collectively extract two key matrices that are used as input for the variational autoencoder. The query embedding module focuses on derivation of meaningful representations of queries for user intent inference. Since a single outcome can be associated with multiple queries, we use a recurrent neural network to aggregate the query embeddings for each outcome before inserting them into vectors in the query embeddings matrix M_q.

The user modeling component extracts individual, as well as aggregated measures for users belonging to each tuple. These values are intended to capture users' information foraging behavior. Data generated by the user modeling module is used to create a user feature + user embedding matrix M_u. Ultimately, we are interested in finding entity-outcome tuples (i.e., rows) of the feature + user embedding matrix M_u. The query matrix is used for enrichment (i.e., regularization strategy) analogous to a denoising autoencoder [50].

The two matrices generated by the user modeling and query embedding components are each passed through separate Feedforward Neural Networks to extract representations that are concatenated and non-linearly aggregated together by another Feedforward Neural Network to form a single compressed representation which is constrained to follow an isotropic Gaussian distribution. By upsampling, this representation is converted back into the original M_u. As noted, the matrix for query embeddings is not reconstructed since it is only used to help the auto encoder reconstruct the core matrix M_u, which is used to identify adverse events.

DeepSAVE is trained and tested using a sliding time series window approach. Given test window w_{i+1}, training employs a cumulative growing window spanning w_0 to w_i.

Consistent with prior autoencoders, reconstruction loss is used to train the VAE [28], resulting in a fully unsupervised method for adverse event detection (in the sense that no apriori event labels are used). For each test window w_{i+1}, reconstruction loss is calculated for each outcome row in the entity-outcome text matrix. Entity-outcome tuples with reconstruction loss above a certain threshold are considered potential adverse events. Details about the enriched VAE, query embedding, and user modeling appear in the remainder of the section.

3.4 Enriched Variational Auto Encoder

Variational auto encoders (VAEs) have shown great promise for efficiently compressing input data into specific distributions. These distributions in turn exhibit statistically sound properties for threshold-based anomaly detection – for instance, only 5 percent of data lies outside two standard deviations in a Gaussian distribution. In order to leverage these properties, DeepSAVE uses a VAE as its core anomaly detection engine. The algorithmic intuition guiding our enriched VAE is that since adverse-event related searches are less common, when properly accounting for query intent and diversity in user search behavior, they will exhibit anomalous patterns relative to regular searches in a time series modeling context. More specifically, as we train the enriched VAE for each time window, it learns the distribution of entity-outcome row and entity-level matrix association patterns – whenever an adverse event occurs, if its entity matrix falls outside the learned data distribution due to spikes in certain entity-outcome searches, those entity-outcome time periods will be flagged as anomalous signals [58]. The enriched VAE leverages query embedding and user modeling-based enrichment to the input as a regularization strategy to account for query intent and user heterogeneity, thereby enabling more accurate reconstruction loss measurement.

As alluded to in our framework overview, DeepSAVE trains on a set of entity matrices, each consisting of all possible outcomes as rows. Therefore, each row can also be called an entity-outcome tuple. The query embedding and user modeling components (described later in Sections 3.3 and 3.4) are used to generate two input matrices: M_q, M_u, for the enriched VAE. From relevant queries, data for these matrices is generated and passed through separate feedforward neural networks and global max pooling layers to extract compressed global representations for each matrix. For instance, the compressed query embeddings representation is a global representation that encompasses query intent for the input entity as a whole, as well as the individual outcomes. These representations are non-linearly aggregated together before being applied with a Gaussian constraint that acts as a regularization strategy. Aggregation is done in order to obtain a single global representation from all components, which is used to reconstruct the feature + user embedding matrix.

Formally, let $u_1, u_2, \ldots, u_n \in M_u$ denote the outcome rows of an entity matrix for user features and embeddings and $b_1, b_2, \ldots, b_n \in M_q$ for query embeddings. Row vectors for each matrix are of different sizes s_u, s_b respectively. We pass each matrix through a separate feedforward neural network that compresses the width of the matrix while keeping the number of rows the same. Since the embedding and feature information is present in the columns of the matrices, we start by compressing them using a feedforward neural network with weights W_m

$$C_M = \sigma(b_m + M * W_m), \tag{1}$$

where the number of units m in the neural network layer is much smaller than the number of columns in the matrix M. σ is the activation function and $*$ is the matrix product.

For each entity matrix M_q, M_u, we consider the output C_M as the compressed representation. Let C_u, C_q denote the compressed representations of feature and query matrices respectively. Then, the aggregated compressed representation is given by

$$C = \sigma(b_c + (C_{concat}) * W_c. \tag{2}$$

C_{concat} is the concatenation of the compressed representations which is given by

$$C_{concat} = C_q \oplus C_u. \tag{3}$$

Given C, we convert it into two more compressed representations for mean and variance of the constraining distribution (a Gaussian distribution in our case)

$$Z_\mu = f(W_{zu} * C_{concat} + b_{concat}) \tag{4}$$

$$Z_\sigma = f(W_{z\sigma} * C_{concat} + b_{concat}) \tag{5}$$

$$Z = z_\mu + z_\sigma * \epsilon. \tag{6}$$

In Equation (6), z is sampled using a "reparametrization trick" [28] that enables backpropagation in the network. The $\epsilon \sim$ Normal(0, 1) parameter adds a random node in the network thus allowing for the gradients to flow back. Finally, Z is upsampled using another feedforward neural network to reconstruct the feature matrix. As noted, we do not reconstruct the query matrix since it is only used for enrichment of the autoencoder. Let p be the part of auto encoder that is responsible for compressed (encoder) and q be the part that is responsible for reconstruction. The encoder-decoder network is trained end-to-end with loss function given in Equation (7).

$$\begin{aligned} L(\theta, \phi, M_u, M_q) = E(M_u, M_u Z) \\ -KL(q_\phi(Z|M_u, M_q)||p_\theta(Z)) \end{aligned} \tag{7}$$

$$E(M_u, M_u|Z) = |M_u - (M_u||Z)|_q \tag{8}$$

$$KL(q||p) = p(M_u) * (log(p(M_u) - log(q(M_u))). \tag{9}$$

Equation (7) contains two terms, the first term is the reconstruction loss given by E in Equation (8), while the second term is the Kullback Leibler divergence (9) which quantifies the misfit between the posterior distribution of Z and a unit Gaussian. θ are the parameters of the encoder network p while ϕ represents the parameters of decoder network q.

As noted, training reconstruction loss is only calculated across the entire feature matrix M_u. However, during testing, we calculate the reconstruction loss for each outcome row (entity-outcome tuple) in the feature + user embedding matrix (Eq. (10)). This represents the anomaly score for the signal; if it is higher than a threshold $thrsh$, we consider it an adverse event signal.

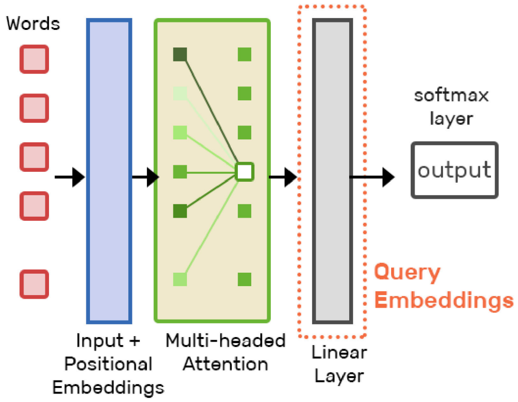


Fig. 2. Transformer classifier used to derive query embeddings.

$$\begin{aligned} &\forall\ e \in Entities\ \forall\ V_o \in M_{ue} \\ &E(V_o, V_{o^*}) = |V_o - V_{o*}|_1\ >\ thrsh \\ &thrsh \rightarrow \text{AE Signal for } V_o \text{ and e.} \end{aligned} \tag{10}$$

3.5 Query Embeddings

Query intent is an important contextual consideration for search-based detection models [10], [29]. For instance, the query "does adderall give headache relief" contains an entity (adderall) and outcome (headache), but is obviously not referring to a potential adverse event since the intent is to ask a clarification question as a prospective user of Adderall, and the context of the search is focusing on potential benefits of the entity (i.e., relief).

To mitigate this issue, we build a neural embedding for query intent detection (query embedding). The key intuition in our query embedding is to infer latent intent based on observed post-query clickstream behavior for a subset of users. For instance, if a user goes to a number of health sites after a query, it is more likely a signal than if they visit celebrity news sites. To this end, a classification model trained on this query-clickstream interplay is used on the auxiliary task of determining the category relevance of sites for post-query clicks, using the model's inner representation to derive our query embedding. Details are as follows.

For this task, we require a high-level taxonomy of websites by topic to categorize what type of information a user clicked after a query. For our system, we used DMOZ, a crowd-sourced hand-labeled directory of thousands of websites commonly employed by researchers for such tasks [14], [24]. We categorize queries as relevant if at least one of the URLs visited within a time window after the query has a category germane to the entity. For instance, if detecting adverse drug events, "health" categories in DMOZ [61] would be considered relevant. Using this procedure, we build a labeled dataset of query-relevance pairs and train a Transformer [49] model for query categorization. Transformers were used since we want to focus on query intent, and such models use self-attention effectively to capture context. The inner representation (i.e., last linear layer of the Transformer) is used as the query embedding. As shown in Section 5.3, this query embedding significantly improves our event detection capabilities.

Fig. 2 shows the query classifier used to derive our query embedding. Words along with their positional encoding are input to a Transformer architecture which converts them into dense vector representations called embeddings. These embeddings are fed to a multi headed attention layer which focuses on the important information in the query and assigns attention weights to each word. This allows us to attend to the important parts of the query during learning. The attention weights are finally used with a feed forward neural network which converts everything into a 1D vector representation q_o which is a global representation of the query. q_o is input to a softmax classifier for classification.

The outputs q_o, which are the output representation of the final layer encompass semantic meaning of the input queries in a high dimensional space and constitute our query embedding. In the evaluation section, we empirically show that these embeddings are semantically meaningful and help in user intent inference, thereby reducing false positives.

Formally, given a sequence of words $w_1, w_2, \ldots, w_n \in q$ in a query where w_i is a vector for word embedding [35] and p_i is a vector of positional embeddings [49], the Transformer creates three separate embeddings from the input which are called s (words of sentence/query), k (key), v (values) embeddings. These are gathered in a single matrix to give us S, K, V. An attention operation is performed on these three matrices to give us weights for each word.

$$Attention(S, K, V) = softmax\left(\frac{SK^T}{\sqrt{d_k}}\right)V. \tag{11}$$

In Equation (11), d_k is the dimension of the key embeddings. In order to further enhance the performance of attention, a multi-headed attention mechanism is used which is described by the following equation.

$$\begin{gathered} MultiHead(S, K, V) = Concat(head_1, \ldots, head_h)W^0 \\ \text{where } head_i = Attention(SW_i^S, KW_i^k, VW_i^V). \end{gathered} \tag{12}$$

W_i^S, W_i^k, W_i^V are the weights for each matrix. After applying the multi-headed attention to get attention weights for each word, a feed forward neural network is applied to aggregate embeddings in S into a 1D vector representation q_o which is used with a softmax layer for classification. q_o is called the query embedding. Finally, q_o is passed to a feed-forward layer followed by a softmax layer for classification.

$$L = W_Q * q_o + b \tag{13}$$

$$P_{class=c} = \frac{e^{Lc}}{(\sum_{j=0,n} e^{Lj}}. \tag{14}$$

We maximize the cross entropy loss function to train this model. Given θ as parameters of the model, loss function is given by

$$L(q; \theta) = log(P_c | w_1, w_2, \ldots, w_n). \tag{15}$$

3.5.1 Analysis of Query Embeddings

In the same vein as other neural embeddings, we extract the outputs of the Transformer q_o and use them as our query embedding. To illustrate the potential value of the proposed embedding, similar to prior embedding studies, we examined

TABLE 2
Clusters Encompassing Different Types of Queries

Example Clusters	Sample Queries within Clusters
1. Not relevant to the entity	*urex* iphone dvd ripper, battery *portalac*, chicago *metra* train schedule, *ibuprofen* coupon
2. Broad preliminary information searches	*will* prozac work, *can* cefadroxil treat chlamydia, *can* you abuse subutex, *can* cromolyn be substituted for prednisone
3. Specific adverse outcome searches	*bactrim* and sudden death, *accutane* and headaches, *neostigmine bromide* for sexual anixety, *cisplatin* delayed nausea
4. Clarification seeking searches	*why* albuterol causes jitteriness, *how* do i fix fentanyl withdrawal, *how* does elavil effect pain management
5. Specialized question-related searches	*if* allergic to tylenol what can i take instead, *if* allergic to penicillin would it not rid strep, *if* i take progesterone and feel nauseated
6. Value proposition and effectiveness of entities	*abilify* for depression reviews, *abreva* cold sore treatment reviews, *garcinia cambovia* weight loss review

the semantic composition of closely related groups of queries to see how effectively they captured diverse query intent information. We performed this analysis by analyzing the nearest neighbors of queries and clustering them via k-Means to find patterns.

Table 2 shows results from a partitional cluster analysis using k-Means, applied on query embeddings derived from searches related to a health context (i.e., where the entities are pharmaceutical drugs and outcomes of interest are drug reactions). The first column depicts the relevance or intent of queries within the cluster, whereas the second column shows sample queries for each cluster. Italics are used to highlight certain intent-related facets of each cluster. Looking at the table, if the goal were to identify adverse drug events, Clusters 3 and 4 seem especially relevant since users are searching for information or clarification about adverse drug reactions. Some queries from Cluster 2, which seem to be broader preliminary searches, might also relate to adverse events. In contrast, Cluster 5 seems to be highly specialized searches from prospective users of the drug entities. Similarly, Cluster 6 is seemingly asking about the possible value of a given drug from pre-experience users. Cluster 1 contains queries completely irrelevant to the adverse drug event entity-outcome tuple context. It is also worth noting that the resulting intent clusters appear to be differentiated on the basis of not only entity and outcome composition of the queries, but also their stop/function word presence (e.g., is, can, how, if). By leveraging a classifier that connects queries to subsequent URL clickstream behavior, the proposed query embedding is able to identify subtle intent patterns. As alluded to, later in the evaluation section, we show that inclusion of query embeddings in DeepSAVE enhances overall event detection precision and recall. Moreover, our query embedding also offers better performance relative to alternative query classification approaches.

3.6 User Modeling

User heterogeneity is an important aspect of search [7]. Different users seek information in different ways. For instance, continuing with our health example, if we consider drugs as entities and reactions as outcomes, some users are paranoid about their health and frequently seek medical information for drugs and reactions (i.e., hypochondriacs) [54]. Similarly, people taking multiple drugs are at greater risk of drug-drug interactions, which may result in differences in search patterns. Given the anomaly identification nature of adverse event detection, accounting for heterogeneity in user characteristics is important for disentangling signal from noise. However, many of these user characteristics are not observable, but rather latent factors that influence user behaviors. By taking into consideration how these latent characteristics vary across a heterogenous population, we can significantly improve detection of adverse events from user searches. Because we need to estimate latent, unobserved characteristics that are heterogenous across users, we turn to hierarchical Bayesian models. Bayesian techniques have been used extensively in social science literature to create explanatory models that assume observed behaviors are, in part, functions of unobserved heterogenous traits or opinions such as aptitude or utility [62]. These models are ideal for this type of inference because they allow for the structured distribution of latent factors across users to be estimated simultaneously with the impact they have on discretely observed behaviors [5]. Bayesian techniques have been used previously in adverse event detection, but not, to our knowledge, in this way. For instance, the Multi-Item Gamma Poisson Shrinker algorithm uses Bayesian estimation to hierarchically model reporting ratios for adverse events as draws from a population of true, unknown values [15]. Another study utilizes prior specifications within the Bayesian framework to incorporate domain knowledge into the predictive model [33]. Bayesian network structures have also been used for estimating conditional probabilities for predicting adverse events and medical diagnoses [6], [38]. Because of its various strengths, there is increasing interest in incorporating Bayesian techniques into sophisticated predictive models [52].

3.6.1 User Embeddings

Similar to how query embeddings signify the semantic meaning/intent of queries, we develop user embeddings to represent the individualized search behaviors of users. Accordingly, we develop a hierarchical Bayesian model [5] to identify heterogeneity in users' latent information seeking propensities for various categories of entities and outcomes. Specifically, the model predicts what type of site a user will visit (i.e., healthcare or other) after searching for each category of drugs and reactions. We then use the user-specific coefficients for each drug and reaction in our predictive model to represent latent user characteristics. For example, in our adverse drug event detection example context, assume some users are hypochondriacs who search for and seek information from healthcare sites for certain drugs and reactions very frequently (a latent user characteristic). These users are not likely to provide a good signal for adverse event detection. Another group of more normal users, who may provide a more reliable signal, may search for drug or reaction terms less frequently and visit fewer health related sites when they do. When a user in this second group does experience an adverse reaction, searches

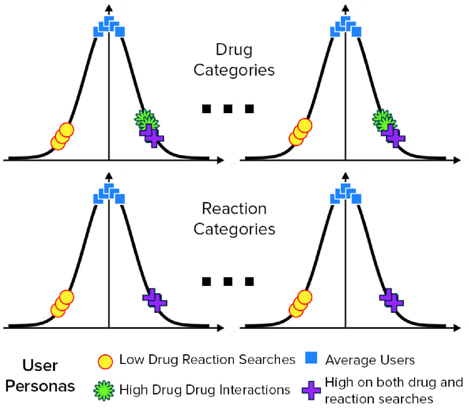


Fig. 3. Intuition for Bayesian model used to create user embeddings.

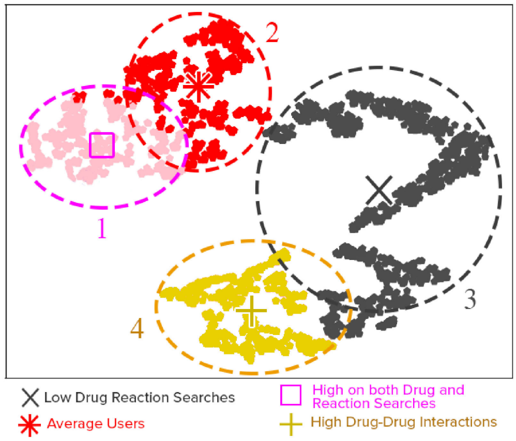


Fig. 4. Cluster analysis of user embeddings.

for a drug-reaction tuple, and visits health-related sites to obtain information, the VAE can use the coefficients representing such latent characteristics to adjust signal strength and emphasize information from these more reliable users, significantly improving performance.

Fig. 3 illustrates how the Bayesian model quantifies this intuition in the context of our health example. Each Gaussian distribution represents a particular category of drug (entity) and reaction (outcome). Each user is represented along each category distribution by a beta value that indicates their placement relative to others – a user's vector of betas across entity and outcome categories can shed light on behavior patterns. Users lying on the mean of the distribution are average users that might not exhibit any atypical properties. However, users on the high end of the distribution might express patterns like hypochondria or high drug-drug interactions.

The input to the Bayesian model is categories of entity and outcome tuples. The output is the likelihood of the user visiting a target category website (e.g "Health" if working with adverse drug events). The model employs logistic regression where the weights are estimated by allowing the coefficients to vary randomly by users according to a Gaussian distribution with mean and standard deviation freely determined by the model. Formally, let S_{jk} denote a binary variable indicating if a user k visited a particular site type on their jth search. Let C_{jk} be the search category for an entity-outcome tuple for user k on jth search. Then, the model is defined by the following equation.

$$P(S_{jk}C_{jk}\ \forall\ i) = \frac{1}{(1 + exp^{-}(\beta_{0k} + \sum_{i=1,n} \beta_{ik} * S_{jk} + e_{jk}))} \tag{16}$$

$$e_{jk} \sim N(0, \sigma^2),\ \ \beta_{ik} \sim N(\bar{a_{ik}}, \sigma_k^2)\ \forall\ i = 0 \ldots n. \tag{17}$$

In the above equations, i denotes the total number of categories for entity-outcome tuples, j denotes the total observations of a user, and k denotes the total number of users. After training, we are only concerned with the values of β, which we call "user betas". For every user u, we create a vector V_u signifying our user embedding, containing beta values for the user for every entity and outcome category that the user belongs to. Along with aggregated user level features f which will be discussed in an upcoming section, we add user embeddings for all users of an input entity-outcome tuple into a matrix representation in DeepSAVE, which we denote as M_u.

3.6.2 Analysis of User Embeddings

Although user embeddings consist of user betas which are latent constructs, we conducted a partitional cluster analysis to dive deeper into the patterns these embeddings exhibit. For all users in our corpus, we clustered them into k regions via k-means and converted their user embedding vector into 2 dimensions using t-distributed stochastic neighbor embedding (TSNE) for visualization.

Fig. 4 depicts the clusters created using the aforementioned process. The partitioning confirms that there are indeed separate regions of users based on their user embeddings. On manual inspection of cluster centroids, we found that the four clusters corresponded to users with different beta values, indicating different behaviors. For instance, users in the first cluster have high beta values for both drug and reaction categories, implying frequent medical searches, while cluster 4 corresponds to users with only high drug beta values. On the other end, cluster 3 contains users with small beta values, i.e users with searches that infrequently lead to health sites. Cluster 2 contains somewhat "average" users – those with beta values closer to the mean. These embeddings are intended to enhance the regularization capabilities of the VAE in DeepSAVE, as described later in the evaluation section. They are also reconstructed by the VAE.

3.6.3 Aggregated User Features

Consistent with prior adverse event detection studies involving user-generated time series data, we use window-level aggregated time series data to account for natural spikiness and smooth out data sparsity [2], [3]. Entity-outcome co-occurrences are converted into aggregate-level features as depicted in Table 3. We rely on a small set of meaningful features, some of which have been used in previous literature [56], [59]. These features, which provide a small but dense representation of users' collective search behavior, are input to the VAE via the feature + user embedding matrix M_u and are reconstructed at the output. In DeepSAVE, their respective reconstruction errors are used as the basis for detecting adverse events.

TABLE 3
Formulas for User Features

Feature	Formula
Average Freq.	$f^u_{ave} = ((\sum_{t \in W} f^u_{e_i o_j t}))/(\|W\|)$
Frequency Variation	$\sqrt{\sum_{t \in W}(f^u_{e_i o_j t} - f^u_{ave})/(\|W\| - 1\|)}$
Weighted Sum	$\sum_{t \in W}(f^u_{e_i o_j t}/(max(t \in W) - t))$
Entity Query Prop.	$\|Q^u_{e_i o_j}\|/\|Q^u_{e_i}\|$
Outcome Query Prop.	$\|Q^u_{e_i o_j}\|/\|Q^u_{o_j}\|$
Web Time	$\sum_{q \in Q^u} d_q$
Target Category URLs	$\sum_{q \in Q^u} c_q$
Tuple Freq.	$\sum_{t \in W} f^u_{e_i o_j t}$
User Embeddings	V_u

TABLE 5
User Search Query Data Statistics

Search Data Statistics	Health & Automotive
Unique Users	2,357,854
Total Entity Queries (health & automotive)	75,522,063
Entity Queries/Month	1,161,877
Average Entity Queries/User	32.03
Query-related URLs Visited	535,580,255
Average Entity URLs/Month	8,239,696
Average Entity URLs/User	227.15

Formally, For each user u, let Q^u be the set of all queries that user performed and $Q^u_{e_i o_j} \in Q^u$ be the set of queries that user performed related to the entity-outcome tuple $e_i o_j$. Let $f^u_{e_i o_j t}$ be the frequency with which the user u performs queries related to $e_i o_j$ on day t within a time window W. Let c_q be the number of target category websites visited after query $q \in Q^u$, and d_q be the duration of time spent on those sites. Table 3 shows formulas for user features. Feature V_u represents user embeddings. We hypothesize that along with other user features, reconstructing user embeddings forces the VAE to learn the interplay between users' benign and anomalous search behavior. As we empirically show in the results, these features significantly enhance detection performance.

4 EVALUATION

4.1 Test Bed

Two types of data were incorporated in our evaluation test bed. The first were three event databases comprising over 800 verified adverse events from the US Food and Drug Administration (FDA), Health Canada, and the US National Highway Transportation Safety Agency (NHTSA). The FDA and Health Canada databases comprise adverse drug events, whereas the NHTSA database are adverse automotive events. For each event, the databases provided a detailed description of the event along with a timestamp indicating when they internally discovered the incident. We included all events appearing from 2013 through 2018. Table 4 summarizes the event database. As noted earlier, in the FDA and HealthCanada contexts, entities are drugs whereas in the NHTSA data they are vehicle makes and models (e.g., "Toyota Prius"). Since the same entity can be associated with multiple adverse events at different points in time, in Table 4, *unique entities* signify the set of non-redundant entity appearances in the event data sets. *Potential entity-outcome tuples* are all unique entity-outcome tuples appearing in the search data at least once related to these unique entities. These tuples constitute the total hypothesis space for DeepSAVE and comparison models - all true/false positives and negatives are a subset of these tuples.

TABLE 4
Adverse Event Data Statistics

Data Set	No. Events	Unique Entities	Potential Entity-Outcome Tuples
FDA	426	210	62,351
HealthCanada	234	131	19,160
NHTSA	290	79	19,168

The second type of data in our test bed set encompassed user-generated data provided by Comscore. Comscore maintains a panel encompassing over 2 million users. All search queries and clickstream behavior for these users are tracked, along with user demographics. This data affords opportunities for examining search context considerations such as intent and user modeling. Table 5 summarizes the panel-based search and clickstream data. The *total entity queries* signify the number of search queries performed by the users in the panel that encompass an entity term related to the events databases described in Table 4. The *entity-query URLs visited* are the total number of URLs visited as a result of these queries. In the evaluation, we applied DeepSAVE and comparison methods on this user panel data to detect events in the event database.

4.2 Metrics

We adopted four evaluation metrics commonly employed in prior ADE detection studies [55], [56]: precision, recall, f-measure, and timeliness. Precision and recall measure the ability to accurately identify adverse events. Recall denotes detection rate, while precision is a measure of false positive rate, with implications for alert fatigue. F-measure is the harmonic mean of precision and recall. Timeliness is how much earlier an adverse event can be detected, in comparison to the point in time when the event is timestamped in the official event database. Since our task entails identifying adverse events earlier, when calculating recall and precision, positives were only those signals that occurred prior to the first-report date for that particular event in the gold standard database. More formally, precision and recall were computed as TP/(TP+FP) and TP/(TP+FN), respectively, where TP were all events detected earlier than their database timestamp.

For timeliness, suppose we have n events $e_1, e_2, \ldots, e_n$ under consideration with each event having a timestamps $t_{d1}, t_{d2}, \ldots, t_{dn}$, the date it was officially defined as a true adverse event (by the FDA or relevant organization), and timestamp $t_{r1}, t_{r2}, \ldots, t_{rn}$, the initial inception (e.g., drug release) date of the entity or the inception date of our query dataset, 1/1/2012, whichever is later. Let m reflect the number of events the algorithm identifies prior to their official

TABLE 6
Summary Results for DeepSAVE

Method & Type		FDA				HealthCanada				NHTSA			
Method	**Type**	**Fmeas**	**Prec**	**Rec**	**Timely**	**Fmeas**	**Prec**	**Rec**	**Timely**	**Fmeas**	**Prec**	**Rec**	**Timely**
Dis-proportionality Analysis	ROR [39]	13.0	11.4	15.2	0.63	7.0	8.7	5.9	0.67	17.1	18.5	16.0	0.67
	PRR [13]	12.5	11.2	14.0	0.62	6.9	8.3	5.9	0.66	17.1	18.5	16.0	0.67
	RRR [8]	11.6	11.6	11.6	0.62	9.5	13.2	7.5	0.67	18.0	19.7	16.5	0.66
	IC [8]	12.0	14.0	12.2	0.61	9.5	13.2	7.5	0.67	18.0	19.7	16.5	0.65
	MGPS [43]	16.4	15.9	16.8	0.61	9.4	14.6	6.9	0.65	19.4	20.1	18.8	0.66
DA atop Event Mention Classifier	SVM [41]	7.4	15.7	4.9	0.59	2.2	13.0	1.2	0.47	As noted, supervised classifiers couldn't be run due to lack of labeled automotive training data			
	CRNN [17]	13.0	11.4	15.2	0.63	7.0	8.7	5.9	0.71				
	CNN [17]	13.6	15.3	12.2	0.64	7.2	12.0	5.1	0.63				
	FASTTEXT [22]	3.0	7.9	1.8	0.57	2.9	23.5	1.6	0.66				
Association Rule Mining	LEV [19]	14.1	13.8	14.3	0.48	9.6	13.1	7.6	0.62	15.8	16.5	15.2	0.67
	CL [45]	13.8	14.2	13.5	0.56	9.2	13.0	7.1	0.65	17.0	17.2	16.8	0.68
	ECL [19]	16.5	16.7	16.4	0.45	10.4	14.5	8.1	0.69	18.0	18.8	17.3	0.70
	EXCLEV [21]	16.6	17.2	16.1	0.53	10.6	14.2	8.5	0.67	19.1	19.2	19.1	0.71
Twitter Event Detection Methods	PyMABED [36]	21.0	13.1	**52.6**	**0.75**	20.7	16.2	28.6	**0.86**	25.1	20.6	32.1	0.76
	T-Topic [18]	20.9	13.5	46.7	0.64	20.8	16.3	29.0	0.81	26.2	24.7	28.0	0.72
	SEDTWik [34]	19.2	**34.1**	14.6	0.68	15.2	29.9	10.2	0.86	22.1	**31.8**	16.9	**0.80**
	PeakLabel [3]	18.7	17.1	20.7	0.69	17.5	17.0	17.9	0.71	22.7	19.9	26.6	0.73
Auto Encoders & Dimensionality Reduction	PCA [54]	27.1	20.9	38.5	0.65	19.2	16.1	23.9	0.76	25.1	19.8	34.2	0.77
	SVD [15]	26.8	20.5	38.7	0.64	19.7	16.7	23.9	0.76	25.1	19.8	34.2	0.77
	AE [16]	15.9	11.8	24.4	0.57	13.7	28.8	9.0	0.71	21.6	14.2	45.9	0.69
	DAE [47]	22.4	17.2	32.3	0.64	15.9	**35.6**	10.2	0.63	20.3	13.1	44.4	0.71
	VAE [25]	25.5	24.5	26.7	0.63	14.8	32.2	9.6	0.68	20.4	13.2	44.7	0.70
DeepSAVE		**35.2**	26.6	52.0	0.73	**26.0**	21.1	**33.8**	0.74	**37.0**	27.1	**58.3**	0.74

recognition date, where $m \leq n$, with timestamps $t_{a1}, t_{a2}, \ldots, t_{am}$ representing when the algorithm would have predicted the event based on available data. The timeliness metric is measured as the average time between detection by the algorithm and official recognition, normalized by the length of time from entity/data inception to official recognition of the event i.e $(\sum_{i=1,\ldots,m}(t_{di} - t_{ai})(t_{di} - t_{ri}))/m \in [0, 1]$. Normalization was performed in order to provide a more equal footing to events officially recognized at various times. However, we can also calculate the timeliness in months/days by removing the normalization factor.

Consistent with prior studies [2], [3], for each such identified positive signal, a determination of true/false positive (i.e., TP or FP) was made using a two-stage approach. First, the key entity and outcome keywords appearing in the signal were automatically compared against those appearing in the gold standard database descriptions. If the similarity was below a certain threshold, the signal was automatically rejected as a false positive. For those above a threshold, an independent domain expert examined a sample of documents pertaining to the signal (e.g., the underlying queries and URLs) to determine relevance.

4.3 Implementation Details

For DeepSAVE's query embeddings, Transformer [49] was used with 128 units in the hidden layer, resulting in 128-dimensionsal query embeddings. For the user modeling module, we mapped all outcomes and entities into a few categories. A Bayesian model was trained per each entity-outcome category tuple, across all relevant users. Each Bayesian model included six beta values per user: entity category, outcome category, entity and outcome category, other drug entities, other outcome categories, and other entity and outcome categories. The feedforward neural networks used in the architecture had 64 layers each except for at the compressed layer where we only had 16 units in the layer. Mean Absolute Error was used as the reconstruction loss measure to train and test the VAE. For the parameters, we tried out different learning rates, number of layers, and window sizes and kept the best performing ones. They impact of parameters is also discussed in Section 5.2. For DeepSAVE and all comparison methods, precision, recall, and f-measure performance on the top 10 percent reactions with highest error were reported in the main results table, although we also plotted these measures across a broader range of thresholds to show performance domination.

4.4 Comparison Methods

As noted, similar to DeepSAVE, the top n% reactions with highest DA measures, ARM measures, and AE reconstruction errors were kept for testing against the gold standard database. For the classification-based comparison methods (SVM, CRNN, CNN, FASTTEXT), classifiers were trained and tuned on a labeled adverse drug mention data set [44]. However, these techniques couldn't be run on the automotive test bed due to lack of labeled training data. CNN [19] and CRNN [19] were run using a window size of 5. For CRNN, a single layer with 128 recurrent units was employed. FASTTEXT [25] used 100 sized word vectors with a window size of 5 and 0.1 learning rate. AE [18] was run using a 3 layer neural network with 8 units in the compressed layer and 128 in the remaining ones. For DAE [50], we added small random Gaussian noise (0.01*N(0,1)) to the input. For VAE, we used the same architecture as AE but added a Gaussian constraint on the compressed layer. For twitter event detection methods, we used the default parameters.

5 Experimental Results

In all results tables, f-measure, precision, and recall are reported as percentages. The "Timely" column denotes the average timeliness of the true positives, as described in Section 4.2. Table 6 presents the experiment results for DeepSAVE and 22 benchmark methods. DeepSAVE significantly

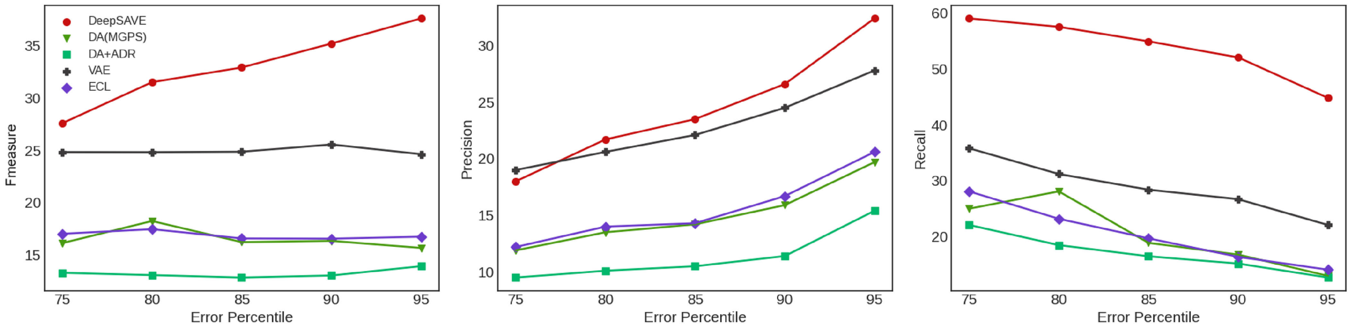


Fig. 5. Effect of reconstruction error threshold on f-measure, precision, and recall performance. Twitter Event Detection methods were omitted since they don't have thresholds.

outperformed all benchmark methods on f-measure and recall on all three data sets. On these metrics, DeepSAVE performed at least 6 to 10 percentage points better than all comparison methods. Most notably, recall rates were two-three times higher than many comparison techniques. In terms of precision, DeepSAVE was close to other methods on the FDA and NHTSA test beds. DeepSAVE's far superior true positives with relatively decent false positive rates is crucial since recall is considered essential for adverse event detection [2]. In general, auto encoders and twitter event detection methods produced better results relative to DA, association rule, and classifier techniques. Interestingly, entity-outcome classifiers coupled with DA methods did not work well. Overall, the results underscore the effectiveness of DeepSAVE for search-based adverse event detection relative to DA, association rule, classifier, and standard VAE methods. In the case of NHTSA, supervised entity-outcome classifiers couldn't be run due to lack of labeled automotive training data. Therefore, there are no results for Disproportionality analysis atop Mention Classifier.

5.1 Effect of Threshold

Fig. 5 depicts the effect of test reconstruction loss error on the performance of DeepSAVE versus the best benchmark methods in each comparison category on the FDA data. Results on NHTSA and Health Canada were similar. As we increase the threshold for reconstruction error, thereby making the number of positive signals generated fewer and more selective, f-measure steadily increases for DeepSAVE while it decreases or remains constant for many comparison methods. This can be seen by looking at the recall and precision figures – recall for DeepSAVE is at least 25 points higher than other methods while precision is at most 5 points lower than VAE. Since recall is markedly higher and precision is slightly lower at every threshold, but with a steep increase for higher thresholds, DeepSAVE's f-measures dominate top comparison methods across a wide range of event detection thresholds. These results suggest that DeepSAVE's performance gains are robust across a wide range of thresholds.

5.2 Parameter Impact

Like any deep learning model applied to time series data, the DeepSAVE framework includes a few parameters such as the learning rate (lr) of the model, number of convolutional layers, and the time series sliding window size for analysis (in months). In order to examine the impact of different parameter values on performance, we examined various combinations of layers (1,2,3), learning rate (0.001,0.0005,0.0001), and window size (4,6,8) on our three data sets. While in Table 6 we report *lr = 0.0005, layers = 1, and window size = 4*, in general, we found that the total impact on f-measures across these 27 parameter settings was less than 2 percentage points on all three data sets. To illustrate this point, we show the results on the NHTSA events, Fig. 6, where the greatest variance was observed. As depicted, DeepSAVE's performance was fairly robust to changes in learning rates and number of convolutional layers. With respect to sliding window size, the 4 and 6 month windows garnered fairly similar f-measures (i.e., within 1 percentage point). Increasing the window size to 8 months did reduce the f-measure by about two points for a couple of settings. Though not depicted in this figure, timeliness values for all these settings also remained similar, as did the precision and recall profiles, across all three event data sets. Collectively, these results suggest that DeepSAVE is robust across an array of parameter settings.

5.3 Ablation Analysis

DeepSAVE encompasses novel query and user embeddings. In order to analyze the additive impact of each component, Table 7 shows the performance of the model as we incrementally added components on top of a baseline VAE for the FDA, Health Canada, and NHTSA event data. Adding either query embeddings or user embeddings gives a 3 to 13 point lift in f-measure and augments recall by upto

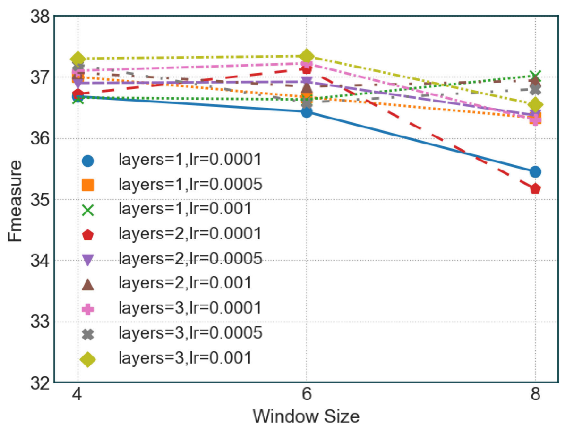


Fig. 6. Impact of parameters on DeepSAVE F-measure.

TABLE 7
Ablation Analysis of DeepSAVE Components

	FDA				Health Canada				NHTSA			
Method	Fmeas	Prec	Rec	Timely	Fmeas	Prec	Rec	Timely	Fmeas	Prec	Rec	Timely
VAE	25.5	24.5	26.7	0.63	14.8	32.2	9.6	0.68	20.4	13.2	44.7	0.70
VAE+Q	30.4	24.8	39.2	0.62	25.8	19.6	37.7	0.78	35.9	25.2	**62.4**	0.74
VAE+U	34.5	26.1	51.2	0.72	24.9	19.9	33.3	0.75	35.7	25.7	58.2	**0.77**
DeepSAVE	**35.2**	**26.6**	**52.0**	**0.73**	**26.0**	**21.2**	33.8	0.74	**37.0**	**27.1**	58.3	0.74

50 percent. These results highlight the notion that provisions for better understanding query intent and user heterogeneity are invaluable for better contextualized search-based event detection. Lastly, combining both embeddings in the feature matrix gives us the final results for DeepSAVE with a further increase in recall, precision and f-measure. The results in Table 7 underscore the fact that all components of DeepSAVE contribute to its overall performance.

5.3.1 Performance of Query Embeddings

In order to further examine the effectiveness of the proposed query embeddings, we compared it against three other query embedding methods: mean embeddings [30], query2vec [26], and using LSTM instead of Transformers in our query classifier. We did this by replacing the query embedding matrix in DeepSAVE with embeddings from the selected methods on the FDA event data. The results appear in Table 8. DeepSAVE query embeddings outperform both other methods on all four metrics, with performance lifts of 2 to 12 percentage points. These results, which were also observed on NHTSA and Health Canada, support the value of the proposed query embeddings for search-based event detection.

5.3.2 Effect on AutoEncoder

The above results show the effectiveness of the query and user embeddings from a performance metric perspective. Digging deeper into their inner workings, Fig. 7 depicts their effect on reconstruction loss (error) distribution for the enriched VAEs. We illustrate this using the entity matrices for 4 randomly chosen drugs in the data. The figure shows the original error (loss) distributions as well as those after adding the query embedding atop the VAE. Without the query embeddings, the VAE reconstruction error is much more compressed. After adding the query embeddings, the error distribution becomes less compressed, better reflecting the intended Gaussian shape. By smoothing out the Gaussian distribution, the query embeddings help reduce the number of false positives incurred at different thresholds.

TABLE 8
Ablation Analysis for Query Embeddings

Method	Fmeas	Prec	Rec	Timely
Mean Embeddings [30]	28.3	22.1	39.5	0.65
Query2Vec [26]	28.7	23.2	37.8	0.63
LSTM Embeddings	*34.2*	*26.7*	*47.6*	*0.68*
DeepSAVE Q.Embed.	**35.2**	**26.6**	**52.0**	**0.73**

Similar to the query embeddings, we further analyze the viability of our user embeddings by examining the VAE reconstruction loss error distributions on the same drug entities (bottom of Fig. 7). The stark difference between error distributions in VAEs with and without the user embeddings is clearly visible. After the inclusion of user embeddings atop query embeddings, the error distribution still follows a Gaussian shape, but becomes less compressed, with more of a long tail towards the right (higher loss). This is advantageous since by dispersing the distribution of losses, the resulting model is able to more easily discern high reconstruction loss cases (i.e., possible true positives). Upon manual inspection, it was found that most of the points that are on these long tails were indeed true positives thus highlighting the efficacy of user embeddings in improving model performance. These results speak to the intended regularization benefits of adding query and user embeddings to the enriched VAE.

5.4 Case Study: Stock Movement Events

The three data sets used in our evaluation encompass adverse events. However, search context factors such as user heterogeneity and query intent may manifest in other event detections settings as well. In order to examine the effectiveness of DeepSAVE in such contexts, we compared its performance against two of our top benchmark methods - pyMABED and SEDTWiK - on a stock movement event detection task. Following prior studies that dealt with event detection in stocks [47], our events were stocks publicly traded on the NASDAQ, S&P 500, and Russell 2,000 which had attained significant gains or losses over a certain period of time. Hence, the entities were companies and outcomes were upward or downward stock price movement. We defined our events as stocks which gained or lost 20 percent within a 6-month time period. These values were chosen based on prior literature, and since these values resulted in a quantity of events and entities that were in the same range as our adverse event data sets. Overall, the stock movement event data set was comprised of 330 events related to 100 unique entities, spanning the time period 2016-2018.

Similar to the approach described in Section 4.1, we used our existing search log corpus to derive entity queries. Examples of queries include 'Amazon has poor support' and 'shorting Tesla stock'. For DeepSAVE and the comparison methods, we then derived potential event signals and compared them against the events to compute precision, recall, f-measure, and timeliness. Table 9 shows the results for DeepSAVE and the two aforementioned comparison benchmark methods. It is worth noting that the overall results were higher since such macro time-period stock movement events are generally considered easier to detect relative to adverse

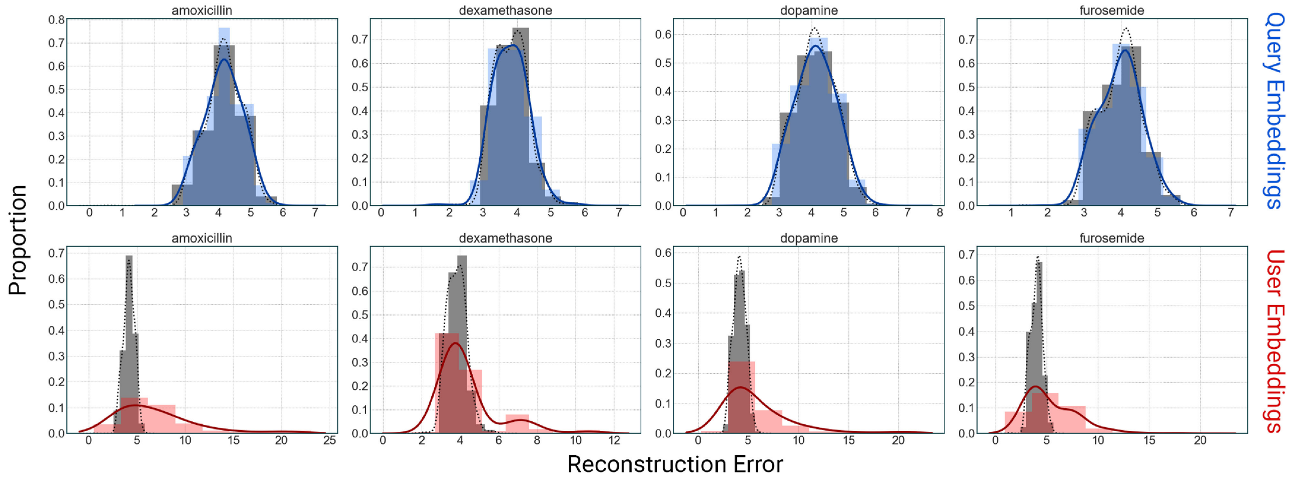


Fig. 7. Effect of user and query embeddings on VAE reconstruction loss.

events. Nevertheless, DeepSAVE attained a 10-30 percentage point lift in f-measure and recall, and also garnered higher precision. The case study further underscores the importance of holistic methods for search-based event detection that take into account search context factors.

6 Conclusion

In this paper, we proposed DeepSAVE, a novel deep learning framework for adverse event detection from web search query logs. DeepSAVE uses an enriched variational autoencoder comprising of novel query embeddings for enhanced contextualization via intent clarification and user-level modeling to account for heterogeneous adverse experiences. Evaluation on three event databases in the health and automotive domains encompassing nearly 1,000 adverse events reveals that DeepSAVE garners enhanced recall and f-measures relative to existing state-of-the-art adverse event detection methods. Given the lack of prior work on application of novel autoencoder architectures for this problem, the results contribute to the nascent body of knowledge on advanced machine learning methods for adverse event detection. DeepSAVE has important practical implications. For example, it could be used to detect adverse events in various critical contexts such as disease surveillance, socio-political incidents, product defect identification, and e-commerce. While we focused mostly on adverse event contexts, our case study on financial events suggests that the proposed method might also be suitable for more general-purpose event detection problems. We believe this study signifies an important first step toward these directions.

TABLE 9
Results on Stock Dataset

Method	Fmeas	Prec	Rec	Timely
pyMABED [39]	*71.6*	85.4	*61.6*	*0.40*
SEDTWik [37]	60.9	*86.7*	46.8	0.38
DeepSAVE	**81.9**	**89.6**	**75.4**	**0.40**

Acknowledgments

This work was supported in part by U.S. NSF Grants: IIS-1553109, IIS-1816504, BDS-1636933, CCF-1629450, IIS-1552860, and IIS-1816005. The research was also partially funded through MOST Grants 2019AAA0103405 and 2016QY02D0305, NNSFC Innovative Team Grant 71621002, and CAS Grants XDC02060600 and ZDRW-XH-2017-3.

References

[1] A. Abbasi and D. Adjeroh, "Social media analytics for smart health," *IEEE Intell. Syst.*, vol. 29, no. 2, pp. 60–64, Mar./Apr. 2014.
[2] A. Abbasi, J. Li, D. Adjeroh, M. Abate, and W. Zheng, "Don't mention it? Analyzing user-generated content signals for early adverse event warnings," *Inf. Syst. Res.*, vol. 30, no. 3, pp. 1007–1028, 2019.
[3] D. Adjeroh, R. Beal, A. Abbasi, W. Zheng, and M. Abate, "Signal fusion for social media analysis of adverse drug events," *IEEE Intell. Syst.*, vol. 29, no. 2, pp. 74–80, Mar./Apr. 2014.
[4] R. Agrawal *et al.*, "Fast algorithms for mining association rules," in *Proc. 20th Int. Conf. Very Large Data Bases*, 1994, vol. 1215, pp. 487–499.
[5] G. M. Allenby and P. E. Rossi, "Hierarchical bayes models," in *The Handbook of Marketing Research: Uses, Misuses, and Future Advances*, Thousand Oaks, CA, USA: SAGE, 2006, pp. 418–440.
[6] A. Bate, M. Lindquist, I. R. Edwards, and R. Orre, "A data mining approach for signal detection and analysis," *Drug Saf.*, vol. 25, no. 6, pp. 393–397, 2002.
[7] T. Bodnar, C. Tucker, K. Hopkinson, and S. G. Bilén, "Increasing the veracity of event detection on social media networks through user trust modeling," in *Proc. IEEE Int. Conf. Big Data*, 2014, pp. 636–643.
[8] R. Böhm, "Primer on disproportionality analysis," [Online]. Available: https://pdfs.semanticscholar.org/3d1e/0589a9a17386587fc9790f63c3e4860ea2a4.pdf
[9] D. A. Broniatowski, M. J. Paul, and M. Dredze, "Twitter: Big data opportunities," *Science*, vol. 345, no. 6193, pp. 148–148, 2014.
[10] E. Brynjolfsson, T. Geva, and S. Reichman, "Crowd-squared: Amplifying the predictive power of search trend data," *MIS Quart.*, vol. 40, pp. 941–961, 2016.
[11] D. Butler, "When Google got flu wrong: US outbreak foxes a leading web-based method for tracking seasonal flu," *Nature*, vol. 494, no. 7436, pp. 155–157, 2013.
[12] H. Choi and H. Varian, "Predicting the present with Google trends," *Econ. Rec.*, vol. 88, no. 1, pp. 2–9, 2012.
[13] S. J. Evans, P. C. Waller, and S. Davis, "Use of proportional reporting ratios for signal generation from spontaneous adverse drug reaction reports," *Pharmacoepidemiology Drug Saf.*, vol. 10, pp. 483–486, 2001.
[14] S. Flaxman, S. Goel, and J. M. Rao, "Filter bubbles, echo chambers, and online news consumption," *Public Opinion Quart.*, vol. 80, no. S1, pp. 298–320, 2017.

[15] D. M. Fram, J. S. Almenoff, and W. DuMouchel, "Empirical Bayesian data mining for discovering patterns in post-marketing drug safety," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2003, pp. 359–368.

[16] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," in *Linear Algebra*, Berlin, Germany: Springer, 1971, pp. 134–151.

[17] M. Hasan, M. A. Orgun, and R. Schwitter, "A survey on real-time event detection from the Twitter data stream," *J. Inf. Sci.*, vol. 44, no. 4, pp. 443–463, 2018.

[18] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[19] T. Huynh, Y. He, A. Willis, and S. Rüger, "Adverse drug reaction classification with deep neural networks," in *Proc. 26th Int. Conf. Comput. Linguistics*, 2016, pp. 877–887.

[20] G. Ifrim, B. Shi, and I. Brigadir, "Event detection in Twitter using aggressive filtering and hierarchical tweet clustering," in *Proc. 2nd Workshop Soc. News Web*, 2014, pp. 33–40.

[21] Y. Ji, R. M. Massanari, J. Ager, J. Yen, R. E. Miller, and H. Ying, "A fuzzy logic-based computational recognition-primed decision model," *Inf. Sci.*, vol. 177, no. 20, pp. 4338–4353, 2007.

[22] Y. Ji, H. Ying, J. Tran, P. Dews, A. Mansour, and R. M. Massanari, "A method for mining infrequent causal associations and its application in finding adverse drug reaction signal pairs," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 4, pp. 721–733, Apr. 2013.

[23] H. Jin, J. Chen, H. He, C. Kelman, D. McAullay, and C. M. O'Keefe, "Signaling potential adverse drug reactions from administrative health databases," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 6, pp. 839–853, Jun. 2010.

[24] B. Joshi, M. R. Amini, F. Iutzeler, and Y. Maximov, "Aggressive sampling for multi-class to binary reduction with applications to text classification," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 4162–4171.

[25] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," 2016, *arXiv:1607.01759*.

[26] D. Kang and I. Kang, "Query2Vec: Learning deep intentions from heterogeneous search logs," Carnegie Mellon Univ., Tech. Rep., 2015. [Online]. Available: http://www.cs.cmu.edu/~dongyeok/papers/querygram.pdf

[27] N. Kanhabua, T. Ngoc Nguyen, and W. Nejdl, "Learning to detect event-related queries for web search," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 1339–1344.

[28] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2013, *arXiv:1312.6114*.

[29] D. Lazer, R. Kennedy, G. King, and A. Vespignani, "The parable of Google flu: Traps in big data analysis," *Science*, vol. 343, pp. 1203–1205, 2014.

[30] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, pp. II-1188–II-1196.

[31] K. Lee *et al.*, "Adverse drug event detection in tweets with semi-supervised convolutional neural networks," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 705–714.

[32] B. Liu, W. Hsu, and Y. Ma, "Mining association rules with multiple minimum supports," in *Proc. 5th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 1999, pp. 337–341.

[33] B. Liu, Y. Li, S. Ghosh, Z. Sun, K. Ng, and J. Hu, "Complication risk profiling in diabetes care: A Bayesian multi-task and feature relationship learning approach," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 7, pp. 1276–1289, Jul. 2020.

[34] X. Liu and H. Chen, "AZDrugMiner: An information extraction system for mining patient-reported adverse drug events in online patient forums," in *Proc. Int. Conf. Smart Health*, 2013, pp. 134–150.

[35] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[36] J.-L. Montastruc, A. Sommet, H. Bagheri, and M. Lapeyre-Mestre, "Benefits and strengths of the disproportionality analysis for identification of adverse drug reactions in a pharmacovigilance database," *Brit. J. Clin. Pharmacol.*, vol. 72, pp. 905–908, 2011.

[37] K. Morabia, N. L. B. Murthy, A. Malapati, and S. Samant, "SEDTWik: Segmentation-based event detection from tweets using Wikipedia," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2019, pp. 77–85.

[38] D. Nikovski, "Constructing Bayesian networks for medical diagnosis from incomplete and partially correct statistics," *IEEE Trans. Knowl. Data Eng.*, vol. 12, no. 4, pp. 509–516, Jul./Aug. 2000.

[39] F. Odeh, "Event detection in heterogeneous data streams," Lyon, Tech. Rep., 2018. [Online]. Available: https://www.researchgate.net/profile/Firas_Odeh2/publication/327537608_Event_detection_in_heterogeneous_data_streams/links/5b9454ad299bf147392b3619/Event-detection-in-heterogeneous-data-streams.pdf

[40] M. Rafferty, X. Liu, D. M. Laverty, and S. McLoone, "Real-time multiple event detection and classification using moving window PCA," *IEEE Trans. Smart Grid*, vol. 7, no. 5, pp. 2537–2548, Sep. 2016.

[41] J. Ritter, "Minimising harm: Human variation and adverse drug reactions (ADRs)," *Brit. J. Clin. Pharmocology*, vol. 65, no. 4, pp. 451–452, 2008.

[42] K. J. Rothman, S. Lanes, and S. T. Sacks, "The reporting odds ratio and its advantages over the proportional reporting ratio," *Pharmacoepidemiology Drug Saf.*, vol. 13, no. 8, pp. 519–523, 2004.

[43] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proc. 2nd Workshop Mach. Learn. Sensory Data Anal.*, 2014, pp. 4–11.

[44] A. Sarker and G. Gonzalez, "Portable automatic text classification for adverse drug reaction detection via multi-corpus training," *J. Biomed. Informat.*, vol. 53, pp. 196–207, 2015.

[45] A. Sun and M. Hu, "Query-guided event detection from news and blog streams," *IEEE Trans. Syst., Man, Cybern. A: Syst. Humans*, vol. 41, no. 5, pp. 834–839, Sep. 2011.

[46] A. Szarfman, S. G. Machado, and R. T. O'neill, "Use of screening algorithms and computer systems to efficiently signal higher-than-expected combinations of drugs and events in the us FDA's spontaneous reports database," *Drug Saf.*, vol. 25, no. 6, pp. 381–392, 2002.

[47] E. Toth and S. Chawla, "GTΔ Detecting temporal changes in group stochastic processes," *ACM Trans. Knowl. Discov. Data*, vol. 12, no. 4, 2018, Art. no. 4.

[48] L. Troiano, G. Scibelli, and C. Birtolo, "A fast algorithm for mining rare itemsets," in *Proc. 9th Int. Conf. Intell. Syst. Des. Appl.*, 2009, pp. 1149–1155.

[49] A. Vaswani *et al.*, "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[50] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103.

[51] J. von Gordon, "Boehringer ingelheim announces comprehensive settlement of us pradaxa (dabigatran etexilate) litigation," 2014. [Online]. Available: https://www.boehringer-ingelheim.us/press-release/boehringer-ingelheim-announces-comprehensive-settlement-us-pradaxa-dabigatran

[52] H. Wang and D.-Y. Yeung, "Towards Bayesian deep learning: A framework and some existing methods," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 12, pp. 3395–3408, Dec. 2016.

[53] G. M. Weiss, "Mining with rarity: A unifying framework," *ACM SIGKDD Explorations Newslett.*, vol. 6, no. 1, pp. 7–19, 2004.

[54] R. W. White and E. Horvitz, "Cyberchondria: Studies of the escalation of medical concerns in web search," *ACM Trans. Inf. Syst.*, vol. 27, no. 4, 2009, Art. no. 23.

[55] R. W. White, N. P. Tatonetti, N. H. Shah, R. B. Altman, and E. Horvitz, "Web-scale pharmacovigilance: Listening to signals from the crowd," *J. Amer. Med. Informat. Assoc.*, vol. 20, no. 3, pp. 404–408, 2013.

[56] R. W. White, S. Wang, A. Pant, R. Harpaz, and E. Horvitz, "Early identification of adverse drug reactions from search log data," *J. Biomed. Informat.*, vol. 59, pp. 42–48, 2016.

[57] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics Intell. Lab. Syst.*, vol. 2, no. 1/3, pp. 37–52, 1987.

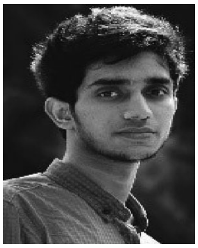
[58] H. Xu *et al.*, "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," in *Proc. WWW Conf.*, 2018, pp. 187–196.

[59] E. Yom-Tov and E. Gabrilovich, "Postmarket drug surveillance without trial costs: Discovery of adverse drug reactions through large-scale analysis of web search queries," *J. Med. Internet Res.*, vol. 15, no. 6, 2013, Art. no. e124.

[60] H. Zamani and W. B. Croft, "Estimating embedding vectors for queries," in *Proc. ACM Int. Conf. Theory Inf. Retrieval*, 2016, pp. 123–132.

[61] Y. Zhang, N. Zincir-Heywood, and E. Milios, "World wide web site summarization," *Web Intell. Agent Syst.*, vol. 2, no. 1, pp. 39–53, 2004.

[62] Z. Zheng and P. Pavlou, "Toward a causal interpretation from observational data: A new Bayesian networks method for structural models with latent variables," *Inf. Syst. Res.*, vol. 21, no. 2, pp. 365–391, 2010.



Faizan Ahmad (Member, IEEE) received the BS degree in CS from FAST NUCES in Lahore, and the MS degree in computer science from the University of Virginia, Charlottesville, Virginia. He was previously a researcher at the University of Illinois Urbana-Champaign, Champaign, Illinois, University of Iowa, Iowa City, Iowa, and worked as a data scientist at Facebook. He has received bug bounties from more than 30 companies, including Google, Microsoft, and Apple.

Ahmed Abbasi (Senior Member, IEEE) received the PhD degree from the AI Lab, University of Arizona, Tucson, Arizona. He is currently the Giovanini Endowed Chair at the University of Notre Dame, Notre Dame, Indiana. He serves as an associate or senior editor at the *IEEE Intelligent Systems*, *ACM Transactions on Management Information Systems*, and *INFORMS ISR*. He has published more than 100 articles on machine learning and NLP, and has received various grants and research awards, including the IBM Faculty, IEEE Technical Achievement and the INFORMS design science awards.

Brent Kitchens received the PhD degree from the University of Florida, Gainesville, Florida. He is an assistant professor of IT at the University of Virginia, Charlottesville, Virginia. His research interests include data analytics, health IT, and online information dissemination. He has published in top journals such as the *INFORMS ISR*, *MIS Quarterly*, and *Journal of Management Information Systems*. Before pursuing a career in academia, he worked in IT Risk Advisory at EY.

Donald Adjeroh (Member, IEEE) received the PhD degree in CS from the Chinese University of Hong Kong, Hong Kong. He is currently a professor and associate department chair at the Lane Department of CS and EE at West Virginia University, Morgantown, West Virginia. His research interests include machine learning and biomedical informatics. His work has been supported by grants from various federal agencies. He received the Department of Energy CAREER Award, in 2002.

Daniel Zeng (Fellow, IEEE) received the BE degree in operations research/economics with minor in CS from the University of Science and Technology of China, Hefei, China, and the ME and PhD degrees in industrial administration from Carnegie Mellon University, Pittsburgh, Pennsylvania. He is a professor at the Institute for Automation and School of Artificial Intelligence, Chinese Academy of Sciences, Beijing. He is also with the Shenzhen Artificial Intelligence and Data Science Institute, Shenzhen, China. His research interests include spatial-temporal modeling, online surveillance, and social computing. He previously served as editor-in-chief for the *IEEE Intelligent Systems*.