Data Mining and Knowledge Discovery (2021) 35:450–504
https://doi.org/10.1007/s10618-020-00725-5

Detecting singleton spams in reviews via learning deep anomalous temporal aspect-sentiment patterns

Yassien Shaalan[1] · Xiuzhen Zhang[1] · Jeffrey Chan[1] · Mahsa Salehi[2]

Received: 7 November 2019 / Accepted: 17 November 2020 / Published online: 2 January 2021

Abstract

Customer reviews are an essential source of information to consumers. Meanwhile, opinion spams spread widely and the detection of spam reviews becomes critically important for ensuring the integrity of the echo system of online reviews. Singleton spam reviews—one-time reviews—have spread widely of late as spammers can create multiple accounts to purposefully cheat the system. Most available techniques fail to detect this cunning form of malicious reviews, mainly due to the scarcity of behaviour trails left behind by singleton spammers. Available approaches also require extensive feature engineering, expensive manual annotation and are less generalizable. Based on our thorough study of spam reviews, it was found that genuine opinions are usually directed uniformly towards important aspects of entities. In contrast, spammers attempt to counter the consensus towards these aspects while covering their malicious intent by adding more text but on less important aspects. Additionally, spammers usually target specific time periods along products' lifespan to cause maximum bias to the public opinion. Based on these observations, we present an unsupervised singleton spam review detection model that runs in two steps. Unsupervised deep aspect-level sentiment model employing deep Boltzmann machines first learns fine-grained opinion representations from review texts. Then, an LSTM network is trained on opinion learned representation to track the evolution of opinions through the fluctuation of sentiments in a temporal context, followed by the application of a Robust Variational Autoencoder to identify spam instances. Experiments on three benchmark datasets widely used in the literature showed that our approach outperforms strong state-of-the-art baselines.

Keywords Opinion spam · Aspect sentiment modelling · Anomaly detection · LSTM · Robust variational AutoEncoder · Deep learning

Responsible editor: Johannes Fürnkranz.

Extended author information available on the last page of the article

1 Introduction

Customer reviews are an abundant source of information from the web. It has become customary for an online shopper to read online reviews written by other customers before their first interaction with a business. According to the *Local Consumer Review Survey (2016)* by BrightLocal,[1] 84% of the online shoppers were found to trust product reviews as much as personal recommendations. Positive reviews with high star ratings result in substantial financial gains for businesses, while negative reviews can cause reputation damage and financial losses. Unfortunately, opinion spammers appear amongst honest consumers, betraying the public trust implicit in reviewing platforms. In spite of raised awareness about spam reviews and the use of filters on websites, it is reported that the number of fake reviews is on the rise: for example, on Yelp, a rise from 5% in 2006 to 20% in 2013 has been recorded (Le and Mikolov 2016).

Opinion spam detection has attracted significant research interest (Ott et al. 2011; Rayana and Akoglu 2015; Sandulescu et al. 2015; Savage et al. 2015; Xie et al. 2012; Huayi et al. 2017; Kumar et al. 2018; Luyang et al. 2016). Existing approaches focus on extracting spam signals from review texts (Ott et al. 2011; Li et al. 2014; Sandulescu et al. 2015), reviewer behaviour (e.g. rating deviation, early rating, number of reviews) (Liu 2010; Savage et al. 2015), or analysing the reviewer-product networks (Rayana and Akoglu 2015; Ye and Akoglu 2017; Fei et al. 2013). To identify spammers, many approaches assume that spammers write at least several, if not many fake reviews. On many E-commerce sites, the majority of reviewers are found to be singleton reviewers (as identified by unique IDs). In fact, on many websites the proportion of singleton reviewers ranges from 60 up to 90% (Sandulescu et al. 2015; Kumar et al. 2018). Moreover, 65–70%, of reviewers on the publicly available Yelp datasets (Rayana and Akoglu 2015) are singletons. As a matter of fact, the majority of singleton reviewers are not spammers—they just represent the majority of the public. This phenomenon is the result of the lenient account registration policies of some platforms. Spammers make use of this loophole to set up multiple 'sock-puppet' accounts to hide their malicious intent and avoid detection.

Existing approaches based on reviewer behaviour and reviewer-product networks are not very effective in detecting singleton spam reviews. This can be attributed to the lack of behavioural clues left behind by such reviewers. There are two main approaches in the literature focusing singleton spam detection. One approach exploits review texts (Ott et al. 2011; Sandulescu et al. 2015), arguing that spamming text will be different in the writing and language styles. Text-based approaches have mainly adopted classical supervised learning utilising manually engineered features such as lexical, semantic, syntactic and psycho-linguistic features extracted from texts (Shojaee et al. 2013; Ott et al. 2011; Fei et al. 2013). The other approach mines the reviewer-product-target network attempting to discover collusiveness among singleton reviewers targeting specific products to detect suspicious singleton reviewers (Kumar et al. 2018).

Supervised learning and the utilisation of labelled data is a plausible solution, however, gaining access to larger number of diverse annotated spam datasets specifically for neural-based approaches- is very difficult. It is both unpractical and very expensive

[1] https://www.brightlocal.com/learn/local-consumer-review-survey/.

for humans to annotate large amounts of spam reviews due to size and complexity of the data that varies from one domain to another. In this light, focusing attention on unsupervised approaches is more appealing, where we can leverage the huge amounts of unannotated reviews to enhance the learning process. To detect singleton spam reviews without annotations, we cast the problem as an anomaly detection problem, where the goal is to identify spams as outliers from the majority of normally behaving observations.

A strong link is noticed between sentiment patterns found in textual contents and non-truthful information. It is found that false information has different emotional patterns from true information; this finding has been practically exploited to detect false news (Ghanem et al. 2020). Moreover, salient linguistic features and patterns of words are found to be correlated with deceptive text, e.g. emphasising certainty (Mihalcea and Strapparava 2009). Psycho-linguistic analysis using LIWC lexicons has confirmed that the higher the number of cognition and affect terms, the higher the likelihood of the text being deceptive (Tausczik and Pennebaker 2010). These patterns cannot be easily conveyed through the analysis of reviews' star ratings (McAuley and Leskovec 2013); in fact, it is found that temporal rating variations are not always strongly correlated with spam attacks, especially in the middle range (three to four stars) (Ye et al. 2016; Ye and Akoglu 2017). Thus, the most discriminative features are suggested to be found in the underlying textual details in reviews.

Our analysis of spam reviews has confirmed that spammers usually target important aspects (or product properties) with sentiments opposing the general consensus. The notion of aspect importance also varies for different products and even from time to time. For example, spammers trying to engender a fake positive review for a mobile phone usually maliciously target important aspects such as battery life, screen size and processor power. However, they provide partial positive opinions on some of these aspects for undeserving products, which can be clearly identified as different from genuine opinions for the same products. To cover their deception, they may comment negatively on less important aspects (such as colour and accessories) to give their reviews a look of authenticity and objectivity. These variations of sentiments towards different aspects present challenges for current state-of-the-art spam detection models.

As a supporting real-life example, we present real-life reviews from our annotated Yelp restaurants dataset for one top restaurant known for its high-quality food. The main aspects reviewed here are food, service and atmosphere. First, we present a truthful review with a five-star rating as a simple baseline in terms of the level of detail, tone and sentiment. Then, we show two spam reviews for the same restaurant: one gave a four-star rating to avoid detection, but still aimed to pull the rating down, while the second gave one star. Both reviews cover more than one aspect; try to spot the differences in the level of detail and the sentiment distribution provided.

– **Truthful review** 'Overall, was it worth the hype? Yes and more. Was it worth every penny? Hell yes! Is it a life changing food experience? Yes! Will I be coming back? You bet! Soon. I'd also like to see a different menu. Truly 5-star experience. I will say without hesitation was the best $1000 I've ever spent. Everything from the food—pure flavors of the ingredients, lamb with 52 different complementary ingredients, the kitchen tour was amazing as well, wine pairing, service (sort of)

some of the staff was not very friendly, atmosphere, was everything you paid for and more'.
- **Spam review** 'In a nutshell, it is a sophisticated place with good food. The only thing I dislike about this restaurant is you need to book room months before. Coming to the food, Forget about the actual taste, but only the presentation manages to convince you psychologically that the food is awesome. Having said that, I would also like to advice them to work on the service'.
- **Spam review** 'Don't understand why we keep going back. Waiter joins our conversation. Ugh, which makes me uncomfortable. Food occasionally tastes good Out of 16 courses, most were OKAY, but mostly this is all about presentation, The dessert was not tasty. Hard to find a well priced wine'

To detect such spam reviews, we want to capture the fine variations in text; hence, reviews should be modelled in a precise, comprehensive and discriminative way to represent these disparities of sentiments on the aspect level. To capture the distributed aspect-level sentiments from text for finer-grained opinion representation, we need to devise a review modelling process that mimics the generation process of opinionated texts (reviews) as a distribution of opinionated aspects.

The quality of products and services may vary noticeably over time, which is automatically reflected in reviews. For example, relating to the previous example of restaurant reviews, on some days the service may be worse due to incompetent staff recently hired, poor ingredients recently used or a range of other reasons. Thus, capturing the changes in opinions towards the different aspects of reviewed entities over time is deemed to be necessary for the precise definition of an evolving genuine opinion space. This will help in detecting suspicious anomalous patterns (in terms of outlier-like opinions) not conforming to the majority of reviews at given points in time.

In this paper, we aim for a totally unsupervised anomaly detection approach for singleton spam detection. Our analysis suggests that spams can be identified by extracting deep features of fine-grained opinions towards aspects from review texts. Regarding this objective, we face several challenges:

- Given the textual contents of reviews, how to accurately model aspects and sentiments simultaneously and in an unsupervised setting?
- Given the learned opinion trend evolution time series, how to detect spam reviews effectively and unsupervised?

Our design objective is to propose a solution to the above problems that is both accurate and practical. First, we aim to learn a comprehensive opinion representation of reviews on the aspect level in a cost-effective and accurate way. Then, we need to learn the latent temporal correlations found in the high-dimensional review representation in the presence of anomalies (spams in the form of noise) to differentiate genuine from malicious reviewing patterns. Therefore, we present our unsupervised anomaly detection approach for singleton spam detection. Towards this goal, we make the following contributions:

- To capture the distributed aspect level sentiments from text for finer-grained opinion representation, we propose to join two deep Boltzmann machine (DBM)

networks to model both aspects and sentiments and their governing relations simultaneously.
– To discover the hidden non-linear correlation of both short-term and long-term patterns of trend evolution in time series, we propose to employ a long short-term memory (LSTM) neural network. LSTM is adaptive to emerging patterns and can help in tracking opinion fluctuations, while effectively learning the normal behaviour in an unsupervised setting.
– To avoid the need to set a predefined error threshold (proposed by domain experts, calculated from small-size annotated datasets or created by setting an assumption on the error distributions), we propose to utilise a novel Robust Variational Autoencoder (RVAE) to detect anomalous observations from opinions' temporal patterns unsupervised. This becomes possible by reconstructing the discovered opinion trends along with expected error distributions, while iteratively isolating spam instances to minimise reconstruction errors.

To the best of our knowledge, none of the existing work in the opinion spam detection literature employs the evolution of temporal aspect-sentiment patterns for spam review detection. Extensive experiments on three real-world opinion spam datasets from Yelp were conducted to quantitatively and qualitatively evaluate the effectiveness of our model in detecting singleton spam reviews. Comparison against state-of-the-art approaches from the literature shows the superiority of our proposed approach in terms of detection accuracy. For simplicity, we will refer to our model throughout the paper as *DTOpS* (Detector of Temporal Opinion Spam). Moreover, we offer our implementation as an open source on Github in support of the research community and to further advance the research boundaries.[2]

This paper is organised as follows: Sect. 2 describes related work to spam detection, document modelling and learning representations respectively. Section 3 clarifies the rationale behind the choices taken towards devising our proposed solution. Section 4 provides the necessary preliminary information needed to be known in advance prior to diving into the details of our proposed model. Section 5, introduces our methodology in tackling the spam detection problem. It is divided into two main parts: Sect. 5.2 introduces our novel joint aspect-sentiment modelling of reviews and Sect. 5.3 describes the spam detection element and how we learn the anomalous behaviour from variations of sentiments towards aspects over time. Section 6 presents our thorough evaluation results and analysis of both components of our solution. Section 7 shares some of the limitations of our proposed work. Finally, Sect. 8 summarises our findings.

2 Related work

Related work tackling our problem of study can be broken down into three main parts. First, studies related to the detection of review spams. Second, we complement the study of spam with introducing other related work from the field of anomaly detection, shedding light on how spam can be similar to anomalies. Finally, studies

[2] https://github.com/yassienshaalan/DTOPS.

related to aspect based sentiment analysis of opinionated text and how review texts can be modelled and represented.

2.1 Opinion spam detection

In recent years, the detection of spam reviews has attracted significant research. According to the source information used to analyse spam reviews, the existing literature can be divided into three broad categories: (1) review textual content, (2) behaviour-based techniques, (3) reviewer network-based techniques.

Text-based approaches focus on text to extract spamming signals. It can be considered one of the most straight forward approaches to detect singleton spams due to the limited focus on a single source of information (i.e. text) to find evidence of malicious interactions. Supervised learning text-based approaches formulate spam review detection as a classification problem and the emphasis is on extracting lexical, sentiment, syntactic and psycho-linguistic features from the text (Shojaee et al. 2013; Ott et al. 2011; Fei et al. 2013). Unsupervised learning approaches (Li et al. 2013) adopt topic modelling techniques (e.g. LDA) to identify differences in topic word distributions of fake reviews versus truthful ones. Textual content similarity using cosine similarity is employed as a spamicity measure, detecting spammers who echo reviews that already exist in the space (Sandulescu et al. 2015). In a similar fashion, Narisawa et al. (2007) introduced three measures to quantify text outlyingness in terms of sub-strings that are alike within the documents. Recently, Saini and Sharan (2017) has touched on using aspects in spam detection, by manually extracting approximately 400 aspect terms, then combining the polarity of the collected aspect terms found in each review with other quantitative and POS features to train an SVM classifier.

Deep neural network models for detecting spam reviews from review texts have recently been investigated on a significant scale (Lai et al. 2015; Zhao et al. 2018; Luyang et al. 2016; Wang et al. 2017) to overcome the limitations of traditional approaches (i.e. building handcrafted features). Sentence-level representations have been investigated by splitting the document into sentences (Luyang et al. 2016), extracting features on the sentence level, then building up collectively to represent the whole document. In an alternative approach, word-level (Lai et al. 2015) features are extracted, building up to sentence and then to document level. Although the majority of deep models perform well, they require significant effort to manually annotate datasets and enormous time for training.

Reviewer behaviour analysis approaches (Lim et al. 2008; Savage et al. 2015; Mukherjee et al. 2013; Fei et al. 2013; Xie et al. 2012; Ye et al. 2016; Huayi et al. 2017; Mukherjee and Liu 2010), on the other hand, study interaction-centric statistical metrics and devise a scoring function to measure spamicity based on abnormal reviewing patterns. Signals used for score computation include rating deviation, early rating deviation, groups multiple high ratings or multiple low ratings and number of reviews given per day. One solution to aggregate these scores is the use of linear weighted combination functions (Lim et al. 2008; Savage et al. 2015). After a threshold has been set by experts, if the chosen signal value exceeds that threshold, then a suspicious behaviour is detected (Mukherjee and Liu 2010; Huayi et al. 2017). However, the

mean and standard deviation of these scores for singleton reviewers are meaningless as they are computed per reviewer.

Time series analysis of user behaviour is also proposed, where the focus is on identifying attacks from the behavioural temporal patterns (Mukherjee et al. 2013; Huayi et al. 2017). The aim is to find distributional divergence between the latent population distributions of spammers and non-spammers in different time series. In addition, (Fei et al. 2013) expand on the study of the burstiness nature of reviews at certain time windows to identify review spam based on the assumption that the same bursts tend to have the same nature. This means that the signals are either mostly from spammers or mostly from genuine reviewers. Recently, the hidden Markov model *(HMM)* has been proposed to detect spammers' co-bursting behaviour by identifying spammers who work in groups (Huayi et al. 2017). On a similar front, several approaches are proposed (Xie et al. 2012; Ye et al. 2016) that do not detect individual reviews as being spam or genuine, but predict the time when an entity is most likely to be a victim of a spam attack. Behavioural analysis cannot be applied to detect singleton spams due to the lack of potential behaviour trails.

Network-based approaches are introduced (Akoglu et al. 2013; Rayana and Akoglu 2015; Ye and Akoglu 2017; Shehnepoor et al. 2019) to detect opinion spam by analysing the links between reviewer, review and product to detect opinion spamming groups. In practice, all these approaches assume that a reviewer in a group writes more than one fake review for multiple products and in turn the detection process is based on the explicit reviewer-product association (in terms of direction and density) to compute trustworthiness scores. Yet, not suitable for singleton spam reviews. It has been found that spamming behaviour is not an individual-reviewer-based activity, but rather a group event (Mukherjee et al. 2011; Liu and Zhang 2012). Multiple spammers may collaboratively carry out an attack at a given time to cause maximum harm. Moreover, it is found that a group of reviewers are not necessarily a group of different real individuals, but in fact one or a few spammers disguised under different identities (Kumar et al. 2018).

To summarise existing research on singleton spam detection, studies on text-based approaches require extensive feature engineering and do not address the inherent complexity of the subtle differences of the aspect-level variations in reviews, supervised approaches are impractical and cannot be easily extended to other types of customer reviews while behavioural and network-based approaches are not applicable due to the lack of spammer footprints.

2.2 Anomaly detection

Anomaly detection is another relevant, widely explored, line of research that supports many real-life applications (Srivastava et al. 2008; Garcia-Teodoro et al. 2016; Solberg and Lahti 2005; Christy et al. 2015; Kazemi et al. 2016). In data mining and analysis, differentiating between normal data points and those that are dissimilar (often called anomalous data points) is a common practice in understanding the underlying data for any given problem (Wang et al. 2019; Ahmed et al. 2016). Research in this area can be categorised into three groups according to the anomaly type. Point anomalies is the

type where a data point is considered anomalous if it is dissimilar to the rest of the data points (Eskin et al. 2002; Liu et al. 2010). Contextual or conditional anomalies is another type of anomaly, where data instances can be considered anomalous only in a certain context, which is usually defined by the problem structure and domain. Research on contextual (conditional) anomalies are commonly explored in time series data (Phua et al. 2004; Srivastava et al. 2008; Salvador et al. 2004; Kou et al. 2006). The third type of anomalies are collective anomalies, where a collection of related data points is considered anomalous with respect to the entire data (Zheng et al. 2015; Bontemps et al. 2016). In this case, individual data points may not be considered anomalous in isolation, but the co-occurrence of a number of data points at the same time form an anomaly.

Supervised anomaly detection involves training a supervised classifier, using labelled data with normal or anomalous target classes (Görnitz et al. 2013; Ma et al. 2016). Deep supervised learning approaches can be considered most accurate where they learn the separation boundary (normal versus anomalous) when annotated data is available (Görnitz et al. 2013; Kim et al. 2015; Erfani et al. 2017); Deep semi-supervised techniques assume the knowledge of one class and build the decision boundary accordingly (Aygun and Yavuz 2017; Estiri and Murphy 2019; Akcay et al. 2018; Principi et al. 2017; Racah et al. 2017). Unsupervised deep learning anomaly detection techniques assume that the majority of data come from the normal class and the separation can take place using the learned hierarchical complex latent feature. Autoencoders are the most dominant technique in this area (Baldi 2012; Sakurada and Yairi 2014; Zong et al. 2018; Xu et al. 2018). Its model of operation is based on the assumption that with sufficient training samples, the normal instances produce low reconstruction errors compared to outlying instances.

2.3 Aspect based sentiment analysis

Mining (determining and classifying) opinions from textual reviews towards aspects has gained extensive attention in recent years. SemEval (Pontiki et al. 2014, 2015, 2016) has adopted the task of aspect-based sentiment analysis (ABSA) and provided a few aspect-level annotated datasets. Related work tackling this problem with aspect and sentiment analysis can be categorised into three lines of research: topic modelling-based (aspect and sentiment mixture modelling), text span extraction (for aspects and/or sentiments terms) and document representation learning approaches.

Topic modelling, the discovery of the hidden semantic structures in text, has long been widely used in analysing text. Latent Dirichlet allocation (LDA) (Blei et al. 2003), a generative statistical model, is fundamental in this area, and was proposed to discover the mixture of topics describing documents. Later, (Titov and McDonald 2009) introduced a customised LDA to build a multi-grain topic model for online review text. Similarly, probabilistic mixture models (Mei et al. 2007; Titov and McDonald 2008; Moghaddam and Ester 2011; Jo and Oh 2011; Sauper and Barzilay 2013) have been proposed to learn word distribution through the use of a mixture of multinomials to categorise words in documents as words about topics and, later, words representing positive or negative opinions. To enrich the generative process of modelling, (Zhao

et al. 2010a; Mukherjee and Liu 2010; Card et al. 2018; Gupta et al. 2019) incorporate supplementary information such as ratings and sentiment labels to steer the modelling towards better topic discovery. Informed priors from POS and grammatical rules are also proposed (Zhao et al. 2010a; García-Pablos et al. 2018) to help the LDA topic modelling process to better differentiate between aspects, sentiments and background words. On another front, DBMs were employed in document modelling where topics are defined as the discovered latent variables in the last hidden layer (Srivastava et al. 2013).

Text-span extraction of aspects and sentiments, on the other hand, is defined as the task of extracting text snippets that capture aspects and/or sentiments. Aspects are extracted using multiple *human-crafted rules* based on observations and previous linguistic knowledge, such as grammatical rules, syntactic rules (Zhao et al. 2010b; Qiu et al. 2009; Zhang et al. 2010), semantic rules (Kobayashi et al. 2006; Hu et al. 2004a) and frequency-based rules (Hai et al. 2011; Hu et al. 2004b). Supervised learning-based methods have been proposed, formulating the problem as a classification problem (Jakob and Gurevych 2010; Zhuang et al. 2006; Toprak et al. 2010). Sentiment analysis techniques on document, sentence and aspect levels using dictionary-based methods have also been suggested (Hu et al. 2004b; Zhu et al. 2009).

Unsupervised learning methods have been investigated to address some of the limitations of supervised methods, attempting to incorporate as many external non-annotated data sources, relations and rules as possible to increase the power of their operation. Sentiment-specific word embedding (Maas et al. 2011; Tang et al. 2015; Fu et al. 2018) has been employed to improve downstream sentiment analysis tasks. To extract aspects and sentiments simultaneously, (Wang et al. 2015) proposes a variation of an RBM, where the hidden units contain heterogeneous nodes to capture aspects, sentiments and background separately. Although this model performs better than the LDA-based variations presented previously, it has a few drawbacks: (1) it is dependent on many forms of prior (POS, sentiment lexicons and the need to run an LDA iteration to obtain initial aspect distributions); (2) it is not ideal for multi-aspect and multi-sentiment documents as it only extracts one aspect and the sentiment towards this aspect per document; and (3) multi-grained aspect-sentiment modelling is too complex to be solved with a one-hidden-layer RBM architecture using one hidden unit per aspect to capture all related terms.

Representation learning is another related line of research. It studies how documents are formed and how the words making up each document are related. One type of representation learning model is the generative model used for inferring aspects and sentiments, which is still closely related to neural models for document representation learning. Neural approaches (Mikolov et al. 2013; Pennington et al. 2014) are proposed to tackle this problem. Different granularities of text are targeted for learning, such as sentence-level embeddings (Logeswaran and Lee 2018), paragraph-level embeddings (Le and Mikolov 2014) and document-level embeddings (Chen et al. 2017). Recently, BERT (Devlin et al. 2019), a very powerful general-purpose language model that applies bidirectional training of transformers, a popular attention mechanism, has been used to extract pretrained contextual embeddings representing documents. The pretrained representation can also be fine-tuned towards many

downstream tasks, including sentiment classification, next-sentence prediction and named-entity recognition.

To summarise related work to representing text, for topic modelling-based approaches, the inference of the posterior distribution over topics is an approximate rather than exact inference, which may reduce the accuracy and quality of discovered topics. Mixture models-based approaches cannot exploit the co-occurrence of topics to yield high probability predictions for words that are different from the distributions predicted for individual topics. Moreover, most studies on document modelling and document representation focus on modelling topics, while sentiments are considered as supplemental information that influences the distribution of topics; sentiments are not distinguished as an explicit latent variable. Aspect extraction techniques either require expensive term-level annotated data or a variety of sets of informed priors, and yet cannot be easily generalised to other domains. Document representation-based approaches often encode very limited information of the distributions of the fine-grained aspect-sentiments, which suggests that they are not the best choice for aspect-sentiment analysis.

3 Motivation

In this section, we discuss the challenges for detecting spam reviews to motivate our proposed solution.

To achieve spam detection that does not require supervision, we need to first learn a representation of opinions that allows us to more easily identify spam, and then learn the trends and fluctuations of these learnt opinion presentation over time in order to identify anomalous spams out of the truthful temporal opinion patterns.

The representation of opinionated text has not been extensively studied in the literature, despite the development of general language models for text representations (Peters et al. 2018; Devlin et al. 2019). Existing text representation techniques do not take into account the document structure as a distribution of opinionated aspects, while the use of sentiments or ratings as supplements in document modelling for deeper sentiment analysis cannot decompose the rating scores to the individual aspect level (Card et al. 2018; Wang et al. 2015). To learn an accurate opinion representation of reviews that takes into account of the topic and sentiment dimensions of documents in a cost-effective way, we take a DBM approach. Our choice of adopting DBMs specifically rather than traditional document modelling techniques and other neural networks (e.g. RBMs and Autoencoders) is supported by the following considerations:

- DBM has more extensive power in discovering deep complex nonlinear representations of the input than other topic modelling techniques (Srivastava et al. 2013; Wang et al. 2015; Xie et al. 2015).
- The inference process of DBMs using Markov chain Monte Carlo (MCMC) is more accurate than the variational inference process used in other solutions (Card et al. 2018), which is computationally more efficient but less accurate.

- DBM was found to be very successful in decoding the document generation process as a distribution of topics (Srivastava et al. 2013). This supports their candidacy for decoding the sentiment dimension as well.
- DBM can effectively account for missing and noisy data by allowing top-down feedback in addition to the regular bottom-up pass.
- DBM can be pretrained in advance, leveraging huge sets of unlabelled data, and can later be fine-tuned towards the specific task in hand, which saves a lot of computation time.

To capture and represent both dimensions of the opinion, we propose a network of two DBMs, one DBM for aspect modelling and one for sentiment modelling. The DBMs are connected by a bridging layer to learn the distribution of sentiments towards aspects and vice versa (details are in Sect. 5.2).

To distinguish the truthful reviewing patterns from the suspicious ones, we need to analyse the trend evolution and fluctuations of opinions along the time dimension. For this purpose, we adopt LSTMs rather than traditional time series analysis methods (e.g., ARMA and ARIMA) and other neural networks (e.g., RNNs and Bayesian networks) for the following reasons:

- LSTM is known for its extensive power in modelling temporal information which allows for high accuracy of future opinion predictions.
- LSTM can effectively handle our complex high-dimensional temporal sequences and discover hidden correlations. LSTM utilises the context-aware weighted self-loop for each node, which allows it to forget older information (lossy memory) while accumulating new information.
- LSTM allows for sharing a single set of parameters over different time steps, which gives more flexibility than fitting a different set of parameters for each time step.
- LSTM can overcome minor levels of noise (or anomalies) in training data effectively while still capturing the overall trend.

LSTMs were adopted previously to detect anomalies on its own (Malhotra et al. 2015; Hundman et al. 2018) under two main assumptions: (1) LSTM needs to be trained on anomaly-free data to correctly capture the normal behaviour. (2) The error threshold (of how far predicted values from expected ones can be) needs to be known in advance in order to identify anomalous instances from the trend during prediction time.

The challenge then becomes how to use LSTMs to identify anomalies in the temporal abstract space under a fully unsupervised setting. In a representation learning context, anomalies cannot easily be compressed properly, and thus cannot easily be retrieved from the latent space with acceptable reconstruction error, similar to non-anomalous (or normal) data points. Thus, an effective representation learning method that can automatically isolate anomalies in the abstract space becomes the final piece in our desired solution. We propose to apply RVAE to identify spam as anomalies in the temporal abstract space learnt by LSTM. (1) We first use RVAE to clean up the training set so that the LSTM can learn the truthful trend of opinions accurately; (2) At prediction time, unsupervised, it can iteratively identify suspected spamming instances from the predicted opinion trend (details are in Sect. 5.3). We employ the robust principal component analysis methodology (Candès et al. 2011) to VAEs rather than the traditional Autoencoders of Zhou and Paffenroth (2017) for a few reasons:

– Instead of learning an arbitrary function (used in Autoencoders) to reconstruct the input data, VAEs learn the probability distribution that can actually model the data.
– Autoencoders do not allow for easy interpolation as the latent space where their encoded vectors lie may not always be continuous.
– Modelling the latent variables as Gaussians allows each dimension in the representation to push itself as far as possible from the other factors to attain more precise control over the generated latent representations. This also allows for easier sampling from the continuous latent space.
– In learning opinion variations, we do not want to replicate the review representation as is, but to have an acceptable variation along the different dimensions of the opinion. In fact, this imitates real-life scenarios where reviews from different reviewers can be similar without being exact duplicates of each other.

4 Preliminaries

In this section, we introduce the foundations of constructing the main building blocks of our solution depicted in Sects. 5.1, 5.2 and 5.3. More specifically, we explore deep representation learning neural networks operating on sequences of data. The main constructs discussed below include Deep Boltzmann Machine (DBM), Autoencoder (AE), Variational Autoencoder (VAE) and Long Short-Term Memory (LSTM) networks. We aim at giving the reader an overview of what each module is used for and its expected model of operation. Further details of the specific intended-use cases and model of operation of each of these modules will be discussed later in the designated sections.

The *DBM* is a type of generative model that is described as being an energy-based undirected graphical probabilistic model that follows the Markov random field (Salakhutdinov and Hinton 2009). DBMs are known for their capability for learning complex representations from the input and have been employed in many tasks, e.g. image recognition (He et al. 2013) and document modelling (Srivastava et al. 2013). A DBM comprises one visible layer $x \in \{0, 1\}^D$ (for input), where D is the number of input units, and multiple layers (two or more) of hidden units that are often referred to as hidden layers, e.g. $\{h_1, h_2\}$ where $h_1 \in \{0, 1\}^{F_1}$ and $h_2 \in \{0, 1\}^{F_2}$, where F_1 and F_2 are number of units in each layer respectively. The probability assigned to input vector x is:

$$\begin{aligned} P(x;\theta) &= \sum_{h_1,h_2} P(x, h_1, h_2 : \theta) \\ &= \frac{1}{Z(\theta)} \sum_{h_1,h_2} e^{(\sum_{i,j} W_{ij} x_i h_j^1 + \sum_{j,l} W_{jl} h_j^1 h_l^2)} \end{aligned} \tag{1}$$

where the model parameters $\theta = \{W^1, W^2\}$ represent the parameters on both the visible-to-hidden and hidden-to-hidden layer interactions, and Z is a partition function, which is a special case of a normalising constant in probability theory. No connection is allowed between units of the same layer but are rather between units of neighbouring

layers. The DBM can be understood as a series of restricted Boltzmann machines (RBMs) stacked on top of each other. For instance, a two-layer DBM can be thought of as two RBMs (each containing one visible and one hidden layer) stacked on top of each other. The training of the original Boltzmann machine was slow because its approximation to the likelihood of the gradient employed a random Markov chain for initialisation. To make the learning process efficient, a greedy layer-wise or layer-by-layer pretraining using contrastive divergence is applied. After the initial pretraining step, the DBM is fine-tuned using the back-propagation algorithm.

The DBM is a generative model, where the inference takes place by computing the posterior probability. However, the calculation of the exact posterior probability for the data distribution $P(x;\theta)$ is intractable, which means an approximation is needed; we denote this by $Q(x;\theta)$. $Q(x;\theta)$ is often referred to as the variational or mean field approximation. The optimisation objective for DBMs is to minimise the Kullback–Leibler (KL) divergence between the actual data distribution and the approximated data distribution. Due the powerful generative capabilities of DBMs in discovering latent relations of complex data distributions while operating in a totally unsupervised setting they are found to be very successful in decoding the document generation process. Thus, the DBM is found to be very efficient in the area of document modelling (Srivastava et al. 2013; Xie et al. 2015). Even RBMs—the building blocks of DBM—have shown great success when employed specifically to extract aspects and sentiments of opinionated texts (Wang et al. 2015; Nguyen-Hoang et al. 2016).

The *Autoencoder* is a form of unsupervised feed-forward neural network that is trained to compress and encode data and be able to reconstruct it from the reduced encoded representation with minimal error. The Autoencoder model of operation can be characterised by three main properties. It is data specific, which means it is able to only encode similar data to what it has been trained on (e.g. if trained on images of faces it may perform poorly while encoding images of cars). It is lossy, which means the reconstructed data is a degraded version of the original version. It requires no labels for training, but rather is self-supervised, using the input itself as a target. The learning objective is not as simple as learning the identity mapping, which is disallowed automatically due to the introduction of hidden layers, adding sparsity constraints on the activity of the hidden representations and adding regularisation.

The two main applications of Autoencoders are dimensionality reduction for representation purposes and data denoising. This is typically reached by learning a lower dimension representation (encoding) for data by training the network to ignore noise. Generally, to build an Autoencoder, we need to build three elements: an encoder that converts data into a smaller and dense representation, a decoder that maps data from latent space back to original input and a distance function (or loss function) that optimises the learning objective (minimising loss between original and reconstructed data). Usually, the loss is implemented by employing functions such as mean-squared error or cross-entropy between the output and input. The mapping process can be understood as follows:

$$\tilde{X} = D(E(X)) \tag{2}$$

where X is the original data, which is first encoded using encoder **E** (from input space to latent space) and later decoded using decoder **D** (from latent space to original data space) to $\tilde{X}$, the reconstructed version of the data. The optimisation function can be formulated as the minimisation to the reconstruction difference (**X**-$\tilde{\mathbf{X}}$) as:

$$\min_{D,E} ||X - D(E(X))|| \tag{3}$$

This can usually be optimised using stochastic gradient descent algorithms to learn the hidden layer parameters of both E and D with respect to minimising the distance function. Traditionally, both the decoder and encoder contain more than one layer (deep Autoencoder) to increase the model's ability to learn more useful representations found in complex data distributions.

The *VAE* is a very interesting variation of the Autoencoder. Instead of leaving the network to freely learn an arbitrary mapping function, the encoding side is constrained to learn the parameters of a probability distribution that can actually model the input data. It then becomes possible to sample points from the learned distribution that generates new data. The encoder of the VAE squashes input X into two parameters μ and σ in a latent space. Typically, these are two vectors, one for the means μ and one for standard deviations σ, both of size n (desired size of latent space layer). Thus, the ith element in vectors of μ and σ represents the mean and standard deviation of the ith random variable. Then, we can randomly sample an encoded point X_i from the latent normal distribution that is assumed to generate the data.

One of the great benefits of VAEs is the allowed degree of local variation due to the random sampling, where the same input is not always strictly limited to the same encoding, as in Autoencoders. This is because the encoding of the input is centred around the mean, while the standard deviation controls how far from the mean is allowed. Thus, the ideal case is to have encodings for similar entities as close as possible to each other, while still being distinct, to allow for smooth interpolation. At the same time, we still want a degree of overlap between samples that are not very similar, in order to interpolate between points from different classes. The decoder on the other end maps sampled points from the latent space back to the original input data. Since the decoder is exposed to a range of variations of the encoding of the same input during training, it learns that not only a given point in the latent space comes from a given class, but that nearby points that slightly vary also belong to the same space.

The parameters of the VAE model are trained via two loss functions: (1) a reconstruction loss (similar to the Autoencoder) forcing the reconstructed samples to match the input, and (2) KL divergence between the learned latent distribution and the prior distribution. Minimising the KL divergence means optimising the probability distribution parameters (μ and σ) to properly fit the target distribution as follows:

$$\sum_{i=1}^{N} \sigma_i^2 + \mu_i^2 - \log(\sigma_i) - 1 \tag{4}$$

The total loss is the summation of all the KL divergences of all components of X_i drawn from $N(\mu_i, \sigma_i^2)$ in X. Thus, theoretically, the minimum loss value can be reached when μ_i equals 0 and σ_i equals 1. Interestingly, the generative power of VAEs have recently been employed in detecting anomalies (Sun et al. 2018).

An *LSTM* network is a type of neural network that is capable of learning sequential dependencies in an ordered list of inputs, where the context of each element in the sequence is required to understand the whole picture (Hochreiter and Schmidhuber 1997). Originally, LSTMs were proposed to overcome two main drawbacks of recurrent neural networks (RNNs), namely the vanishing gradient and exploding gradient problems, which both lead to ineffective network training and consequently bad performance (Graves and Schmidhuber 2005). LSTMs have been successfully applied to many tasks, including language modelling on character and word levels (Sundermeyer et al. 2012; Verwimp et al. 2017), machine translation (sequence-to-sequence learning) (Sutskever et al. 2014) and image processing (Byeon et al. 2015).

In our context, we are more interested in the time series trend prediction capabilities of LSTM. This comes mainly from the fact that they can store sequential information for an arbitrary period of time and can tolerate random fluctuations (noise) in the input compared to expected output, to a certain extent. LSTMs allow for capturing latent correlations (especially non-linear) between long-term and short-term status over time. This is done by balancing the weights between the come-in status and the ongoing historical status. As a result, LSTM-trained models are more adaptive to emerging patterns along the time series. Even vanilla LSTMs are sufficient for some problems where performance is acceptably fair (Greff et al. 2016). The most performance-critical tuneable hyper-parameters for LSTMs are the learning rate and network size.

LSTMs can be used to model trends in time series that are both *univariate* (where only one variable varies over time) and *multivariate* (where multiple variables vary over time). Multivariate time series—the focus of this work—can take one of two forms: multiple-input series, where the input comes from more than one parallel input time series, with one output variable that is dependent on all the input time series; and multiple-parallel series, where there are multiple parallel time series, and one value needs to be predicted for each. The expected input sample to a model is defined by the number of time steps and the number of features. The output can be thought of as a two-dimensional vector of the number of samples and the number of output series. The latter form of multivariate models is the adopted variation in our proposed model.

To define the training instances of an LSTM model, we need to define two main variables. First, the time step which is equivalent to the historical time interval that an LSTM model uses to predict future observations; for example, for a network to consume ten data points it is set to 10. Second, the number of features which describes the input variables used in learning. In our problem definition, this will be the number of features (m-dimensional) of the extracted reviews' aspect-sentiment representation.

Time series prediction is mostly done in a supervised learning setting. In order to formulate the dataset into the required expected format, a few (n) lag observations are used as input (X); next, as the required forecast, the output is observation y. Then, all the observations are shifted down by one time step, moving forward. Thus, at current time (t), previous observations ($t-1, \ldots, t-n$) are used to make forecasts to future times ($t+1$), or ($t+1, \ldots, t+k$) in the case of multi-time-step prediction. Having

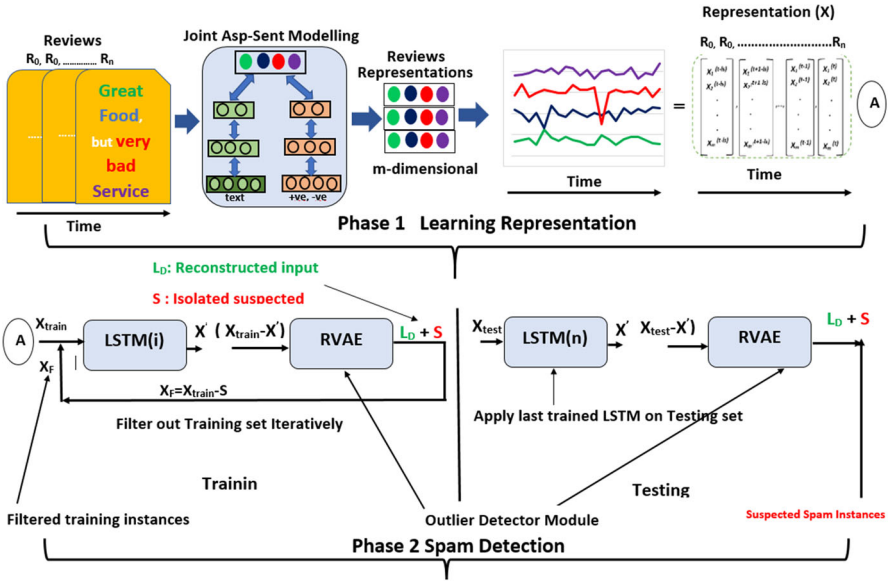


Fig. 1 DTOpS model operation pipeline

prepared the target signal to be predicted (y), we can now set our loss function. The most common loss function used when compiling LSTM time series prediction models is the mean squared loss ($\frac{1}{m}\sum_{i=1}^{m}(y_i - \tilde{y}_i)^2$).

5 Methodology

In this section, we explain our proposed methodology for detecting opinion spam. We start by giving a brief overview of the two main phases that constitute our solution. Then, we move on to explaining the full details of each phase, and giving step-by-step directions for clear execution.

5.1 DTOpS model overview

In Fig. 1, we present the full pipeline of our proposed model architecture. Our solution comprises two main phases applied in series. The first phase is to learn a descriptive representation that can highlight the subtle variations in opinions. The input to this phase is shown on the top left of the figure as the time-stamped textual reviews for reviewed entities. Opinion representation is learned through modelling the textual contents of reviews by analysing topical and sentiment words and how they are linked together to make up opinions. The analysis is designed to capture the main topics of the text (usually referred to as aspects or features of an entity), the main sentiments expressed towards reviewed entities and latent relationships governing both aspects and sentiments. To reach this goal, we join two DBM networks, one analysing

topics in the text and the second analysing word polarities. The expected output of our representation is an m-dimensional vector representing the discovered latent topics and sentiments expressed in each review as presented in the third box of phase 1 in Fig. 1. The full details of joint aspect-sentiment representation learning are depicted in Sect. 5.2. A time series is then constructed from the discovered review representation (sorted by time-stamp) in preparation for the next step of our model presented by the fourth box of phase 1 in Fig. 1.

The second phase represents the actual spam detection process. It comprises two main sub-modules, an LSTM module and a RVAE module represented as square boxes in phase 2 of Fig. 1. The LSTM module is used to analyse opinion trend evolution. This is obtained by building a time series from the extracted m-dimensional aspect-sentiment review representation over the timeline. Then, with the knowledge of historical opinions, we try to predict future opinion observations (in terms of the joint review representation). This allow us to create an understanding of what the trend evolution of opinions is like and how it is expected to be in the future by analysing the latent temporal correlations between the different dimensions of the representation.

To compare the trend for suspicious fluctuations, we input the original and predicted opinion trends to our RVAE module to identify spamming instances. The input X to RVAE is the aspect-sentiment extracted representation along with the time series aspect-sentiment predicted representation (LSTM output). The VAE tries to understand the underlying structure behind the review representation, the predicted representation and the difference (error in all dimensions) by transforming the input data X into two matrices ($L_D + S$). First, L_D represents the reconstructed noise-free version of the input data. Second, S represents the isolated noise (or outliers) on both the feature level (column-wise in the input matrix) and the instance level (row-wise in the input matrix). This process occurs in a totally unsupervised setting via training the VAE and identifying suspected instances. Then, noise is iteratively removed from reconstructed data L_D and the VAE is retrained. This allows the Autoencoder's hidden layers to learn a representation that is closer to the true representation of the clean (truthful) data without the distracting effect of outliers. As a result, we can highlight the spamming instances automatically with no need for spam annotated labels.

The reviews space may become mixed with spam reviews along the way, which can prevent the LSTM model from capturing the normal reviewing behaviour accurately. This is why we use RVAE twice, as shown in Fig. 1: first to clean up the original training set so that our LSTM model trains on spam-free data for maximum benefit and second to finally identify spam reviews in the unseen data, given the original representation (aspect-sentiment representation) and the predicted representation (LSTM-predicted aspect-sentiment representation) and without access to spam labels.

5.2 Learning aspect sentiment document representation phase

To effectively analyse reviews, we first need to transform them in a descriptive space from which it becomes easier to capture the variations of opinions towards the different aspects. Thus, our objective here is to learn a representation of documents that encodes the fine-grained aspects and sentiments simultaneously. Aspects are the latent repre-

sentation of topics of documents. In our case topics are inferred from the observed word distributions in the given corpus, thus the goals is to uncover these latent variables. Words are likewise assigned to topics that describes them. Latent representations of documents are expected to capture the hidden semantics of the underlying documents.

To generate more coherent topics, we aim to model the sentiments towards aspects (topics) that precisely reflects the opinions generation process by modelling both aspects and sentiments as separate latent variables. The generation process of opinionated documents can be described as a distribution of opinionated aspects which depicts the coupling between aspects and sentiments. Thus, implicitly there exists two sub problems to solve. One is to model aspects and sentiments simultaneously, and the second is to model the interaction between aspects and sentiments in the review generation process. To achieve this goal, we leverage two different document input representations, one to model aspects and one to model sentiments, and then pair the learned knowledge in an additional step. This formulation is like merging two different points of views represented by different modalities for the same document (review) and present one collective holistic view encompassing both sides.

Aspects in reviews resemble topics in document modelling. Therefore, for the aspect modelling sub-problem, the aim is to infer the latent topics by building a graphical generative probabilistic model using words as input to represent the reviews. For the sentiment modelling sub-problem, the aim is to infer the latent sentiments of documents from a bag of polarities generated from sentiment lexicons for words in the document. The initial process of learning the latent representations of aspects and sentiments separately can be thought of as a pre-training step to the joint modelling step. However, our objective is not to solve these two sub-problems in isolation from one another, but rather to jointly optimised the modelling of aspects and sentiments simultaneously. There is an intuitive form of collaboration between the two tasks, where the knowledge of one complements the learning for the other. This can be done by sharing the learning process parameters after learning the initial latent aspects and sentiments in the abstract space and iterate until convergence to a plausible configuration. The input representation received by each sub-network can be thought of as a likelihood or soft label for each other, which provides a complementary step to the learning process. For example, we would like our model to correlate the occurrence of the phrase "good food" with the positive sentiment and represent them jointly, so that the model assigns high probability to one conditioned on the other.

5.2.1 Overview of joint document modelling

To achieve our design objectives, we choose to employ two DBMs to model both latent aspects and sentiments of documents. We place one layer on top of the two DBMs to mediate the coupling information of the two sources of knowledge between the two sides. This layer will retain the final learned joint aspect-sentiment representation for the given documents. In Fig. 2, we present the design of our proposed joint aspect-sentiment model. It comprises two paths, the green path on the left receives the textual representation of reviews, while the orange one on the right receives the sentiment representation of reviews. The 'W's represent the weight matrices of learned parameters and 'h's represent the stochastic hidden variables containing the learned

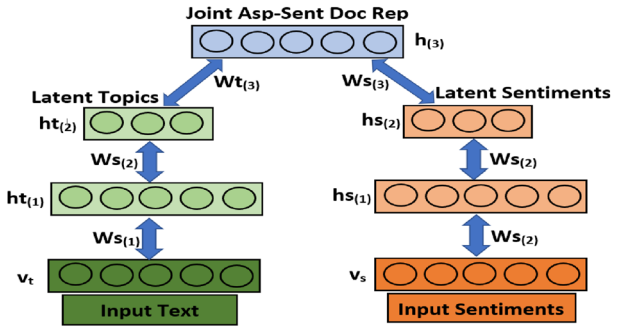


Fig. 2 Joint aspect-sentiment model architecture (Color figure online)

latent topics and sentiments. For simplicity, we will refer to our opinion representation model throughout the paper as 'JASM'.

Our model receives input through the two visible (input) layers; one layer receives textual input $\{v_t\}$ as a vector of words, while the second layer $\{v_s\}$ receives input as a vector of polarities representing word sentiments. The network is then stacked with five hidden layers. Two hidden layers $\{h_{t_1}, h_{t_2}\}$ are responsible for inferring latent aspects, and two hidden layers $\{h_{s_1}, h_{s_2}\}$ are responsible for inferring the latent sentiments of documents. The top hidden layer $\{h_3\}$ passes the knowledge learned between the two sub-networks and will hold the final joint aspect-sentiment representation. The number of hidden units in each of the hidden layers is a design choice that is different from one problem to another and it can be empirically verified as will be shown in Sect. 6.2.3. However, the number of hidden units in higher layers $\{h_{s_2}, h_{s_2}\}$ should be smaller than that of lower layers $\{h_{s_1}, h_{s_1}\}$, forcing the network to learn meaningful latent aspects and sentiments. Layer $\{h_3\}$ is pre-set with the desired number of aspect-sentiments to represent documents and is usually higher than or equal to the combination of $\{h_{s_2}\}$ and $\{h_{s_2}\}$. Since our model is unsupervised, only the number of topics, not the topics themselves, are prespecified. This way, the conditional probability for each word and sentiment polarity is learned on an abstract level and in turn summarises the joint aspect-sentiment representation.

Each DBM is a stack of two layers of RBMs, building a generative stochastic undirected graphical model learning a probability distribution from the input word representation. Two-hidden-layer DBM is empirically found to be sufficient to balance the trade-off between computation time and acceptable level of accuracy.

For simplicity, we use the BOW representation of documents as input to the aspect DBM. We believe that our model can discover deep complex representation of aspect distributions from this simple input representation. Another benefit of our modelling structure is that it allows for the incorporation of other forms of input word representations, as will be shown in the Experiments section. The input to the sentiment DBM is a vector of polarity scores extracted from a sentiment lexicon (e.g. NLTK Vader lexicon) for words of input text. We represent each word in the sentiment vector by three values (positivity score, negativity score and neutrality score). This helps in capturing aspect words from the sentiment side as they are represented by neutral sentiments

as opposed to other opinionated words. Additionally, not all neutral words have zero positivity or negativity scores, some of them having quite important non-zero scores for these polarities (Colhon et al. 2017). The three polarity scores for each word are then multiplied by the number of occurrences of each word to represent the distribution of polarity intensity as well.

5.2.2 JASM training process

In this section, we describe the process followed to train the aspect and sentiment sub-networks along with the joint layer. For the aspect sub-network, the input vector is represented by v_t, where $v_t \in \mathbb{N}^K$, and v_{t_k} refers to the frequency of word k in a document with vocabulary of size K. We assign the following probability to the input vector v_t, taking into account the two hidden layers $\{h_{t_1}, h_{t_2}\}$ responsible for learning latent aspects:

$$\begin{aligned} P(v_t; \theta_t) &= \sum_{h_{t_1}, h_{t_2}} P(v_t, h_{t_1}, h_{t_2} : \theta_t) \\ &= \frac{1}{Z(\theta_t)} \sum_{h_{t_1}, h_{t_2}} e^{(-E(v_t, h_{t_1}, h_{t_2}))} \end{aligned} \tag{5}$$

$$\begin{aligned} E(v_t, h_{t_1}, h_{t_2} : \theta_t) = &\sum_{k=1}^{K} \sum_{j=1}^{F_1} W_{kj}^{t_1} h_j^{t_1} v_k \\ &+ \sum_{i=1}^{F_1} \sum_{j=1}^{F_2} W_{kj}^{t_2} h_j^{t_1} h_i^{t_2} + \sum_{k=1}^{K} v_k b_k \\ &+ M \sum_{j=1}^{F} b_j^{t_1} h_j^{t_1} + \sum_{i=1}^{F} b_i^{t_2} h_i^{t_2} \end{aligned} \tag{6}$$

In Eq. 6, M refers to the number of words per document and F refers to $\{F_1, F_2\}$ which are the numbers of hidden units in layers h_{t_1} and h_{t_2}, respectively, representing discovered latent topics. The number of units of $\{F_1, F_2\}$ is a design choice based on domain knowledge (level of detail reviewed for each product category) and experimentation can help in choosing a reasonable number; this needs to be specified before training starts. In general, the objective of energy-based models is to reach a plausible configuration by minimising a predefined energy function in Eq. 6.

Similarly, for the sentiment sub-network we assign probability to the sentiment polarity vector $v_s \in \mathbb{R}^D$ of the document of size D using a Gaussian distribution:

$$P(v_s; \theta_s) = \frac{1}{Z(\theta_s)} \sum_{h_{s_1}, h_{s_2}} e^{(-E(v_s, h_{s_1}, h_{s_2}))} \tag{7}$$

and the energy function can be represented as follows:

$$E(v_s, h_{s_1}, h_{s_2} : \theta_s) = \sum_{i=1}^{D}\sum_{j=1}^{F_1} W_{ij}^{s_1} h_j^{s_1} \frac{v_{s_i}}{\sigma_i} + \sum_{l=1}^{F_1}\sum_{j=1}^{F_2} W_{lj}^{s_2} h_j^{s_1} h_l^{s_2} + \sum_{i=1}^{D} \frac{(v_{i_s} - b_{i_s})^2}{2 * \sigma^2} + \sum_{j=1}^{F_1} b_j^{s_1} h_j^{s_1} + \sum_{l=1}^{F_2} b_l^{s_2} h_l^{s_2} \tag{8}$$

where σ represents the model parameters similar to θ_t.

One advantage of our architecture is that it is open to the utilisation of different input forms of word-level representations such as BOW, TF–IDF and pretrained word embeddings. If TF–IDF or trained word embedding are chosen as input for the aspect network where the values are real numbers, then we need to assign the input probability in 5 to the energy function of Eq. 8 instead of Eq. 6. This is because Eq. 6 is built for input of integer values representing word counts over the vocabulary.

Finally, the joint distribution over both text and sentiment inputs can be represented by adding the top connecting hidden layer h_3, retaining the final learned joint aspect-sentiment representation as:

$$P(v_t, v_s : \theta) = \sum_{h_{t_2}, h_{s_2}, h_3} P(h_{s_2}, h_{t_2}, h_3) \left(\sum_{h_{t_1}} P(v_t, h_{t_1} | h_{t_2})\right) \left(\sum_{h_{s_1}} P(v_s, h_{s_1} | h_{s_2})\right) \tag{9}$$

where the second and third terms come from Eqs. 5 and 7, respectively, while the first term can be calculated as follows:

$$P(h_{s_2}, h_{t_2}, h_3) = \sum_{d=1}^{F}\sum_{n=1}^{F} W_{dh}^{t_3} h_d^{t_2} h_n^3 + \sum_{d=1}^{F}\sum_{n=1}^{F} W_{dh}^{s_3} h_d^{s_2} h_n^3 + \sum_{d=1}^{F} b_d^3 h_d^3 \tag{10}$$

The objective of the model is to maximise the expected log-likelihood probability over the training samples. However, the exact maximum log-likelihood for this model is intractable. Approximate learning procedures can be adopted to compute the log-likelihood. Mean field inference is proposed to estimate the data-dependent expectation parameters using MCMC approximation, as in Salakhutdinov and Hinton (2009). The approximation is done by constructing a Markov chain with the desired distribution; after a few steps we can obtain a sample of the required distribution by observing

the chain. Ultimately, the procedure aims at approximating the true posterior for both pathways $P(h \mid v; \theta)$ as $Q(h \mid v; \mu)$, with fully factorised approximated distribution over the five hidden units $\{h_{t_1}, h_{s_1}, h_{t_2}, h_{s_2}, h_3\}$ as:

$$Q(h|v_t; \mu) = \left(\prod_{j=1}^{F_1} q(h_j^{s_1}|v) \prod_{l=1}^{F_2} q(h_l^{s_2}|v)\right) \left(\prod_{i=1}^{F_3} q(h_i^{t_1}|v) \prod_{l=1}^{F_4} q(h_l^{t_2}|v)\right) \left(\prod_{k=1}^{F_5} q(h_k^{3}|v)\right) \tag{11}$$

where v $\in \{v_t, v_s\}$, $\{F_1, F_2, F_3, F_4, F_5\}$ are the prespecified numbers of hidden units in each layer of $\{h_{s_1}, h_{s_2}, h_{t_1}, h_{t_2}, h_3\}$, respectively, and μ (field parameters) $\in \{\mu_{s_1}, \mu_{s_2}, \mu_{t_1}, \mu_{t_2}, \mu_3\}$ represent the parameters of each of the hidden layers.

The learning algorithm iterates until it finds values for μ that maximise the variational lower bound of the model parameters of θ using the MCMC-based stochastic approximation. To make sure that the model parameters start in a good position (sweet spot) and to minimise the training time, we perform a pretraining step by initialising the parameters. This is achieved by running a few iteration steps for each individual RBM making up the DBMs of both pathways and then using these parameters in the full model training steps. Finally, after estimating the approximate posterior $P(h \mid v; \theta)$ of the last hidden layer h_3, the hidden units of this layer represent the final joint aspect-sentiment representing the given documents.

5.3 Spam detection phase

In this section, we describe our approach of utilising the extracted fine-grained aspect-sentiment representations for effective spam detection. First, we describe our proposed formulation of the problem as anomaly detection, along with the persisting challenges, then we present our detailed solution.

We suggest mining the temporal correlations found in the variations of sentiments presented towards aspects of entities for precise opinion analysis in relation to spam detection. Incorporating the temporal patterns of the high-dimensional aspect-sentiment representation of the reviews allows for the construction of a natural multi-variate time series, which suggests using LSTM.

The problem of spam detection can now be cast as an anomaly detection problem, where the goal is to identify spams as outliers from the majority of normally behaving observations. In pursuit of this goal, we suggest adopting deep learning-based methods to effectively detect the spam anomalies. Deep learning approaches have recently been proven to outperform traditional machine learning approaches in anomaly detection. Among the reasons for this are their inherit flexibility in learning nested hierarchical concepts from the data incrementally from each layer of the neural network (Wang

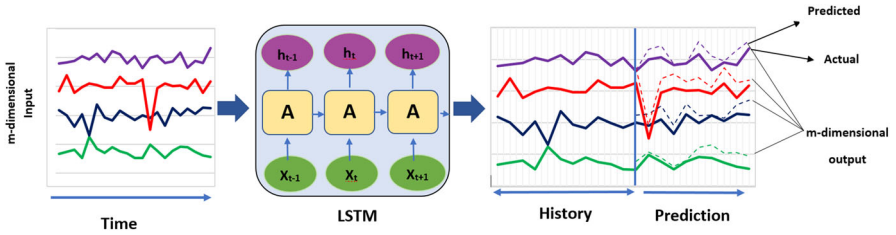


Fig. 3 Multivariate LSTM high-level representation in terms of input and output

et al. 2019). Moreover, when the size and complexity of the data space increases, the performance of traditional methods degrades quickly. In our case specifically, they can help us uncover the complex structure of anomalous reviewing patterns. Theoretically, this can be achieved by discovering a low-dimensional hidden embedding space (a compact, more representative form) within the original data space, where normal instances and anomalies can be separated adequately. The original data is then reconstructed from the discovered embedding with minimal error. Higher reconstruction errors become an indication of non-conforming observations to the learned distribution of normal instances.

Our proposed solution to spam detection comprises two main steps applied in series. The first step reformulates the original extracted aspect-sentiment representation to be input to the next step. This involves employing time series analysis to study the notion of how the opinions expressed in reviews are evolving with time. We employ an LSTM network for this purpose. The expected output of this step is two versions of the aspect-sentiment representation: the original representation X and the predicted aspect-sentiment representation $X^{'}$ over the reviewed entity's lifespan. The two input vectors are presented to a VAE, coupled with l_{2_1} regularisation for identifying spamming instances. This takes place by reconstructing X, $X^{'}$ and the difference between them e (error), while iteratively isolating the outlying instances based on how far the reconstructed data are from inputs.

In our problem setting, the input data to the LSTM is review-learned m-dimensional joint aspect-sentiments. Each dimension can be considered a time series on its own; thus, there are multiple parallel time series and a value must be predicted for each. Figure 3 shows a high-level representation of our LSTM module with the m-dimensional inputs and m-dimensional predicted outputs. The solid lines present the original representation, while the dotted lines represent the predicted representation for each dimension. For each time step t, the input vector can be represented as $\{x_1^t, x_2^t, \ldots, x_m^t\}$ where $x^t \in \mathbb{R}^m$. The output vector can be thought of as a two-dimensional vector in the number of samples and the number of multiple output series values (m). The two main parameters responsible for controlling the prediction are, first, l_s, the number of historical previous time steps used in training, and second, l_p, the number of steps ahead to predict for each of the m dimensions. For example, if the network is required to consume ten observations to learn to predict one future observation, l_s is set to 10 and l_p is set to 1. Limiting l_p to smaller values helps reduce the processing time and accounts for subtle changes within the time series, leading to better accuracy.

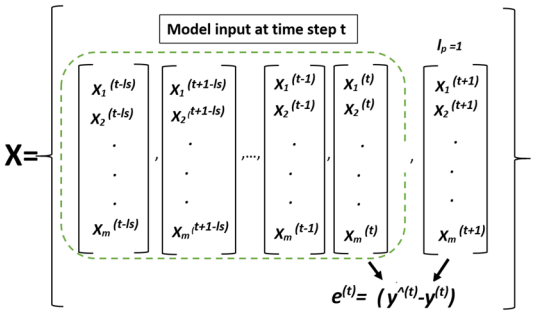


Fig. 4 Sample input to our LSTM model at different time steps visualising the different parameters

Figure 4, visualises a sample input to the LSTM at different time steps presenting the dimensionality of each vector in terms of m and the controlling parameters l_s and l_p as described previously. It also shows how the LSTM learning target is set and how the optimisation function is calculated to minimise the error .

Our problem setting is unsupervised, i.e. no access to human-labelled data is required. However, in our quest to train the LSTM model (which requires target labels), we reformulate the reviews data along the time lime to be used in supervised setting. This can be done by arranging the data in each time step to build a sample data point i of l_s lagging observations as input (X_i) and l_p as the target output observation y_i. Then, all the observations are shifted forward along the timeline by one time step to create another sample, and so on. The prediction vector is then represented by $\hat{y^t}$, while the ground truth vector is represented by y^t. We adopt mean squared loss function to train our LSTM model, so for the m-dimensional time series we need to optimise:

$$\min_{\theta} \frac{1}{m} \sum_{i=1}^{m} (y_i - \tilde{y_i})^2 \tag{12}$$

where θ represents the model parameters.

Given the merits of adopting LSTM for trend prediction analysis, it can also be directly applied to detect anomalies on its own. This is done by calculating the error difference between predicted signals and original observations and using this as an anomaly score. Then, a threshold is set and observations with prediction error values above this are considered anomalies. However, this model of operation of LSTMs is only valid under certain conditions. First, LSTMs need to be trained on nominal data (noise-free) to capture the nominal behaviour, then future non-complying patterns can be easily detected as anomalies. If training data contains noise that come in long sequences, in this case, LSTMs will not be able to capture the normal behaviour precisely. Second, for anomaly detection a threshold must be set; this can be established in several ways. One way is from problem-domain knowledge (rule-based or by expert direction), which is different from one domain to another. Another way is to assume that error distributions fit a Gaussian distribution and then to estimate the anomaly (Malhotra et al. 2015); however, to calculate this threshold the knowledge of some labelled data from the normal class is assumed to be known in advance. Other non-parametric thresholds have also been proposed (Hundman et al. 2018); these still

need to be trained on noise-free data yet are applied on scalar error values and cannot be easily applied to high-dimensional data. Thus, in order to operate in a fully unsupervised manner and without impractical assumptions regarding the input data or any assumed error distributions, an extra step needs to be added on top of the temporal review trend analysis to highlight suspicious error patterns.

That extra layer is our RVAE, which is a combination of VAE (strong generative non-linear data modelling) and robust principal component analysis (noise-detection capabilities). Anomalies cannot easily be compressed and retrieved properly with minimum reconstruction error such as nominal data points. The proposed model works by iteratively removing noise from input data X and isolating data in a separate container S, resulting in filtered data contained in $L_D = X - S$, such that $X - L_D - S = 0$.

In order to reach our objective, the loss function for our system to reconstruct the input needs to be balanced with the l_2 norm to account for non-zero values in S. The optimisation function therefore becomes:

$$\min_{\theta} ||L_D - D_\theta(E_\theta(L_D))||_2 + \lambda||S||_{2_1} \tag{13}$$

where L_D represents a low-dimensional reconstructed input, S contains the isolated noise in the original data, λ is a hyper-parameter to tune sparsity in S, and E_θ and D_θ are the VAE encoding and decoding layers learned through optimising loss functions of Eqs.3 and 4. λ plays an essential role in controlling the rate of detected anomalies and is considered a pivotal factor in tuning how loose or strict the spam detection process is going to be. Larger values restrict the number of data points to be isolated into S to very low level, consequently increasing the reconstruction error for the VAE. However, smaller values of λ drive many points to be isolated as anomalies and in turn diminish the reconstruction error. This hyper-parameter is important in controlling the rate of false positives in detecting spam as we run unsupervised.

RVAE is next trained through back propagation to minimise the reconstruction error shown in Eq. 13. Note that the choice is made to apply l_{21} regularisation as a penalty on S instead of l_1 because in our problem the assumption is that the spam is structured; so, if data is to be viewed as a matrix in terms of number of features and number of samples, we need to identify instances (row-wise) of data as spam where anomalies appear on many features (not necessarily all features), unlike in other problems, where anomalies may be unstructured and scattered element-wise for instances of the data where l_1 can then be suitable.

l_{21} norm on S can then be defined as:

$$||S||_{2_1} = \sum_{i=1}^{n} \left(\sum_{j=1}^{m} |s_{ij}|^2 \right)^{\frac{1}{2}} \tag{14}$$

where n is the number of instances in input data X and m is the number of features for each instance. In order to solve Eq. 13, we need to optimise each of the terms; hence, we follow an iterative optimisation methodology provided in Zhou and Paffenroth (2017). This involves using an alternating optimisation method implemented using

Algorithm 1 Spam Detection

procedure DTOP(X_{tr}, X_{ts}) ▷ 'Training/Testing' joint aspect-sentiment time series representations
  $i \leftarrow 0$ ▷ Iteration index
  $max_i \leftarrow 0$ ▷ Maximum number of iterations
  $c_1 \leftarrow 0$ ▷ Acceptable number of spams detected in training as desired
  $s_1 \leftarrow 0$ ▷ Number of detected spams in training
  $c_2 \leftarrow 0.8 * Length(X_{tr})$ ▷ Acceptable size for training after filtering suspected spams
  $s_2 \leftarrow Length(X\_tr)$ ▷ Size of training set so far
  $X\_tr_t \leftarrow X\_tr$ ▷ Temporary training set
  $ratio \leftarrow 0.7$ ▷ train/test split ratio, e.g. 70/30, preserving time series order
  $X_{tr_f}, X_{ts_f} \leftarrow Split(X\_tr_t, ratio)$ ▷ Filtered training/testing
  $\lambda_{tr} \leftarrow 0.00010$ ▷ Set λ_{tr} training as desired
  while $(i < max_i)$ **do** ▷ Train exit criteria
    $lstm\_model \leftarrow Train\_LSTM(X_{tr_f})$ ▷ Train/update the LSTM model
    $\tilde{X_{ts_f}} \leftarrow lstm\_model.Predict(X\_tr_t)$ ▷ Predict expected trend
    $E \leftarrow (X_{ts_f} - \tilde{X_{ts_f}})$ ▷ Error matrix between original and predicted time series
    $X \leftarrow Prepare\_Input(X_{ts_f}, \tilde{X_{ts_f}}, E)$ ▷ Concatenate original, predicted and difference
    $L_D, S \leftarrow RVAE(X, \lambda_t r)$ ▷ Run RVAE on prepared input
    $s_1 = Count\_Spam(S)$ ▷ Count suspected spam instances
    if $s_1 <= c_1$ **then** ▷ Exit if no more spam detected
  break
    $X\_t r_t \leftarrow Filter\_Spam(S, X\_t r_t)$ ▷ Filter out non-zero instances in S as spam
    $s_2 \leftarrow Length(X\_tr_t)$ ▷ Update new filtered training set length
    if $s_2 < c_2$ **then** ▷ Exit if too many instances removed from training
      break
    $X_{tr_f}, X_{ts_f} \leftarrow Split(X_{tr_t}, ratio)$ ▷ Split filtered-out set, preserving temporal ordering
    $i \leftarrow i + 1$
  ▷ Now run on testing set
  $\tilde{X_t s} \leftarrow lstm\_model.Predict(X\_ts)$ ▷ Use latest trained LSTM on unseen data
  $E \leftarrow (X\_ts - \tilde{X\_ts})$ ▷ Compute difference
  $X \leftarrow Prepare\_Input(X\_ts, \tilde{X\_ts}, E)$ ▷ Prepare input vector data
  $\lambda\_ts \leftarrow 0.00015$ ▷ Set λ testing as desired
  $L_D, S \leftarrow RVAE(X, \lambda_t s)$
  $S \leftarrow Identify\_Spam(S)$ ▷ Identify spam entries in S and return their indices
  return S ▷ Suspected instances

convex optimisation built on the alternating direction method of multipliers (ADMM) (Boyd and Vandenberghe 2004), which uses the alternating projection method to find the projections onto the intersection of convex sets. We can see that our objective is non-convex; hence, the optimisation process is non-trivial. ADMM helps in dividing the objective in Eq. 13 into two parts, then the first is optimised with $L_D = X - S$ while fixing S, then with $S = X - L_D$ while fixing L_D. In fact, the optimisation of $||S||_{21}$ is difficult as it is non-differentiable; it is therefore optimised as a proximal gradient problem—more specifically as a l_{21} norm-minimisation problem similar to Zhou and Paffenroth (2017).

Now we know the purpose of each of the two main building blocks (LSTM and RVAE) of our spam detection model. In Algorithm 1, we give the pseudo-code to show how the two modules interact to serve our purpose. To recap, we want to train an LSTM on a spam-free training set so that we can capture the normal reviewing trend. Thus, when compared with future trends, the difference between predicted and original representations signals the anomalies. So, Algorithm 1, the while loop,

Algorithm 2 RVAE Training

procedure RVAE(X, λ) ▷ Input $X \in R^{n*m}$, n samples & m features
  $L_D \leftarrow X$
  $S \leftarrow 0$ ▷ Matrix of n samples & m features
  $\eta \leftarrow 0.1$ ▷ Exit threshold
  $\beta\_1 \leftarrow 0$ ▷ Exit criteria 1
  $\beta\_2 \leftarrow 0$ ▷ Exit criteria 2
  $k \leftarrow 100$ ▷ Number of iterations allowed
  $i \leftarrow 0$ ▷ Iteration index
  while $(\eta > \beta\_1 \ \& \ \eta > \beta\_2 \ \& \ i < k)$ **do** ▷ Train Exit Criteria
    $L_D \leftarrow X - S$ ▷ Initially $L_D = X$
    $L_D \leftarrow VAE(L_D)$ ▷ Reconstruct L_D using VAE
    $S \leftarrow X - L_D$ ▷ Set S to be the difference
    $S \leftarrow Proximal_{\lambda\_l2_1}(S, \lambda)$ ▷ Optimise S using proximal method
    $L_S \leftarrow L_D + S$ ▷ Reconstruct temporary data
    $\beta\_1 \leftarrow (X - L_D - S)/X$ ▷ Recompute exit threshold 1
    $\beta\_2 \leftarrow (L_S - L_D - S)/X$ ▷ Recompute exit threshold 2
    $i \leftarrow i + 1$
  return L_D, S ▷ Filtered reconstructed data & isolated anomalies

represents the iterative process, where we filter out the training set from suspected spam by our employed RVAE module, presented in Algorithm 2, to identify suspected spam instances; we remove them, then train a new LSTM model on the filtered set.

Algorithm 3 VAE Training

procedure $VAE(X)$
  $epochs \leftarrow 100$ ▷ Number of epochs
  $max_patience \leftarrow 5$ ▷ Number of epochs with no improvement
  $i \leftarrow 0$ ▷ Iteration index
  while $(i < epochs || patience > max\_patience)$ **do**
    $\mu, \sigma \leftarrow Encoding(X)$ ▷ Latent distribution $p(Z|X)$
    $Z \leftarrow Sample(\mu, \sigma)$
    $\tilde{X} \leftarrow Decode(Z)$ ▷ Decoding $p(X|Z)$
    $loss \leftarrow Optimize(||X - \tilde{X}||^2 + KL\_Divergence(N(\mu_x, \sigma_x), N(0, 1))$ ▷ Compute loss
    $patience \leftarrow Update\_Patience()$ ▷ Check for loss improvement
    $i \leftarrow i + 1$
  return $\tilde{X}$ ▷ Reconstructed data

We continue doing that until either no more spam is detected or the process has been rigorous and removed a large portion (controlled variable) of the original training set, which defeats the assumption that anomalies are only a minority class. The outcome from this training process is an optimum LSTM model that is trained on review representations from the genuine class to be used on the testing set. Finally, we apply the same drill on testing data (unseen). We use LSTM to predict the trend, then use the original and predicted representations as input to the RVAE to identify spam instances. Technical details related to implementing the RVAE are described in Sect. 6.1.2.

In Algorithm 2, we show a pseudo-code to the training process of RVAE. It is an iterative process that starts by estimating L_D and trains a VAE (as in Algorithm 3)

to reconstruct that input. Next, S is computed and one step of proximal optimisation performed. Then, L_D and S are recomputed iteratively, until either the maximum number of iterations is consumed or an acceptable conversion rate is reached.

Algorithm 4 Proximal Optimisation

procedure $Proximal\_\lambda l_{21}(S, \lambda)$ ▷ Proximal Optimisation Process
▷ S matrix in n*m
  for $i \leftarrow 1$ to m **do**
    $e_i \leftarrow \sum_{k=1}^{n}(|S_{k_i}|^2)^{\frac{1}{2}}|$
    if $e_i > \lambda$ **then**
      for $j \leftarrow 1$ to n **do**
        $S_{j_i} \leftarrow S_{j_i} - \lambda * (\frac{S_{j_i}}{e_i})$
      $continue$
    for $j \leftarrow 1$ to n **do**
      $S_{j_i} \leftarrow 0$
  return s ▷

In Algorithm 4, we show the proximal optimisation process is performed in pseudo-code. It is an iterative process where elements are combined into blocks turning element-wise sparsity to block-wise sparsity.

6 Experiments

In this section, we present our evaluation methodology and experimental results comprising two parts. First, we evaluate our proposed spam detection model against literature's strong baselines covering different research directions. Second, we evaluate our proposed aspect-sentiment modelling in learning our novel joint representation. This takes place by evaluating the success of the learned representation in performing different tasks (e.g. aspect and sentiment classification), along with the clarifying visualisations.

6.1 Spam detection evaluation

In this section, we describe our datasets, evaluation measures, baselines and report the experimental results for evaluating the spam detection component of our solution.

6.1.1 Datasets and baselines

Experiments to test our model for spam detection were conducted on three publicly available online review datasets widely used in the opinion spam literature: YelpChi, YelpNYC, and YelpZIP (Rayana and Akoglu 2015). YelpChi and YelpNYC datasets contain reviews for restaurants in Chicago and New York City. On the other hand, YelpZIP dataset starts with a ZIP code in New York State and incrementally collects reviews for restaurants in that ZIP code moving across a continuous region of the U.S.

map, including NJ, VT, CT, and PA creating a huge dataset. Datasets contain review metadata, e.g. star rating and date, as well as review text. Statistics describing the three datasets are given in Table 1. It can be clearly seen that all datasets contain a high percentage of singleton reviews (from 65.35 to 70.55%), ideally suiting them to testing our model.

The Yelp datasets have individual ground truth labels for each review (spam or truthful) based on a fake/suspicious filtering algorithm used at Yelp.com. Although the Yelp anti-fraud filter is not 100% perfect, it is found to produce accurate results on average, and has been used as ground truth for evaluating similar opinion spam detection tasks (Rayana and Akoglu 2015; Kumar et al. 2018).

We compare our model against eight state-of-the-art baseline approaches based on the use of only review contents as features; thus, network-based and behavioural approaches are not applicable to our study. Out of the baselines, three are unsupervised anomaly detection-based approaches, one is an unsupervised text-based spam detection approach and four are supervised approaches, three of which are deep learning-based models.

1. Anomaly detection baselines:
   (a) **Robust Autoencoder (RAE)** (Zhou and Paffenroth 2017): A state-of-the-art deep Autoencoder algorithm inspired by robust principal component analysis that can eliminate outliers and noise without access to any clean training data. We ran with learning rate 0.15, λ 0.001 and four layers, as suggested in the paper (Zhou and Paffenroth 2017).
   (b) **Robust Variational Autoencoder (RVAE)**: Our new, modified version of RAE, built with a VAE to fulfill the need for generative modelling to better suit spam review detection. To reach the reported performance scores of RVAE, the following λ values were used on YelpChi, YelpNYC and YelpZip datasets 5.5×10^{-5}, 8.0×10^{-6} and 2.7×10^{-5}, respectively.
   (c) **Isolation Forest (IF)** (Liu et al. 2008): An advanced anomaly detection model. It explicitly isolates anomalies instead of profiling normal points, which exploits sub-sampling and runs in linear time with a low memory requirement. It is favoured especially with high-dimensional large datasets. We ran with 100 trees and the suggested fraction of outliers of 10%.
2. Unsupervised spam detection baseline:
   (a) **Semantic Similarity (SS)** (Sandulescu et al. 2015): An approach that detects singleton spam reviews by computing the semantic similarity among pairs of reviews using a cosine similarity measure.
3. Supervised spam detection baselines:
   (a) **Ott** (Ott et al. 2011): A supervised classification model (SVM based) built with a comprehensive set of 80 psycholinguistic features extracted by LIWC from review text, coupled with bi-gram features.
   (b) **SCNN** (Luyang et al. 2016): A deep learning supervised approach based on a sentence-weighted neural network that detects spam reviews by learning the document-level representation of reviews.

Table 1 Yelp customer review datasets statistics

Dataset	# Reviewers	#Reviews	# Restaurants	#Reviews per reviewer	#Reviews per restaurant	#Singleton reviewers(%)	#Spam reviews
YelpChi	38,063	61,541	201	1.77	335.30	70.55	8919
YelpNYC	160,225	359,052	923	2.25	389.00	66.15	36,885
YelpZIP	260,277	608,598	5044	2.34	120.66	65.35	80,466

(c) **RNN** (Lai et al. 2015): A deep learning supervised approach based on an RNN. It works by analysing the text word by word and uses a fixed-sized hidden layer to store the hidden semantics of all previous text.
(d) **BERT** (Devlin et al. 2019): A cutting-edge language model based on context-aware deep bidirectional encoder representations from transformers; we fine-tuned it towards spam classification task.

We use four well-known evaluation metrics for performance evaluation: precision, recall, F1 score, and a receiver operating characteristic (ROC) curve where the points on the curve are obtained by varying the anomaly detection threshold.

6.1.2 Implementation details

In this section, we list all necessary technical (implementation) details of the main components for easier reproduction of our full model. We built our implementation of LSTM, VAE and RVAE using Python 3.7 and TensorFlow version 1.13.1. All other required libraries and their versions are listed on our shared code repository. We ran our experiments on an Ubuntu 18.04 LTS Linux machine with four CPUs and 64 GB of memory.

First, we present how our LSTM model was put together. We employed the deep learning Python library Keras[3] to construct our LSTM. We set the main parameters of l_s to 5 and l_p to 1, while the input dimension was set to 3072 units, which is the previously extracted joint aspect-sentiment representation. A hidden layer of 500 units was placed on top of the input layer, along with a dropout layer with rate of 0.3. Then, for deeper latent correlation analysis, we added another hidden layer of size 500 on top of the previous layer, with 0.3 dropout rate. A final dense layer with linear activation was added on top of this layer, with size of (l_p*input_size), which in our case resembles input size as l_p was set to 1. Consequently, we predict a full review one step ahead of time. The model was trained with training batch size of 32, with 15% of training size used for validation while employing mean squared error (MSE) as a loss function and using the Adam optimiser as the optimisation algorithm. The model was trained for 100 epochs with an early-stopping option if validation loss stopped improving for five iterations, for faster training.

Second, we present the VAE details. The first layer of the Autoencoder is expected to be the data input size, as this will be decided later by RVAE input. The encoder of the VAE comprises three hidden layers. On top of the input layer, we added hidden layers, the first of 500 units and the second of 200 units with rectified linear unit (ReLU) function for layer activation; the final hidden layer is a dense layer of the hidden units, expected to hold μ and σ vectors for estimating the data distribution and sampling for decoding. The decoder is similar to the encoder in terms of the number of layers, but with two main differences. First, the layer structure is inverted, starting with the encoder's last layer as the decoder's first and building its way back up to the original input size. The second difference is that the layer activation is set with sigmoid activation function. In training, we used a batch size of 133 and learning rate

[3] https://keras.io/.

Table 2 Performance evaluation (precision, recall and F1 score [%]) of $DTOpS$ versus unsupervised approaches

Dataset	Metric	DTOpS	RVAE	R-AE	IF	SS
YelpChi	Precision	**50.10**	40.90	23.00	32.00	1.23
	Recall	**82.70**	70.70	51.00	44.10	10.20
	F1	**62.40**	53.60	32.00	37.10	1.80
YelpNYC	Precision	**56.49**	43.40	27.00	20.30	5.21
	Recall	**69.30**	68.10	55.00	48.10	28.17
	F1	**62.20**	53.00	36.00	28.50	8.00
YelpZip	Precision	**66.50**	55.80	31.00	19.00	5.25
	Recall	**82.24**	76.12	59.00	47.00	13.74
	F1	**73.60**	64.44	41.00	27.00	7.00

Bold values indicate the highest value with statistical significance

of 0.15 and for optimisation we used the Adam optimiser, with MSE loss function for 100 epochs and 20% of training size used for validation.

Third, we present the necessary details to rebuild RVAE. The expected input size is 9216 (3*input of LSTM), as we input the original aspect-sentiment representation number, along with LSTM-predicted representation and the error or the difference between the two vectors on all dimensions. The number of outer iterations is 10, while the number of inner iterations to run VAE is 100, as described earlier. The λ values input to RVAE in the iterative training process of our spam detection model in Algorithm 1 for datasets YelpChi, YelpNYC and YelpZip are 3.5×10^{-4}, 7×10^{-4} and 1×10^{-4}, respectively. These are initial values to start training; they increase by 10% on each iteration to isolate fewer instances because the training set shrinks after filtering out suspected instances. The λ values used for RVAE on the testing sets for detecting spam for our experiments on the same Yelp datasets are 7.5×10^{-4}, 2.4×10^{-5} and 1×10^{-4}, respectively.

6.1.3 Results and discussion

Quantitative analysis Tables 2 and 3 summarise the average precision and recall of the singleton spam detection along with F1 scores for unsupervised- and supervised-based baselines. The highest average precision/recall values are shown in bold.

$DTOpS$ is shown to outperform all baselines on precision, recall and F1 scores in Table 2, with a statistically significant difference. The semantic similarity model SS (Sandulescu et al. 2015) obtained the poorest precision and recall scores. The main reason behind the poor performance is that SS is only capable of finding duplicate or near-duplicate text spams; however, in practice, spammers disguise themselves very carefully by providing well-rewritten text in different variations, but they still deviate from the norm, and that is how our model can capture them. On the other hand, anomaly detection methods such as IF (Liu et al. 2008) demonstrate better performance and indeed confirm our hypothesis that the problem can be treated as an anomaly detection problem; our assumption that 'singleton spam is minimal and different' still holds.

Table 3 Performance evaluation (precision, recall and F1 score [%]) of semi-supervised $DTOpS^{+}$ versus supervised approaches

Dataset	Metric	DTOpS^{+}	SCNN	RNN	BERT	Ott
YelpChi	Precision	52.30	55.00	**59.0**	58.00	16.20
	Recall	**94.00**	84.13	28.15	74.00	49.80
	F1	**67.20**	67.00	38.00	65.00	24.20
YelpNYC	Precision	58.00	**65.00**	55.00	58.00	16.24
	Recall	**75.00**	64.12	62.20	64.00	72.81
	F1	**65.00**	62.00	58.00	61.00	26.60
YelpZip	Precision	**76.70**	53.30	45.45	**59.0**	22.63
	Recall	**84.80**	85.30	55.50	83.00	71.61
	F1	**80.50**	67.00	45.00	**69.00**	32.00

Bold values indicate the highest value with statistical significance

RAE (Zhou and Paffenroth 2017) finds a low-dimensional hidden embedding using robust PCA within the original data (our joint representation, which is given as input), where normal instances and anomalies can be separated adequately, as opposed to in the IF (Liu et al. 2008) model, which focuses on finding anomalies. RAE is proven to perform consistently better than IF, discovering hidden correlations in the latent space, but is inferior to *DTOpS*, even though both use our joint representation (without, however, incorporating the temporal information). Hence, incorporating temporal information with review text coupled with the aspect-sentiment representation results in a huge performance improvement.

Another observation to note is that recall is always higher than precision because the data is quite unbalanced, with positives (spams) as the minority class. There are many negative examples that could easily become false positives. Conversely, there are fewer positive examples that could become false negatives.

In Table 3, we compare our model to supervised learning spam detection models. One advantage of our model is that it is unsupervised, i.e. it inherently requires no expensive human annotation for supervision. However, for a fair comparison with these models, we train $DTOpS^{+}$ on truthful (spam-free) data only in order to capture normal behaviour (in a semi-supervised setting), unlike the original version $DTOpS$ shown in Table 2 that was trained on noisy data (with spam and had to automatically clean itself) and compared to unsupervised approaches. The results show that our model performs better than Ott that was originally trained on balanced datasets (equal spam/non-spam); however, in our case the spam ratio is only $\approx 13\%$, which seems insufficient to fully cover the different abstract suspicious text styles, especially in the isolated context in which that text is written. Also, the text in our dataset is much longer in length, adding a limitation to the huge time needed to run such a model; the performance on our dataset is therefore clearly degraded. It can also be seen that our model performs better than RNN (Lai et al. 2015).

It should be noted that since Lai et al. (2015) and Luyang et al. (2016) has extensive GPU power and memory requirements, we could not train it on the whole dataset, but it was trained on a considerable sample of the data. The models in Table 3 were trained

Table 4 Detected singleton spam reviews accuracy

Dataset	True +Ve (%)	False –Ve (%)
YelpChi	85.0	8.0
YelpNYC	80.0	10
YelpZip	91.5	6.0

in a stratified way (i.e. keeping the same class balance as in the original set) on 12,000, 20,000 and 40,000 reviews from the three datasets YelpChi, YelpNYC, and YelpZIP, respectively. Comparing the training time of these models, on average on the three datasets $DTOpS^{+}$ was seven times faster than SCNN, five times faster than both RNN and BERT (Devlin et al. 2019) and three times faster than Ott. Moreover, as was expected, BERT consistently performed better than RNN (Lai et al. 2015) and in some cases better than SCNN (Luyang et al. 2016) because of its context-awareness of words on both directions of text in contrast to these context-free techniques. Our model's accuracy is very comparable to that of both SCNN and BERT on average and even exceeds their performance on YelpNYC and YelpZip, although it is semi-supervised (only seeing truthful reviews). Based on the results in Tables 2 and 3, we can conclude that the temporal analysis of evolving opinions is shown to be very effective. This aligns with our observation that spammers' (textual) opinions may resemble genuine ones on an abstract level, and that it is the timing of an opinion diverging (on some of the aspects) from the majority that is the key to identification.

The AUC scores recorded from our model on YelpChi, YelpNYC and YelpZip are 0.82, 0.74 and 0.88, respectively, at different thresholds on the ROC curve. The optimal threshold is computed based on the best ROC value on the curve at the point where the maximum true positive rate occurs with the minimum false positive rate.

In Tables 2 and 3, we showed the accuracy of our model in detecting all spam reviews found in our dataset irrespective of their type (singleton or not). However, in Table 4, we show the accuracy of our model in detecting singleton spam reviews specifically out of all spam reviews detected in the testing set. We can see that our model was able to detect 85.5% of singleton spam reviews out of all singleton spam cases that exist in our dataset on average. It also shows a very low miss rate; moreover, it was able to detect non-singleton spams as well (total spam detected minus singleton spams as shown previously). This proves how effective our proposed text-based spam detection model in uncovering spam reviews irrespective of their source.

We conducted another set of experiments visualising the prediction performance of anomaly detection along the timeline of restaurants examples.

Qualitative analysis In this section, we perform a qualitative analysis to present more evidence supporting our approach. Figure 5 shows the predicted reconstruction error results for one randomly picked restaurant in the Chicago area along the time dimension. The x-axis represents the incremental timeline of when reviews are written, while the y-axis represents the reconstruction error specified by our model. The idea is to look at the predicted reconstruction errors along the timeline and pick the time window where the highest reconstruction error is recorded and analyse these reviews in detail. The chosen time window is highlighted in the figure by the two vertical lines. The

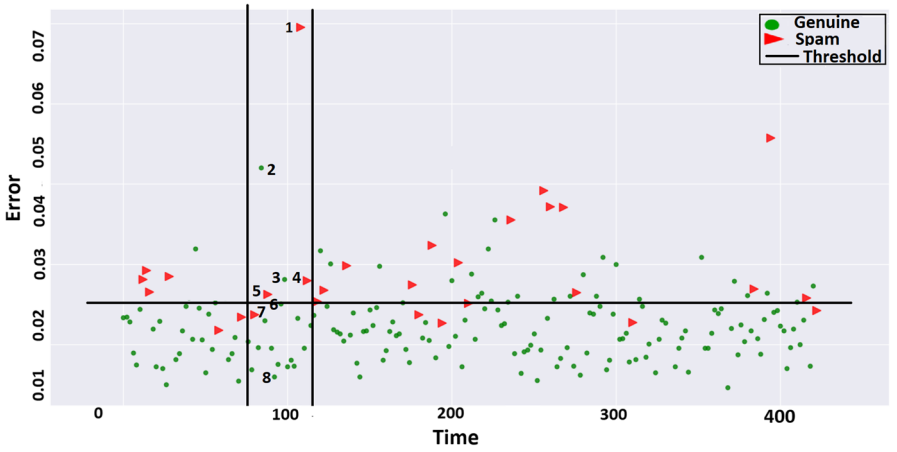


Fig. 5 Prediction reconstruction error versus time: YelpChi example

number of reviews in this time window is 20; 8 reviews are selected for analysis, 5 above the threshold and 3 below it. The reviews are given IDs from 1 to 8 indicating highest to lowest reconstruction errors (i.e. most suspected to least suspected).

In Table 5, we provide a summary of the metrics that are used to conduct the discussion. The reviews in the table are chronologically ordered from top to bottom by time of creation, oldest to most recent.

The eight reviews have one of four prediction outcomes: true positive (tp), true negative (tn), false positive (fp) and false negative (fn), as noted in the table. For each review, we track six metrics: rating, review length (size), aspect distribution percentage (of aspect terms to total length) in reviews, and aspect sentiments, represented by $\uparrow$ for positive sentiments, $\downarrow$ for negative sentiments, and $\longleftrightarrow$ for neutral. We also show a fragment of the review text. The metric values of each review can be compared to the average values of the current window (labelled as CWA) and to the values of the previous window average (lablled as PWA). Out of the 20 reviews in the chosen window, our model predicted 5 reviews as spam, 3 as (tp), 2 as (fp), 1 as (fn) and 14 as (tn), resulting in 60% precision and 75% recall, which aligns with the overall performance evaluation shown in Tables 3 and 2.

Out of the 20 reviews, 14 are from singleton reviewers and only 3 are spams. Our model was able to predict two out of three spams correctly. In fact, methods targeting singleton reviewers would certainly suspect the time window (burstiness of singletons), but only 3 out of 14 singleton reviewers here are spammers, which may result in many false positives. Also, only one out of eight reviews in Table 5 rated the restaurant with a low rating of 2 (genuine reviewer), while six gave it 5 and one gave it 4; the average rating is 4.65 in the current window and 4.53 in the previous window. The average review size in the current and previous windows is high, while spammers write mostly moderate-size reviews. These incorporate many opinionated aspects; in the current window it is a promotion spam. We can see reviews with IDs 1 and 4 mentioning four aspects, while review 5 mentions three aspects. This does not align with the norm of aspects being reviewed in either previous or current windows. If we look at the reviews

Table 5 Metrics used in qualitative analysis for detected spam reviews

Id (class)	Rating	Size	Food	Service	Ambience	Price	Snippet
PWA	4.53	408	60%+	20%↑	15%↑	5%⟷	
CWA	4.65	355	65%↑	20%↑	10%↑	5%⟷	
7 (fn)	4	62	40%↑	⟷	⟷	40%↑	Best lamb, better values, $55 per persons
5 (tp)	5	101	30%↑	5%↑	40%↑	⟷	Great time, food and service excellent, Ambience nice, bad duck breast
8 (tn)	5	62	60%↑	⟷	30%↑	⟷	Bar is awesome, mood right, Love the drinks great place variety of choices
2 (fp)	2	500	10%⟷	70%↓	10%↓	5%↓	Corporate event, stupid bar staff,bill wrong, wine OK, ignored us, flipped out
6 (tn)	5	600	60%↑	20%↑	10%↑	⟷	Hip decor/ambience (5), very good food (4.5), great service (4.5)
3 (fp)	5	496	65%↑	10%↓	10%↑	5%↑	Service slow slow, drinks good, food phenomenal money worth, great lounge comfort
1 (tp)	5	689	65%↑	10%↑	10%↑	10%↑	Disappointed in yelpers, great place,new theme,cheaper price satisfied, staff pleasant, great things
4 (tp)	5	83	40%↑	30%↑	10%↑	10%↑	By far best places, drinks exceptional, staff memorable, like royalty, service on top, worth the price

in chronological order, reviews 7, 5 and 8 are almost similar in terms of size and aspect sentiment distributions, in contrast to the previous window showing less suspicious behaviour, while reviews 2, 3, 4 and 1 show fairly odd behaviour compared to their previous behaviour. The false negative review 7 seems less suspicious because it has short text, lower number of aspects reviewed and fewer strong sentiments and is thus able to blend in.

Tracking the false positives, review 2 (with rating 2) was falsely predicted as spam while being genuine: it was very negative on service and ambience with enormous amount of text, while those aspects were previously moderately positive. Tracking down the actual review revealed that it related to an individual incident of a corporate event that went wrong.

Review 3 was around the norm with regard to food, ambience and price, but gave a suspicious level of detail and was thus suspected. In conclusion, we can see that normal reviewing patterns are temporally consistent.

6.2 Joint aspect sentiment modelling evaluation

In order to perform a thorough evaluation of our aspect-sentiment model, we employ two types of tests. First, we employ intrinsic evaluation, evaluating the learned aspect-sentiment representation on capturing documents' subtle differences and similarities regarding both aspects and sentiments. This will be evaluated using the triplet similarity approach, discussed later. We also present another intrinsic evaluation methodology to our learned representation, supported by visualisations using t-distributed stochastic neighbour embedding (t-SNE) (van der Maaten and Hinton 2008). This is a tool to visualise high-dimensional data by applying non-linear transformations to cluster similar data points. Second, we perform extrinsic evaluation, testing the effectiveness of JASM in performing supervised aspect and sentiment classification tasks.

6.2.1 Datasets and baselines

In our experiments, we test our model on the following four review datasets:

- **SemEval14** (Pontiki et al. 2014) and **SemEval16** (Pontiki et al. 2016) datasets; each consists of 3,000 customer reviews of restaurants. Data were annotated towards the ABSA task, each review being annotated with one aspect (Food, Service, Ambience, Price or Anecdotes/miscellaneous) and the sentiment (positive, negative, neutral or conflict) towards it.
- **City Search** (Ganu et al. 2009) dataset, comprising over 200,000 customer reviews of restaurants collected from 50,000 restaurants in the New York area. Only 3000 reviews were manually annotated, with aspect labels of Food, Staff, Ambience, Price, Anecdotes and Miscellaneous, and with no sentiment annotation. Thus, we manually annotated reviews belonging to three aspects—Food, Staff and Ambience—to evaluate our model. Three annotators were asked to annotate each review with one sentiment of the previously mentioned four choices to be consistent with SemEval (Pontiki et al. 2014). We employed the kappa coefficient (K) (Siegel and Castellan 1988) to measure the pairwise agreement among the

annotators. The inter-annotator agreements of annotations were found to be consistently high, i.e. for the Food aspect, K = 0.750; for Ambience, K = 0.729; and for Staff, K = 0.714.

– **BeerAdvocate** (McAuley et al. 2012) is a dataset that contains 1.5 million customer reviews of beer. However, only a subset of around 9000 reviews were annotated with one of five aspect labels: Feel, Look, Smell, Taste and Overall. Since this dataset is only aspect-annotated, it will only be used for the evaluation of aspect modelling.

In our evaluation of the document modelling component, we tested our performance accuracy in representing and classifying both aspects and sentiments against five baselines covering related lines of research.

– **JASM**: Our model with input of word count to the text sub-network and lexicon sentiment polarity to the sentiment sub-network.
– **JASM$_{\mathbf{B}}$**: A variation of JASM using BERT's pretrained word representation as input to the text sub-network and the same input to the sentiment sub-network, as in the original JASM.
– **Neural Topic Model (NeuralDoc)** (Card et al. 2018): A neural document modelling approach based on the variational inference of Autoencoders that uses sentiments as supplementary covariates.
– **BERT**(Devlin et al. 2019): A state-of-the-art language model based on bidirectional transformer architecture.
– **Aspect classification specific baselines**:
  – **DBMDoc** Srivastava et al. (2013): A two-layer DBM used in document modelling, where the input documents are represented by word counts over the corpus vocabulary.
– **Sentiment classification specific baselines**:
  – **Sent-LSTM** (Miedema 2018): A word-based LSTM neural model for sentiment classification.
  – **IAN** (Ma et al. 2017): A neural interactive attention model for aspect-level sentiment classification.

6.2.2 Experiment setup

In the aspect classification problem, we adopted a binary classification setting such as Food/Not Food, Service/Not Service, i.e. not a multi-classification setting. This is mainly because in the annotated dataset, each review is assigned to only one aspect. Moreover, there is a large class imbalance towards a small number of aspects, e.g. Food and Service comprise 70% of all reviews.

In the extrinsic evaluation section where we evaluate the tasks of aspect classification and sentiment classification, we use the evaluation metrics of precision, recall and F1 score. Precision is the ratio of (tp)/(tp + fp), where tp is the number of true positives and fp the number of false positives; it answers the question of what proportion of predicted positives are truly positive. Recall is the ratio (tp)/(tp + fn), where tp is the

number of true positives and fn the number of false negatives; it answers the question of what proportion of actual positives are correctly classified. When measuring how well the model is performing, it is often useful to have a single number to describe its performance. That number should reflect the two metrics of precision and recall. The F1 score metric serves this purpose; it can be interpreted as a weighted average of precision and recall.

6.2.3 Implementation details

In this section, we list all necessary technical (implementation) details of JASM's main components. We implemented our model and ran our experiments on Red Hat 7.5, Python 3.5, CUDA 8.0 and TensorFlow GPU version 1.11.0. The machine utilised has an Intel(R) Xeon(R) CPU with 56 cores and two NIVIDIA GPUs, each of 16 GB memory. All other required libraries and their versions are listed on our shared code repository. During JASM's training, 4 GB of GPU memory, 20 GB of main memory and 30 GB of disk space needs to be free to avoid out of memory errors.

Post data preprocessing, for SemEval datasets, the input to the text DBM is 12,762 units, with 23,964 units input to the sentiment DBM. The first hidden layer of both text and sentiment DBMs $\{h_{t_1}, h_{s_1}\}$ has 2,048 hidden units, while the second hidden layer for both $\{h_{t_2}, h_{s_2}\}$ has 1024 hidden units. The final joint hidden layer on top of the two DBMs has 2048 hidden units. For Yelp datasets (used for spam detection when BERT representation was used), the input to the text DBM is 3072 units, with 59,088 units input to the sentiment DBM. The first hidden layer of the text DBM h_{t_1} has 2048 units, while for sentiment DBM h_{s_1} has 4096 hidden units. The second hidden layer of the text DBM h_{t_2} has 1024 hidden units, while the second hidden layer h_{s_2} of the sentiment DBM has 2048 units. The final joint hidden layer on top of the two DBMs comprises 3072 units.

The hyper-parameters for the DBM pretraining that is applied to the individual RMBs making up each DMB are as follows: the activation function is set to linear, base epsilon = 0.01, epsilon decay half-life = 10,000, initial momentum = 0.5, final momentum = 0.5, momentum change steps = 10,000, Gibbs step = 1, l_2 decay = 0.001, sparsity target = 0.2, sparsity damping = 0.9, start step up CD after 50,000, step up CD after 10,000, batch size = 128, and edges between visible to hidden layer are initialised using dense Gaussian square root while MSE is applied as loss function. After successfully pretraining the DBMs' main constructs, the whole network is fine-tuned with parameter initialisation from the pretrained weights with linear activation, no dropout, batch size = 128, evaluate after 5000 and optimiser step = 200,000. For completeness, all parameter details for all datasets can also be found in our shared code repository.

6.2.4 Intrinsic evaluation of joint document representation

In this section, we aim to evaluate the extent to which the proposed representation is capable of capturing similarities between documents on the different dimensions (aspects and sentiments). Good models will have words that are semantically similar in each topic. For evaluating similarity, we followed the triplet similarity approach proposed by Le and Mikolov (2014). The idea is to make a triplet of three documents:

two are very similar with regard to aspect or sentiment, and one comes from an unrelated category. Then we measured the cosine similarity of the extracted learned representations of the given documents to decide which pair is the most similar.

In order to thoroughly test our representation, we created sets of triplets for different combinations of aspects and sentiments, fixing the sentiment and changing the aspect, and vice versa; for example, two positive reviews of Food and one positive Non-food, and so on. We evaluated three sets of combinations: aspect-sentiment e.g. (+ve Food, +ve Food, +ve Non-food), aspect only e.g. (Food, Food, Non-food) and sentiment only e.g. (+ve, +ve, –ve). Here, 'Non-food' refers to a review that is randomly picked from any category other than Food when creating the triplet. We performed the experiment on SemEval14, Citysearch and SemEval16 on two shared aspects (Food and Service). The average number of triplets in each class is 650. We compared our representation against NeuralDoc (Card et al. 2018).

Table 6 summarises the accuracy of cosine similarity of triplets on three of our datasets. The accuracy represents the percentage of correctly identified pairs in the triplets. Due to space limitations, column names are given abbreviations, e.g. +ve (F/NF) refers to reviews that are positive Food/Non-food, –ve(S/NS) refers to negative Service/Non service, (F/N) refers to Food/Non Food, and (+ve/–ve) refers to reviews that are positive/negative, irrespective of which aspect they belong to. From this table, we can easily see that JASM's representation performs well in mapping similar reviews about the same aspects and sentiments close together in the latent space. JASM is superior to NeuralDoc, with an average increase in accuracy of 5% for both aspects (Food and Service) and of almost 10% for the sentiment dimension.

The second intrinsic evaluation methodology visualises the learned document representation of reviews. The purpose here is to show how our joint modelling can capture the similarities of various topics and sentiments in reviews, which can be shown as clusters.

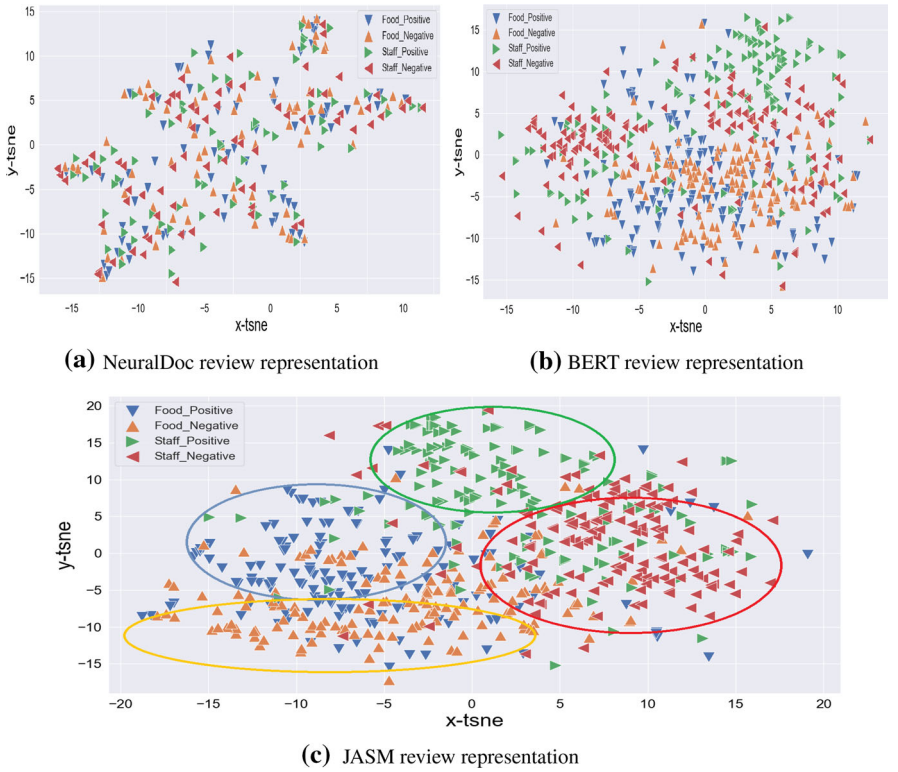
In order to present this, we visualise the representation of reviews of four different clusters: positive reviews about food, negative reviews about food, positive reviews about service and negative reviews about service. Since our representation is multi-dimensional, we employ t-SNE to reduce the number of dimensions to only two for ease of visualisation.

Figure 6 shows the t-SNE visualisation for four cluster aspect-sentiments of Citysearch dataset. We compare our representation to the review representations generated by NeuralDoc and BERT. In the figure, we compare the positive and negative sentiments expressed towards the food and service aspects; thus four clusters to compare. Positive food is represented by the blue down arrow, negative food is represented by the orange up arrow, positive staff is represented by the green right arrow while negative service is represented by the red left arrow. We can clearly see that JASM's representation differentiates all four clusters well, as highlighted by the coloured circles. Positive(Food) is at the top left with an arrow facing down, while negative(Food) is at the bottom left with an arrow facing up. At the top right, we can see positive(Staff), with negative(Staff) at the bottom right, with arrows pointing right and left, respectively. Also, we can see a very interesting proximity between negative(Food) and negative(Service) on the boundaries of their clusters at the bottom of Fig. 6c. Meanwhile, BERT's representation is more consistent and aligned than the NeuralDoc represen-

Table 6 Cosine similarity accuracy comparison of JASM and NeuralDoc

Dataset	Model	Aspect-Sentiment				Aspect		Sentiment
		+ve (F/NF)	–ve(F/NF)	+ve(S/NS)	–ve(S/NS)	(F/N)	(S/N)	(+ve/–ve)
SemEval 14	JASM	**89.89**	**87.65**	**92.96**	**90.15**	**88.32**	**92.21**	**81.03**
	Neural-Doc	88.50	86.54	89.81	86.21	79.4	85.13	71.27
CitySearch	JASM	**91.33**	**87.28**	**89.19**	**85.16**	**91.36**	**91.33**	**84.18**
	Neural-Doc	86.33	80.7	82.16	73.44	87.55	86.44	73.72
SemEval 16	JASM	**73.64**	**75.28**	**67.33**	**64.17**	**73.79**	**74.69**	**77.5**
	Neural-Doc	69.24	69.66	64.94	54.17	72.96	73.77	66.54

Bold values indicate the highest value with statistical significance



(a) NeuralDoc review representation

(b) BERT review representation

(c) JASM review representation

Fig. 6 T-SNE visualisation of Citysearch review representation (Color figure online)

tation. Reviews from different clusters are mixed together; sometimes the proximity of reviews shows that aspects precede sentiments, while other times the sentiments are stronger, irrespective of the aspect of the review. This inconsistency in differentiation comes mainly from the modelling process of leaning more towards topics than sentiments.

In summary, all previous intrinsic evaluations and visualisations confirm the consistency of the results in presenting the high representational power shown by JASM in comparison to other baselines. We can clearly see the different cluster formations of individual and combined aspects and sentiments.

6.2.5 Fine tuning for classification

We have shown that learning the joint aspect-sentiment document model is a fully unsupervised task. For some downstream applications there are some annotated data for specific opinion mining tasks such as classification, regression or even multitask. Traditionally, performing these tasks using regular document representation takes place in two separate steps: extracting the document representation, then using it as features to perform the designated task. However, our model architecture allows the two steps to be incorporated in our model fine-tuning step. This means that the parameters

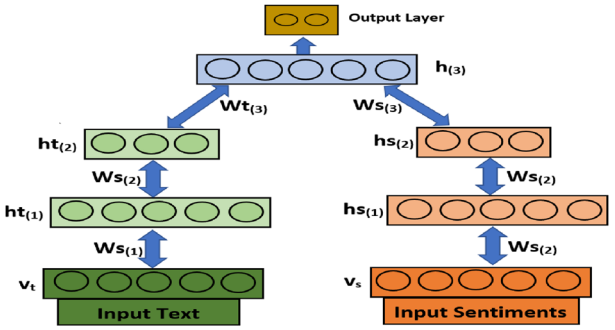


Fig. 7 JASM model fine-tuning for supervised classification tasks

learned from the unsupervised phase serve as initialisation for the supervised phase. Intuitively, this will be beneficial in driving the learning process along the direction of the specific task.

The fine tuning in this context has quite obvious advantages: not only do we perform the required downstream task, but we also learn a specialised representation directed towards the specific task. This means we can learn general-purpose opinion representations and with one extra step can also learn special-purpose ones as easily.

In Fig. 7, we can see the joint aspect-sentiment layer (The final layer in Fig. 2) is directly connected to the output layer of interest, e.g. classification task (aspect classification or sentiment classification). The output layer will hold the prediction outcome y for the designated classification task. Categorical label prediction is computed as:

$$\hat{y} = \arg\max_{W,b,a} P(y|v_s, v_t) \tag{15}$$

The model is refined by optimising loss functions with respect to all parameters (W, b, a) for all network layers of both pathways and the joint layer.

6.2.6 Extrinsic evaluation of joint document representation

In this section, we evaluate how JASM can fine-tune its parameters towards a supervised classification task as shown previously in the architecture given in Fig. 7. For this evaluation step, we test the accuracy by explicitly performing aspect and sentiment classification tasks.

Aspect classification evaluation Table 7 summarises the classification performance evaluation metrics of precision (PR), recall (RC) and F1 scores of the aspect classification tasks. In this table, we show results of Food and Service aspects of SemEval14 and Taste and Look aspects of the BeerAdvocate dataset. We can see that JASM outperforms all baselines. More specifically, in comparison to DBMDoc, which resembles only one side of our model (text, shown on the left side of Fig. 2) in modelling documents, our model is consistently 5% better on average. We can see that the addition in performance can be attributed to the sentiment component (shown on the right side of Fig. 2) that we added to our model. It is obvious that it clearly guides the modelling

Table 7 Aspect classification accuracy on SemEval14 and BeerAdvocate datasets

Dataset	JASM$_B$			BERT			NeuralDoc			DBMDoc		
	PR	RC	F1	PR	RC	F1	PR	RC	F1	PR	RC	F1
SemEval 14												
Food	**0.91**	**0.95**	**0.93**	0.88	0.84	0.86	0.72	0.82	0.77	0.87	0.91	0.89
Not-Food	**0.93**	**0.92**	**0.92**	0.85	0.89	0.87	0.78	0.68	0.72	0.91	0.87	0.89
Avg/total	**0.92**	**0.94**	**0.93**	0.87	0.87	0.87	0.75	0.75	0.75	0.89	0.89	0.89
Service	**0.89**	**0.92**	**0.90**	0.84	0.77	0.80	0.71	0.43	0.53	0.88	0.84	0.86
Not-service	**0.93**	**0.89**	**0.91**	0.82	0.84	0.83	0.59	0.82	0.68	0.84	0.88	0.86
Avg/total	**0.91**	**0.91**	**0.91**	0.83	0.81	0.82	0.65	0.62	0.61	0.86	0.86	0.86
BeerAdvocate												
Taste	**0.89**	**0.85**	**0.87**	0.84	0.81	0.82	0.85	0.74	0.79	0.83	0.73	0.78
Not-Taste	**0.84**	**0.89**	**0.86**	0.83	0.85	0.84	0.77	0.88	0.82	0.76	0.85	0.8
Avg/total	**0.87**	**0.87**	**0.87**	0.84	0.83	0.83	0.81	0.81	0.81	0.8	0.79	0.79
Look	**0.96**	**0.97**	**0.96**	0.88	0.91	0.89	0.95	0.92	0.94	0.93	0.91	0.92
Not-Look	**0.97**	**0.95**	**0.96**	0.92	0.89	0.90	0.92	0.96	0.94	0.91	0.93	0.92
Avg/total	**0.97**	**0.96**	**0.96**	0.90	0.90	0.90	0.94	0.94	0.94	0.92	0.92	0.92

Bold values indicate the highest value with statistical significance

process towards better aspect classification. Moreover, the recall scores of JASM$_B$ are quite impressive: the lowest recall value recorded is 87% and in some cases it is higher than precision for many aspects. This means that the model is sufficiently accurate not to miss the relevant aspects.

We know that BERT utilises the context awareness knowledge learnt in the pretraining stages and then fine-tunes its parameters towards the current classification task. Thus, for fair competition, JASM$_B$ is a variation of our model that incorporates the pretrained word representations provided by BERT for the reviews as input to the text DBM pathway, instead of word counts or TF–IDF. The performance of our model is consistently higher than that of BERT, by 8% and 5% on average on SemEval14 and BeerAdvocate datasets, respectively. The reason our model performs better than BERT is that it harnesses the extensive power and accuracy of DBMs while incorporating the conditioned sentiments over words, which shows an evident edge in efficiently discriminating aspects from reviews. This is also supported by the high performance of DBMDoc using the same inference process.

In general, the average review length in our datasets is 400 words, which was found to result in an expensive training phase for BERT as the longer the text, the longer and the more demanding the training process of BERT becomes. Moreover, it was found that BERT requires extensive computation power (Tensor Processing Unit-based), whereas JASM was trained only on a 1-GPU machine in almost one fifth of BERT's training time. Comparing JASM to NeuralDoc, the results support our previous claim about the accuracy of our inference process compared to the variational inference process. The inference process of NeuralDoc is based on the approximate variational inference process, which comes without warranty on some data distributions, whereas JASM uses MCMC, which is asymptotically exact in approximating the target distribution. On the other hand, BERT consistently performs better than NeuralDoc in classifying aspects of both SemEval14 and BeerAdvocate.

In Table 8, we compare variations of JASM, specifically $JASM_B$, where we utilise BERT's pretrained word embeddings to the base JASM model, utilising word counts as input to the text pathway. We perform this comparison to clarify two points: first, the flexibility of JASM in adopting and utilising different input representations for words, and second, that $JASM_B$'s performance, which is shown in Table 8, is not attributed only to the utilisation of BERT's pretrained embeddings in representing words. It is evident that the performance of $JASM_B$ is only 3% better on average than that of the base JASM using only the simple word representation. However, JASM outperforms the baselines of both NeuralDoc and DBMDoc consistently. This confirms again that the benefits gained by our proposed architecture in the accurate modelling of documents are undeniable.

Sentiment Classification In this section, we evaluate JASM's performance on the task of sentiment classification by fine-tuning its parameters towards this objective. Table 9 summarises the precision, recall and F1 scores of the sentiment classification task on SemEval14 and Citysearch datasets. We evaluated our model against baselines from different research directions, namely document-level sentiment analysis [BERT (Devlin et al. 2019)], aspect-level sentiment analysis [IAN (Ma et al. 2017)] and word-level analysis [Sent-LSTM (Miedema 2018)].

Table 8 Aspect classification accuracy of JASM variations on SemEval14 and BeerAdvocate datasets

Dataset	$JASM_B$			JASM		
	PR	RC	F1	PR	RC	F1
SemEval 14						
Food	**0.91**	**0.95**	**0.93**	0.89	0.94	0.92
Not-Food	**0.93**	**0.92**	**0.92**	0.94	0.89	0.91
Avg/total	**0.92**	**0.94**	**0.93**	0.92	0.92	0.92
Service	**0.89**	**0.92**	**0.90**	0.87	0.91	0.89
Not-Service	**0.93**	**0.89**	**0.91**	0.91	0.87	0.89
Avg/total	**0.91**	**0.91**	**0.91**	0.89	0.89	0.89
BeerAdvocate						
Taste	**0.89**	**0.85**	**0.87**	0.86	0.79	0.82
Not-taste	**0.84**	**0.89**	**0.86**	0.81	0.87	0.84
Avg/total	**0.87**	**0.87**	**0.87**	0.83	0.83	0.83
Look	**0.96**	**0.97**	**0.96**	0.94	0.95	0.95
Not-look	**0.97**	**0.95**	**0.96**	0.95	0.94	0.95
Avg/total	**0.97**	**0.96**	**0.96**	0.95	0.95	0.95

Bold values indicate the highest value with statistical significance

In Table 9, we can see that $JASM_B$ shows the best classification performance on Citysearch dataset, while its performance is only 1–2% lower on average on precision and recall than IAN's performance on the SemEval14 dataset. We can also see that the performance of IAN on SemEval is the best overall; however, it did require aspect term annotation, which is not available on the Citysearch dataset as it is practically very hard to acquire in real-life contexts. In this experiment, given that JASM uses only document-level annotation and not aspect-level annotation, which is more practical than IAN, we can argue that the performance is very comparable. Apart from that, BERT showed better performance on the task of sentiment classification than that of aspect classification (Table 7) on both datasets due to the semantic context learned in the pretraining process. Sent-LSTM shows surprisingly lower performance against all baselines (10% lower on average) than JASM; this may be due to its accounting only for sentiment words, irrespective of aspects assigned to it.

In Table 10, we compare $JASM_B$ to JASM in a similar way to the aspect classification comparison. We can see that $JASM_B$ is better on average, with a 4–5% increase in accuracy over JASM. This aligns with the fact that BERT's performance is better on sentiment classification than on aspect classification, which is mirrored to JASM while using the BERT word embeddings. This is a very interesting observation: we can make use of this in our future work to fine-tune the BERT embeddings towards aspect classification for better performance.

We also wanted to evaluate the effect of varying the number of hidden units in the top joint layer of JASM. Figure 8 shows the effect of such variations on both tasks (aspect and sentiment classification). Our current design choice for the number of hidden units in the top layer is set to the summation of the number of hidden units in the second hidden layer of both the aspect DBM and the sentiment DBM. We fixed the

Table 9 Sentiment classification accuracy on Citysearch and SemEval14 datasets

Dataset	$JASM_B$			IAN			Sent-LSTM			BERT		
	PR	RC	F1	PR	RC	F1	PR	RC	F1	PR	RC	F1
SemEval 14												
Negative	**0.86**	**0.83**	**0.84**	0.79	0.70	0.75	0.67	0.46	0.55	**0.86**	0.81	0.83
Positive	0.84	0.86	0.85	**0.89**	**0.93**	**0.91**	0.79	0.90	0.84	0.82	0.85	0.83
Avg/total	0.85	0.85	0.85	**0.86**	**0.87**	**0.86**	0.75	0.76	0.75	0.84	0.83	0.83
CitySearch												
Negative	**0.85**	**0.94**	**0.89**	–	–	–	0.60	0.63	0.62	0.84	0.89	0.86
Positive	**0.92**	0.82	**0.87**	–	–	–	0.83	0.81	0.82	0.87	**0.85**	0.86
Avg/total	**0.89**	**0.88**	**0.88**	–	–	–	0.76	0.76	0.76	0.86	0.87	0.86

Bold values indicate the highest value with statistical significance

Table 10 Sentiment classification accuracy of JASM variations on Citysearch and SemEval14 datasets

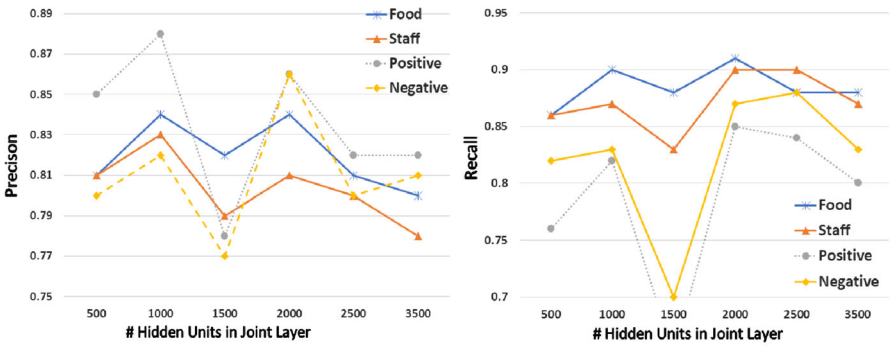
Dataset	$JASM_B$			JASM		
	PR	RC	F1	PR	RC	F1
SemEval 14						
Negative	**0.86**	**0.83**	**0.84**	0.81	0.78	0.79
Positive	**0.86**	**0.86**	**0.85**	0.82	0.80	0.84
avg/total	**0.85**	**0.85**	**0.85**	0.80	0.80	0.85
CitySearch						
Negative	**0.85**	**0.94**	**0.89**	0.81	0.93	0.87
Positive	**0.92**	**0.82**	**0.87**	0.91	0.78	0.84
Avg/total	**0.89**	**0.88**	**0.88**	0.86	0.85	0.85

Bold values indicate the highest value with statistical significance

number of units in the second layer of the DBM in both pathways to 1000 and varied the number of units in the joint layer from 500 to 3500 in increments of 500 units. A higher numbers of units in the last hidden layer was shown not to linearly enhance the classification metrics; rather, these eventually decline after 2000 units. From the figure, we can see that having 1000 hidden units in the last layer shows consistently better recall and precision for all tasks. This demonstrates that the optimal learning capability of the joint model is reached with a lower number of hidden units rather than the summation of the number of units in the second layer of both DBMs. This can still be different to some extent with other datasets, which can be empirically verified as being a design parameter.

7 Limitations

In this section, we discuss some of the limitations of our proposed model. These limitations can be divided into two categories, those related to the joint modelling of aspects and sentiments, and those that are related to the spam detection phase.



(a) Precision vs number of hidden units **(b)** Recall vs number of hidden units

Fig. 8 Precision and recall versus number of hidden units in joint layer

The JASM model is trained to be context-free as we use word count over vocabulary for each document in the training process. This means that the location of words in sentences and the relationship of sentences to other sentences in a paragraph are not taken into account. Contextual analysis gives a clear and nuanced story regarding why reviewers express given sentiments. Non-inclusion of this type of information can adversely affect the correct identification of the polarities of sentiments that are constrained by their context.

The loss of co-references and ambiguity resolution is another unfavourable effect of context-free representations that can also cause degradation in connecting related aspects. We can partially deal with the problem when we utilise BERT's representation as input to the text DBM for better performance, as shown in our experiments, where BERT takes into account word contexts. However, the word polarities input to the sentiment side DBM is still fully context-free. Moreover, when it comes to irony and sarcasm, e.g. where negative sentiments are expressed using positive words, JASM cannot correctly deal with the detection of the right sentiment. Thus, an obvious direction of potential future work for JASM is to explore the efficient inclusion of contextual information in the modelling process.

Regarding the spam detection phase, there are two main limitations. First, a limitation in regard to the time taken to train and run our model is that it requires the iterative training of an LSTM model while applying RVAE to filter out spam from training to learn the normal reviewing opinion trend. At prediction time, we perform the composite step of using the trained LSTM to predict the trend, then run RVAE (which internally takes time) to train a new VAE to isolate spam instances. Although this design is proven to be very effective, it takes a long time to run, especially at prediction time. This limits the use case of our model to batch processing rather than being able to be run on incoming real-time traffic. The second limitation is the λ setting for RVAE, which directly influences the rate of spam detected. A very small value can isolate a huge number of instances as anomalies, which is not optimal. It can be difficult to set it and (more importantly) to mitigate false spam rate alarms. One potential direction for future work relating to this issue is to automatically control λ values in relation to input data characteristics.

8 Conclusion

The importance of online reviews for customers has made them the target of constant malicious manipulation attacks which in turn inspired our research. Existing work on opinion spam detection has mostly focused on employing users' behavioural information to detect spam reviews; we have shown this to be non-optimal for singleton spam detection.

In this paper, we have presented a novel spam review detection approach specifically targeting the common case of singleton spam reviews. We are motivated to minimise the use of expensive human annotations, while analysing customer reviews in the temporal context for identification of spamming attacks. As a result, we propose a novel totally unsupervised approach to detect singleton spam reviews based on the analysis of opinion fluctuations represented by sentiments expressed towards aspects of reviewed entities over time. To reach our goal, we formulated the problem as a temporal anomaly detection problem. First, we proposed a novel joint document modelling to learn a comprehensive aspect-sentiment representation using deep Boltzmann machines (DBMs) from review texts, unsupervised. Then, we designed a long short-term memory (LSTM) network to learn hidden non-linear correlations behind the evolution of opinions over time from our learned abstract space review representation, to discover the normal trend. Finally, to overcome the need for a pre-set error threshold to identify spam instances, we introduced a novel Robust Variational Autoencoder (RVAE). This was achieved by reconstructing the learned opinion trend evolution over time (in comparison to actual trend) along with the expected error distributions, while iteratively isolating spam instances to minimise reconstruction errors.

We conducted thorough experiments to evaluate both components of our solution. In our evaluation, we conducted thorough experiments on three real-life annotated Yelp review datasets. We compared the performance accuracy of our proposed model against state-of-the-art baselines. The experimental results show that our approach accurately and consistently captures singleton spam reviews and outperforms the existing approaches. Based on our analysis, we find strong evidence that links the evolution of sentiment patterns over time to opinion spam. The content provided by spammers is becoming increasingly advanced in terms of the amount of detail and length of reviews provided. This makes traditional methods analysing only the textual content of reviews sub-optimal in detecting spams. We provide our annotated and preprocessed datasets, along with the implementation of our models, as open sources in support of the research community.

References

Ahmed M, Mahmood AN, Hu J (2016) A survey of network anomaly detection techniques. J Netw Comput Appl 60:19–31

Akcay S, Atapour-Abarghouei A, Breckon TP (2018) Ganomaly: semi-supervised anomaly detection via adversarial training. In: Proceedings of Asian conference on computer vision (ACCV), pp 622–637

Akoglu L, Chandy C R andFaloutsos (2013) Opinion fraud detection in online reviews by network effects. In: Proceedings of international conference on web and social media (ICWSM), pp 2–11

Aygun RC, Yavuz AG (2017) Network anomaly detection with stochastically improved autoencoder based models. In: Proceedings of international conference on cyber security and cloud computing (CSCloud), pp 193–198
Baldi P (2012) Autoencoders, unsupervised learning, and deep architectures. In: Proceedings of international conference on machine learning (ICML), pp 37–49
Blei D, Ng A, Jordan I (2003) Latent Dirichlet allocation. J Mach Learn Res 3:993–1022
Bontemps L, McDermott J, Le-Khac NA (2016) Collective anomaly detection based on long short-term memory recurrent neural networks. In: Proceedings of international conference on foundations of restoration ecology (ICFRE), pp 141–152
Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge University Press, Cambridge
Byeon W, Breuel TM, Raue F, Liwicki M (2015) Scene labeling with lstm recurrent neural networks. In: Proceedings of conference on computer vision and pattern recognition (CVPR), pp 3547–3555
Candès EJ, Li X, Ma Y, Wright J (2011) Robust principal component analysis? J ACM 58:1–37
Card D, Tan C, Smith N (2018) Neural models for documents with metadata. In: Proceedings of the Association for Computational Linguistics (ACL), pp 2031–2040
Chen P, Sun Z, Bing L, Yang W (2017) Recurrent attention network on memory for aspect sentiment analysis. In: Proceedings of conference on empirical methods in natural language processing (EMNLP), pp 452–461
Christy A, Gandhi GM, Vaithyasubramanian S (2015) Cluster based outlier detection algorithm for healthcare data. Procedia Comput Sci 50:209–215
Colhon M, Vlăduţescu S, Negrea X (2017) How objective a neutral word is? A neutrosophic approach for the objectivity degrees of neutral words. Symmetry 9(11):280
Devlin J, Ming-Wei C, Kenton L, Toutanova K (2019) Bert: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of North American Chapter of the Association for Computational Linguistics (NAACL), pp 4171–4186
Erfani SM, Baktashmotlagh M, Moshtaghi M, Nguyen V, Leckie C, Bailey J, Ramamohanarao K (2017) From shared subspaces to shared landmarks: a robust multi-source classification approach. In: Proceedings of Association for the Advancement of Artificial Intelligence (AAAI)
Eskin E, Arnold A, Prerau M, Portnoy L, Stolfo S (2002) A geometric framework for unsupervised anomaly detection. In: Barbará D, Jajodia S (eds) Applications of data mining in computer security. Springer, Boston, pp 77–101
Estiri H, Murphy SN (2019) Semi-supervised encoding for outlier detection in clinical observation data. Comput Methods Programs Biomed 181:104830
Fei G, Mukherjee A, Liu B, Hsu M, Castellanos M, Ghosh R (2013) Exploiting burstiness in reviews for review spammer detection. In: Proceedings of international conference on web and social media (ICWSM), pp 175–184
Fu P, Lin Z, Yuan F, Wang W, Meng D (2018) Learning sentiment-specific word embedding via global sentiment representation. In: Proceedings of Association for the Advancement of Artificial Intelligence (AAAI)
Ganu G, Elhadad N, Marian A (2009) Beyond the stars: improving rating predictions using review text content. Web Databases 9:1–6
García-Pablos A, Cuadros M, Rigau G (2018) W2vlda: Almost unsupervised system for aspect based sentiment analysis. Expert Syst Appl 91:127–137
Garcia-Teodoro P, Diaz-Verdejo J, Maciá-Fernández G, Vázquez E (2016) Anomaly-based network intrusion detection: techniques, systems and challenges. Comput Secur 28:18–28
Ghanem B, Rosso P, Rangel F (2020) An emotional analysis of false information in social media and news articles. ACM Trans Internet Technol 20(2):1–18
Graves A, Schmidhuber J (2005) Framewise phoneme classification with bidirectional lstm and other neural network architectures. Neural Netw 18(5):602–610
Greff K, Srivastava RK, Koutník J, Steunebrink BR, Schmidhuber J (2016) Lstm: a search space odyssey. IEEE Trans Neural Netw Learn Syst 28(10):2222–2232
Gupta P, Chaudhary Y, Buettner F, Schutze H (2019) Document informed neural autoregressive topic models with distributional prior. In: Proceedings of Association for the Advancement of Artificial Intelligence (AAAI), pp 6505–6512
Görnitz N, Kloft M, Rieck K, Brefeld U (2013) Toward supervised anomaly detection. J Artif Intell Res 46:235–262

Hai Z, Chang K, Kim J (2011) Implicit feature identification via co-occurrence association rule mining. In: Proceedings of international conference on intelligent text processing and computational linguistics (CICLing), pp 393–404

He S, Wang S, Lan W, Fu H, Ji Q (2013) Facial expression recognition using deep Boltzmann machine from thermal infrared images. In: Proceedings of conference of the Association for the Advancement of Affective Computing (AAAC), pp 239–244

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780

Hu M, Liu B (2004a) Mining and summarizing customer reviews. In: Proceedings of special interest group on knowledge discovery in data (ACM SIGKDD), pp 168–177

Hu M, Liu B (2004b) Mining and summarizing customer reviews. In: Proceedings of special interest group on knowledge discovery in data (ACM SIGKDD), pp 168–177

Huayi L, Geli F, Shuai W, Bing L, Weixiang S, Mukherjee A, Jidong S (2017) Bimodal distribution and co-bursting in review spam detection. In: Proceedings of international world wide web conference (WWW), pp 1063–1072

Hundman K, Constantinou V, Laporte C, Colwell I, Soderstrom T (2018) Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In: Proceedings of special interest group on knowledge discovery in data (ACM SIGKDD), pp 387–395

Jakob N, Gurevych I (2010) Extracting opinion targets in a single- and cross-domain setting with conditional random fields. In: Proceedings of empirical methods in natural language processing (EMNLP), pp 1035–1045

Jo Y, Oh A (2011) Aspect and sentiment unification model for online review analysis. In: Proceedings of conference on web search and data mining (WSDM), pp 815–824

Kazemi S, Abghari S, Lavesson N, Johnson H, Ryman P (2016) Open data for anomaly detection in maritime surveillance. Expert Syst Appl 40(14):5719–5729

Kim S, Choi Y, Lee M (2015) Deep learning with support vector data description. Neurocomputing 165:111–117

Kobayashi N, Iida R, Inui K, Matsumoto Y (2006) Opinion mining on the web by extracting subject-aspect-evaluation relations. In: Proceedings of Association for the Advancement of Artificial Intelligence (AAAI), pp 86–91

Kou Y, Lu CT, Chen D (2006) Spatial weighted outlier detection. In: Proceedings of Society for Industrial and Applied Mathematics (SIAM), pp 614–618

Kumar D, Shaalan Y, Zhang X, Chan J (2018) Identifying singleton spammers via spammer group detection. In: Proceedings of Pacific-Asia conference on knowledge discovery and data mining (PAKDD), pp 175–184

Lai S, Xu L, Liu K, Zhao J (2015) Recurrent convolutional neural networks for text classification. In: Proceedings of Association for the Advancement of Artificial Intelligence (AAAI), pp 2267–2273

Le Q, Mikolov T (2014) Distributed representations of sentences and documents. In: Proceedings of international conference on machine learning (ICML), pp 1188–1196

Le Q, Mikolov T (2016) Fake it till you make it: reputation, competition, and yelp review fraud. Manag Sci 62(12):3412–3427

Li J, Cardie C, Li S (2013) Topicspam: a topic-model-based approach for spam detection. In: Proceedings of The Association for Computational Linguistics (ACL), pp 217–221

Li J, Ott M, Cardie C, Hovy E (2014) Towards a general rule for identifying deceptive opinion spam. In: Proceedings of the Association for Computational Linguistics (ACL), pp 1566–1576

Lim E, Nguyen V, Jindal N, Liu B, Lauw HW (2008) Detecting product review spammers using rating behaviors. In: Proceedings of conference on information and knowledge management (CIKM), pp 939–948

Liu B (2010) Sentiment analysis and subjectivity. Handbook of natural language orocessing, vol 2, 2nd edn. Chapman and Hall, Boca Raton, pp 627–666

Liu B, Zhang L (2012) Sentiment analysis and opinion mining. Synth Lect Hum Lang Technol 5(1):1–167

Liu F, Ting K, Zhou Z (2008) Isolation forest. In: Proceedings of international conference on data mining (ICDM), pp 413–422

Liu FT, Ting KM, Zhou ZH (2010) On detecting clustered anomalies using sciforest. In: Proceedings of European conference on machine learning and principles and practice of knowledge discovery in databases (ECML PKDD), pp 274–290

Logeswaran L, Lee H (2018) An efficient framework for learning sentence representations. In: Proceedings of international conference on learning representations (ICLR)

Luyang L, Qin B, Wenjing R, Liu T (2016) Document representation and feature combination for deceptive spam review detection. Neuro Comput 254:33–41
Ma J, Sun L, Wang H, Zhang Y, Aickelin U (2016) Supervised anomaly detection in uncertain pseudo-periodic data streams. ACM Trans Internet Technol 16(1):235–262
Ma D, Li S, Zhang X, Wang H (2017) Interactive attention networks for aspect-level sentiment classification. In: Proceedings of international joint conferences on artificial intelligence (IJCAI), pp 4068–4074
Maas A, Daly R, Pham P, Huang D, Ng A, Potts C (2011) Learning word vectors for sentiment analysis. In: Proceedings of The Association for Computational Linguistics (ACL)
Malhotra P, Lovekesh V, Shroff G, Argarwal P (2015) Long short term memory networks for anomaly detection in time series. In: Proceedings of European symposium on artificial neural networks (ESANN), pp 665–674
McAuley J, Leskovec J (2013) Hidden factors and hidden topics: understanding rating dimensions with review text. In: Proceedings of conference on recommender systems (RecSys), pp 165–172
McAuley J, Leskovec J, Jurafsky D (2012) Learning attitudes and attributes from multiaspect reviews. In: Proceedings of international conference on data Mining (ICDM), pp 1020–1025
Mei Q, Ling X, Wondra M, Su H, Zhai C (2007) Topic sentiment mixture: modeling facets and opinions in weblogs. In: Proceedings of international world wide web conference (WWW), pp 171–180
Miedema F (2018) Sentiment analysis with long short-term memory. Vrije Universiteit Amsterdam 1
Mihalcea R, Strapparava C (2009) The lie detector: explorations in the automatic recognition of deceptive language. In: Proceedings of international joint conference on natural language processing (IJCNLP), pp 309–312
Mikolov T, Sutskever I, Chen K, Corrado G, Dean J (2013) Distributed representations of words and phrases and their compositionality. Adv Neural Inf Process Syst 26:3111–3119
Moghaddam S, Ester M (2011) Ilda: Interdependent lda model for learning latent aspects and their ratings from online product reviews. In: Proceedings of conference on research and development in information retrieval (ACM SIGIR), pp 665–674
Mukherjee A, Liu B (2010) Modeling review comments. In: Proceedings of The Association for Computational Linguistics (ACL), pp 320–329)
Mukherjee A, Liu B, Wang J, Glance N, Jindal N (2011) Detecting group review spam. In: Proceedings of international world wide web conference (WWW), pp 93–94
Mukherjee A, Kumar A, Liu B, Wang J, Hsu M, Castellanos M, Ghosh R (2013) Spotting opinion spammers using behavioral footprints. In: Proceedings of special interest group on knowledge discovery in data (ACM SIGKDD), pp 632–640
Narisawa K, Hideo B, Hatano K, Takeda M (2007) Unsupervised spam detection based on string alienness measures. In: Proceedings of international conference on discovery science (DS), pp 161–172
Nguyen-Hoang B, Ha Q, Nghiem M (2016) Aspect-based sentiment analysis using word embedding restricted Boltzmann machines. In: Proceedings of international conference on computational social networks (CSoNet), pp 285–297
Ott M, Choi Y, Cardie C, Hancok J (2011) Finding deceptive opinion spam by any stretch of the imagination. In: Proceedings of The Association for Computational Linguistics (ACL), pp 309–319
Pennington J, Socher R, Manning C (2014) Glove: Global vectors forward representation. In: Proceedings of empirical methods in natural language processing (EMNLP), pp 1532–1543
Peters ME, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, Zettlemoyer L (2018) Deep contextualized word representations. In: Proceedings of North American Chapter of the Association for Computational Linguistics (NAACL), pp 2227–2237
Phua C, Alahakoon D, Lee V (2004) Minority report in fraud detection: classification of skewed data. ACM SIGKDD Explor Newsl 6(1):50–59
Pontiki M, Galanis D, Pavlopoulos J, Papageorgiou H, Androutsopoulos I, Manandhar S (2014) Proceedings of international workshop on semantic evaluation-2014 task 4: aspect based sentiment analysis. In: Proceedings of international workshop on semantic evaluation (SemEval), pp 27–35
Pontiki M, Galanis D, Pavlopoulos J, Papageorgiou H, Manandhar S (2015) Proceedings of international workshop on semantic evaluation-2015 task 12: aspect based sentiment analysis. In: Proceedings of international workshop on semantic evaluation (SemEval), pp 486–495
Pontiki M, Galanis D, Papageorgiou H, Androutsopoulos I, Manandhar S, AL-Smadi M, Al-Ayyoub M, Zhao Y, Qin B, of The International Conference on Language Resources ODC, Evaluationq U (2016) Proceedings of international workshop on semantic evaluation-2016 task 5: aspect based sentiment analysis. In: Proceedings of international workshop on semantic evaluation (SemEval), pp 19–30

Principi E, Vesperini F, Squartini S, Piazza F (2017) Acoustic novelty detection with adversarial autoencoders. In: Proceedings of international joint conference on neural networks (IJCNN), pp 3324–3330
Qiu G, Liu B, Bu J, Chen C (2009) Expanding domain sentiment lexicon through double propagation. In: Proceedings of international joint conferences on artificial intelligence (IJCAI), pp 1199-1204
Racah E, Beckham C, Maharaj T, Kahou SE, Prabhat M, Pal C (2017) Extremeweather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events. In: Proceedings of neural information processing systems (NIPS), pp 3402–3413
Rayana S, Akoglu L (2015) Collective opinion spam detection: Bridging review networks and metadata. In: Proceedings of special interest group on knowledge discovery in data (ACM SIGKDD), pp 985–994
Saini M, Sharan A (2017) Identifying deceptive opinion spam using aspect-based emotions and human behavior modeling. Int J Hybrid Inf Technol 10(1):447–456
Sakurada M, Yairi T (2014) Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: Proceedings of workshop on machine learning for sensory data analysis (MLSDA)
Salakhutdinov R, Hinton G (2009) Deep Boltzmann machines. In: Proceedings of artificial intelligence and statistics (AISTATS), pp 448–455
Salvador S, Chan P, Brodie J (2004) Learning states and rules for time series anomaly detection. In: Proceedings of The Florida artificial intelligence research society (FLAIRS), pp 306–311
Sandulescu V, Ester M (2015) Detecting singleton review spammers using semantic similarity. In: Proceedings of international world wide web conference (WWW), pp 971–976
Sauper C, Barzilay R (2013) Automatic aggregation by joint modeling of aspects and values. J Artif Intell Res 46:89–127
Savage D, Zhang X, Yu X, Chou P, Wang Q (2015) Detection of opinion spam based on anomalous rating deviation. Expert Syst Appl 42(22):8650–8657
Shehnepoor S, Salehi M, Farahbakhsh R, Crespi N (2019) Netspam: a network-based spam detection framework for reviews in online social media. IEEE Trans Inf Forensics Secur 12(7):1585–1595
Shojaee S, Murad M, A BA, Sharef NM, Nadali S (2013) Detecting deceptive reviews using lexical and syntactic features. In: Proceedings of international conference on intelligent systems design and applications (ISDA), pp 53–58
Siegel S, Castellan J (1988) Nonparametric statistics for the behavioral sciences, 2nd edn. McGraw-Hill, New York
Solberg HE, Lahti A (2005) Detection of outliers in reference distributions: performance of Horn's algorithm. Clin Chem 51(12):2326–2332
Srivastava A, Kundu A, Sural S, Majumdar A (2008) Credit card fraud detection using hidden Markov model. IEEE Trans Dependable Secure Comput 5(1):37–48
Srivastava N, Salakhutdinov R, Hinton G (2013) Modeling documents with a deep Boltzmann machine. In: Proceedings of conference on uncertainty in artificial intelligence (UAI), pp 616–624
Sun J, Wang X, Xiong N, Shao J (2018) Learning sparse representation with variational auto-encoder for anomaly detection. IEEE Access 6:33353–33361
Sundermeyer M, Schlüter R, Ney H (2012) Lstm neural networks for language modeling. In: Proceedings of conference of the International Speech Communication Association (ISCA)
Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: Proceedings of neural information processing systems (NIPS), pp 3104–3112
Tang D, Wei F, Yang N, Zhou M, Liu T, Qin B (2015) Learning sentiment-specific word embedding for twitter sentiment classification. In: Proceedings of the Association for Computational Linguistics (ACL), pp 1555–1565
Tausczik YR, Pennebaker JW (2010) The psychological meaning of words: Liwc and computerized text analysis methods. J Lang Soc Psychol 29(1):24–54
Titov I, McDonald R (2008) A joint model of text and aspect ratings for sentiment summarization. In: Proceedings of The Association for Computational Linguistics (ACL), pp 308–316
Titov I, McDonald R (2009) Modeling online reviews with multi-grain topic models. In: Proceedings of International World Wide Web Conference (WWW), pp 111–120
Toprak C, Jakob N, Gurevych I (2010) Sentence and expression level annotation of opinions in user-generated discourse. In: Proceedings of The Association for Computational Linguistics (ACL), pp 575–584
van der Maaten L, Hinton G (2008) Visualizing high-dimensional data using t-sne. Mach Learn Res 9:2579–2605

Verwimp L, Pelemans J, , Wambacq P (2017) Character-word lstm language models. In: Proceedings of conference of the European chapter of the Association for Computational Linguistics (EACL), pp 417–427

Wang L, Liu K, Cao Z, Zhao J, de Melo G (2015) Sentiment-aspect extraction based on restricted Boltzmann machines. In: Proceedings of the Association for Computational Linguistics (ACL), pp 616–625

Wang X, Liu K, Zhao J (2017) Detecting deceptive review spam via attention-based neural networks. In: Proceedings of national conference on Natural Language Processing and Chinese Computing (NLPCC), pp 866–876

Wang H, Bah MJ, Hammad M (2019) Progress in outlier detection techniques: a survey. IEEE Access 7:107964–108000

Xie S, Wang G, Lin S, Yu PS (2012) Review spam detection via time series pattern discovery. In: Proceedings of knowledge discovery and data mining (KDD), pp 823–831

Xie P, Deng Y, Xing E (2015) Diversifying restricted Boltzmann machine for document modeling. In: Proceedings of special interest group on knowledge discovery in data (ACM SIGKDD), pp 1315–1324

Xu H, Chen W, Zhao N, Li Z, Bu J, Li Z, Liu Y, Zhao Y, Pei D, Feng Y, Chen J (2018) Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In: Proceedings of International World Wide Web Conference (WWW), pp 187–196

Ye Y, Akoglu L (2017) Discovering opinion spammer groups by network footprints. In: Proceedings of joint European conference on machine learning and knowledge discovery in databases (ECML KDD), pp 97–113

Ye J, Kumar S, Akoglu L (2016) Temporal opinion spam detection by multivariate indicative signals. In: Proceedings of international conference on web and social media (ICWSM), pp 743–746

Zhang L, Liu B, Lim SH, O'Brien-Strain E (2010) Extracting and ranking product features in opinion documents. In: Proceedings of international conference on computational linguistics (COLING), pp 1462–1470

Zhao W, Jiang J, Yan H, Li X (2010a) Jointly modeling aspects and opinions with a maxent-lda hybrid. In: Proceedings of empirical methods in natural language processing (EMNLP), pp 56–65

Zhao Y, Qin B, Hu S, Liu T (2010b) Generalizing syntactic structures for product attribute candidate extraction. In: Proceedings of conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT), pp 377–380

Zhao S, Xu Z, Liu L, Guo M, Yun J (2018) Towards accurate deceptive opinion spam detection based on word order-preserving CNN. Math Probl Eng 2018:2410206

Zheng Y, Zhang H, Yu Y (2015) Detecting collective anomalies from multiple spatio-temporal datasets across different domains. In: Proceedings of international conference on advances in geographic information systems (ACM SIGSPATIAL), pp 1–10

Zhou C, Paffenroth R (2017) Anomaly detection with robust deep autoencoders. In: Proceedings of special interest group on knowledge discovery in data (ACM SIGKDD), pp 665–674

Zhu J, Wang H, Tsou B, Zhu M (2009) Multi-aspect opinion polling from textual reviews. In: Proceedings of conference on information and knowledge management (CIKM), pp 1799–1802

Zhuang L, Jing F, Zhu XY (2006) Movie review mining and summarization. In: Proceedings of conference on information and knowledge management (CIKM), pp 43–50

Zong B, Song Q, Min W M Rand Cheng, Lumezanu C, Cho D, Chen H (2018) Deep autoencoding Gaussian mixture model for unsupervised anomaly detection. In: Proceedings of international conference on learning representations (ICLR)

Affiliations

Yassien Shaalan[1] · Xiuzhen Zhang[1] · Jeffrey Chan[1] · Mahsa Salehi[2]

✉ Yassien Shaalan
yassien.shaalan@rmit.edu.au

Xiuzhen Zhang
xiuzhen.zhang@rmit.edu.au

Jeffrey Chan
jeffrey.chan@rmit.edu.au

Mahsa Salehi
mahsa.salehi@monash.edu

1 RMIT University, Melbourne, Australia

2 Monash University, Melbourne, Australia