CASCADE: Contextual Sarcasm Detection in Online Discussion Forums

Devamanyu Hazarika
School of Computing,
National University of Singapore
hazarika@comp.nus.edu.sg

Soujanya Poria
Artificial Intelligence Initiative,
A*STAR, Singapore
sporia@ihpc.a-star.edu.sg

Sruthi Gorantla
Computer Science & Automation,
Indian Institute of Science, Bangalore
gorantlas@iisc.ac.in

Erik Cambria
School of Computer Science and
Engineering, NTU, Singapore
cambria@ntu.edu.sg

Roger Zimmermann
School of Computing,
National University of Singapore
rogerz@comp.nus.edu.sg

Rada Mihalcea
Computer Science & Engineering,
University of Michigan, Ann Arbor
mihalcea@umich.edu

Abstract

The literature in automated sarcasm detection has mainly focused on lexical-, syntactic- and semantic-level analysis of text. However, a sarcastic sentence can be expressed with contextual presumptions, background and commonsense knowledge. In this paper, we propose a Context-tuAl SarCasm DEtector (CASCADE), which adopts a hybrid approach of both content- and context-driven modeling for sarcasm detection in online social media discussions. For the latter, CASCADE aims at extracting contextual information from the discourse of a discussion thread. Also, since the sarcastic nature and form of expression can vary from person to person, CASCADE utilizes user embeddings that encode stylometric and personality features of users. When used along with content-based feature extractors such as convolutional neural networks, we see a significant boost in the classification performance on a large Reddit corpus.

1 Introduction

Sarcasm is a linguistic tool that uses irony to express contempt. Its figurative nature poses a great challenge for affective systems performing sentiment analysis (Cambria et al., 2017). Previous research in automated sarcasm detection has primarily focused on lexical and pragmatic cues found in sentences (Kreuz and Caucci, 2007). In the literature, interjections, punctuations, and sentimental shifts have been considered as major indicators of sarcasm (Joshi et al., 2017). When such lexical cues are present in sentences, sarcasm detection can achieve high accuracy. However, sarcasm is also expressed implicitly, i.e., without the presence of such lexical cues. This use of sarcasm also relies on context, which involves the presumption of commonsense and background knowledge of an event. When it comes to detecting sarcasm in a discussion forum, it may not only be required to understand the context of previous comments but also the necessary background knowledge about the topic of discussion. The usage of slangs and informal language also diminishes the reliance on lexical cues (Satapathy et al., 2017). This particular type of sarcasm is tough to detect (Poria et al., 2016).

Contextual dependencies for sarcasm can take many forms. As an example, a sarcastic post from Reddit[1], *"I'm sure Hillary would've done that, lmao."* requires background knowledge about the event, i.e., Hillary Clinton's action at the time the post was made. Similarly, sarcastic posts like *"But atheism, yeah *that's* a religion!"* requires the knowledge that topics like *atheism* often contain argumentative discussions and, hence, they are more prone towards sarcasm.

The main aim of this work is sarcasm detection in online discussion forums. In particular, we propose a hybrid network, named CASCADE, that leverages both the *content* and the *context* required for sarcasm detection. It starts by processing contextual information in two ways. First, it performs user profiling to create user embeddings that capture indicative behavioral traits for sarcasm. Recent findings suggest that such modeling of the user and their preferences is highly effective for the given task (Amir et al.,

[1] http://reddit.com

Proceedings of the 27th International Conference on Computational Linguistics, pages 1837–1848
Santa Fe, New Mexico, USA, August 20-26, 2018.

2016). It makes use of users' historical posts to model their writing style (stylometry) and personality indicators, which are then fused into comprehensive user embeddings using a multi-view fusion approach, termed canonical correlation analysis (CCA) (Hotelling, 1936). Second, it extracts contextual information from the discourse of comments in the discussion forums. This is done by document modeling of these consolidated comments belonging to the same forum. We hypothesize that these discourse features would give the important contextual information, background cues along with topical information required for detecting sarcasm.

After the contextual modeling phase, CASCADE is provided with a comment for sarcasm detection. It performs content-modeling using a convolutional neural network (CNN) to extract its syntactic features. This CNN representation is then concatenated with the relevant user embedding and discourse features to get the final representation which is used for classification. The overall contribution of this work can be summarized as:

- We propose a novel hybrid sarcasm detector, CASCADE, that models both content and contextual information.
- We model stylometric and personality details of users along with discourse features of discussion forums to learn informative contextual representations. Experiments on a large Reddit corpus demonstrate significant performance improvement over state-of-the-art automated sarcasm detectors.

The remainder of the paper is organized as follows: Section 2 lists related works; Section 3 explains the process of learning contextual features comprising user embeddings and discourse features; Section 4 presents experimentation details of the model and result analysis; finally, Section 5 draws conclusions.

2 Related Work

Automated sarcasm detection is a relatively recent field of research. Previous works can be classified into two main categories: content- and context-based sarcasm detection models.

Content-based models: These networks model the problem of sarcasm detection as a standard classification task and try to find lexical and pragmatic indicators to identify sarcasm. Numerous works have taken this path and presented innovative ways to unearth interesting cues for sarcasm. Tepperman et al. (2006) investigate sarcasm detection in spoken dialogue systems using prosodic and spectral cues. Carvalho et al. (2009) use linguistic features like positive predicates, interjections and gestural clues such as emoticons, quotation marks, etc. Davidov et al. (2010), Tsur et al. (2010) use syntactic patterns to construct classifiers. González-Ibánez et al. (2011) also study the use of emoticons, mainly amongst tweets. Riloff et al. (2013) assert sarcasm to be a contrast to positive sentiment words and negative situations. Joshi et al. (2015) use multiple features comprising lexical, pragmatics, implicit and explicit context incongruity. In the explicit case, they include relevant features to detect thwarted sentimental expectations in the sentence. For implicit incongruity, they generalize Riloff et al. (2013) by identifying verb-noun phrases containing contrast in both polarities.

Context-based models: The usage of contextual sarcasm has increased in recent years, especially in online platforms. Texts found in microblogs, discussion forums, and social media are plagued by grammatical inaccuracies and contain information which is highly temporal and contextual. In such scenarios, mining linguistic information becomes relatively inefficient and the need arises for additional clues (Carvalho et al., 2009). Wallace et al. (2014) demonstrate this need by showing how traditional classifiers fail in instances where humans require additional context. They also indicate the importance of speaker and topical information associated to a text to gather such context. Poria et al. (2016) use additional information by sentiment, emotional and personality representations of the input text. Previous works have mainly used historical posts of users to understand sarcastic tendencies (Rajadesingan et al., 2015; Zhang et al., 2016). Khattri et al. (2015) try to discover users' sentiments towards entities in their histories to find contrasting evidence. Wallace et al. (2015) utilize sentiments and noun phrases used within a forum to gather context typical to that forum. Such forum-based modeling simulates user

communities. Our work follows a similar motivation as we explore the context provided by user profiling and the topical knowledge embedded in the discourse of comments in discussion forums (subreddits[2]).

Amir et al. (2016) performed user modeling by learning embeddings that capture homophily. This work is the closest to our approach given the fact that we too learn user embeddings to acquire context. However, we take a different approach that involves stylometric and personality description of the users. Empirical evidence shows that these proposed features are better than previous user modeling approaches. Moreover, we learn discourse features which has not been explored before in the context of this task.

3 Method

3.1 Task Definition

The task involves detection of sarcasm for comments made in online discussion forums, i.e., Reddit. Let us denote the set $U = \{u_1, ..., u_{N_u}\}$ for N_u-users, where each user participates across a subset of N_t-discussion forums (subreddits). For a comment C_{ij} made by the i^{th} user u_i in the j^{th} discussion forum t_j, the objective is to predict whether the comment posted is sarcastic or not.

3.2 Summary of the Proposed Approach

Given the comment C_{ij} to be classified, CASCADE leverages *content*- and *context*-based information from the comment. For content-based modeling of C_{ij}, a CNN is used to generate the representation vector $\vec{\boldsymbol{c}}_{i,j}$ for a comment. CNNs generate abstract representations of text by extracting location-invariant local patterns. This vector $\vec{\boldsymbol{c}}_{i,j}$ captures both syntactic and semantic information useful for the task at hand. For contextual modeling, CASCADE first learns user embeddings and discourse features of all users and discussion forums, respectively (Section 3.3). Following this phase, CASCADE then retrieves the learnt user embedding $\vec{\boldsymbol{u}}_i$ of user u_i and discourse feature vector $\vec{\boldsymbol{t}}_j$ of forum t_j. Finally, all three vectors $\vec{\boldsymbol{c}}_{i,j}$, $\vec{\boldsymbol{u}}_i$, and $\vec{\boldsymbol{t}}_j$ are concatenated and used for the classification (Section 3.6). One might argue that, instead of using one CNN, we could use multiple CNNs as in (Majumder et al., 2017), to get better text representations whenever a comment contains multiple sentences. However, that is out of the scope of this work. Here, we aim to show the effectiveness of user-specific analysis and context-based features extracted from the discourse. Also, the use of a single CNN for text representation helps to consistently compare our model with the state of the art.

3.3 Learning Contextual Features

In this section, we explain in detail the procedures to generate the contextual features, i.e., user embeddings and discourse features. The user embeddings try to capture users' traits that correlate to their sarcastic tendencies. These embeddings are created considering the accumulated historical posts of each user (Section 3.4). Contextual information are also extracted from the discourse of comments within each discussion forum. These extracted features are named as discourse features (Section 3.5). The aim of learning these contextual features is to acquire discriminative information crucial for sarcasm detection.

3.4 User Embeddings

To generate user embeddings, we model their stylometric and personality features and then fuse them using CCA to create a single representation. Below, we explain the generation of user embedding $\vec{\boldsymbol{u}}_i$, for the i^{th} user u_i. Figure 1 also summarizes the overall architecture for this kind of user profiling.

3.4.1 Stylometric features

People possess their own idiolect and authorship styles, which is reflected in their writings. These styles are generally affected by attributes such as gender, diction, syntactic influences, etc. (Cheng et al., 2011; Stamatatos, 2009) and present behavioral patterns which aid sarcasm detection (Rajadesingan et al., 2015).

We use this motivation to learn stylometric features of the users by consolidating their online comments into documents. We first gather all the comments by a user and create a document by appending them using a special delimiter `<END>`. An unsupervised representation learning method *ParagraphVector* (Le

[2] `http://reddit.com/reddits`

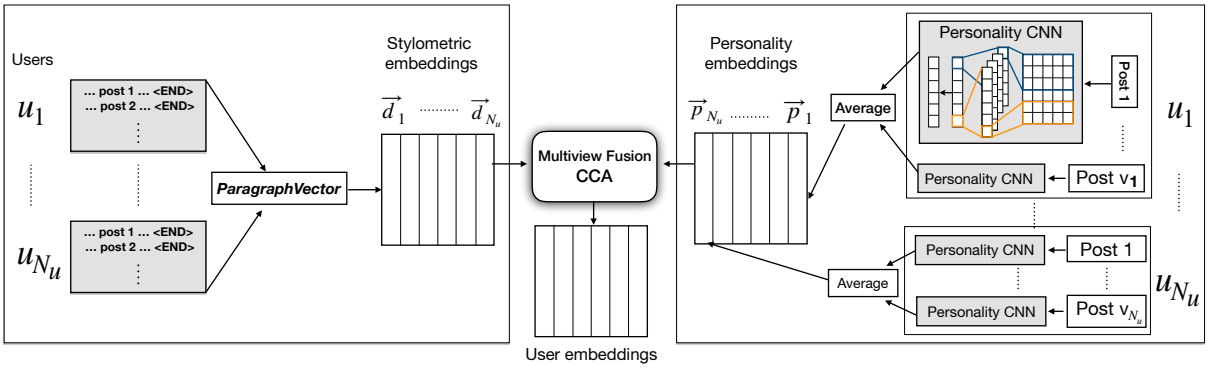


Figure 1: The figure describes the process of user profiling. Stylometric and personality embeddings are generated and then fused in a multi-view setting using CCA to get the user embeddings.

and Mikolov, 2014) is then applied on this document. This method generates a fixed-sized vector for each user by performing the auxiliary task of predicting the words within the documents. The choice of *ParagraphVector* is governed by multiple reasons. Apart from its ability to effectively encode a user's writing style, it has the advantage of applying to variable lengths of text. *ParagraphVector* also has been shown to perform well for sentiment classification tasks. The existence of synergy between sentiment and sarcastic orientation of a sentence also promotes the use of this method.

We now describe the functioning of this method. Every user document and all words within them are first mapped to unique vectors such that each vector is represented by a column in matrix $D \in \mathbb{R}^{d_s \times N_u}$ and $W_s \in \mathbb{R}^{d_s \times |V|}$, respectively. Here, d_s is the embedding size and $|V|$ represents the size of the vocabulary. *Continuous bag-of-words* approach (Mikolov et al., 2013) is then performed where a target word is predicted given the word vectors from its context window. The key idea here is to use the document vector of the associated document as part of the context words. More formally, given a user document d_i for user u_i comprising a sequence of n_i-words $\boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_{n_i}$, we calculate the average log probability of predicting each word within a sliding context window of size k_s. This average log probability is:

$$\frac{1}{n_i} \sum_{t=k_s}^{n_i - k_s} \log p(w_t | d_i, w_{t-k_s}, ..., w_{t+k_s}) \tag{1}$$

To predict a word within a window, we take the average of all the neighboring context word vectors along with the document vector $\vec{\boldsymbol{d}}_i$ and use a neural network with softmax prediction:

$$p(w_t | d_i, w_{t-k_s}, ..., w_{t+k_s}) = \frac{e^{\vec{\boldsymbol{y}}_{w_t}}}{\sum_i e^{\vec{\boldsymbol{y}}_i}} \tag{2}$$

Here, $\vec{\boldsymbol{y}} = [y_1, ..., y_{|V|}]$ is the output of the neural network, i.e.,

$$\vec{\boldsymbol{y}} = \mathrm{U}_d h(\vec{\boldsymbol{d}}_i, \vec{\boldsymbol{w}}_{t-k_s}, ..., \vec{\boldsymbol{w}}_{t+k_s}; D, W_s) + \vec{\boldsymbol{b}}_d \tag{3}$$

$\vec{\boldsymbol{b}}_d \in \mathbb{R}^{|V|}, U_d \in \mathbb{R}^{|V| \times d_s}$ are parameters and $h(\cdot)$ represents the average of vectors $\vec{\boldsymbol{d}}_i, \vec{\boldsymbol{w}}_{t-k_s}, ..., \vec{\boldsymbol{w}}_{t+k_s}$ taken from D and W_s. Hierarchical softmax is used for faster training (Morin and Bengio, 2005). Finally, after training, D learns the users' document vectors which represent their stylometric features.

3.4.2 Personality features

Discovering personality from text has numerous natural language processing (NLP) applications such as product recognition, mental health diagnosis, etc. Described as a combination of multiple characteristics, personality detection helps in identifying behavior, thought patterns of an individual. To model the dependencies of users' personality with their sarcastic nature, we include personality features in the user embeddings. Previously, Poria et al. (2016) also utilized personality features in sentences. However, we take a different approach of extracting the personality features of a user instead.

For user u_i, we iterate over all the v_i-comments $\{S^1_{u_i}, ..., S^{v_i}_{u_i}\}$ written by them. For each $S^j_{u_i}$, we provide the comment as an input to a pre-trained CNN which has been trained on a multi-label personality detection task. Specifically, the CNN is pre-trained on a benchmark corpus developed by Matthews and Gilliland (1999) which contains 2400 essays and is labeled with the Big-Five personality traits, i.e., Openness, Conscientiousness, Extraversion, Agreeableness, and Neuroticism (OCEAN). After the training, this CNN model is used to infer the personality traits present in each comment. This is done by extracting the activations of the CNN's last hidden layer vector, which we call as the personality vector $\vec{\boldsymbol{p}}^j_{u_i}$. The expectation over the personality vectors for all v_i-comments made by the user is then defined as the overall personality feature vector $\vec{\boldsymbol{p}}_i$ of user u_i:

$$\vec{\boldsymbol{p}}_i \;=\; \mathbb{E}_{j\in[v_i]}[\vec{\boldsymbol{p}}^j_{u_i}] \;=\; \frac{1}{v_i}\sum_{j=1}^{v_i}\vec{\boldsymbol{p}}^j_{u_i} \tag{4}$$

CNN: Here, we describe the CNN that generates the personality vectors. Given a user's comment, which is a text $S = [w_1, ..., w_n]$ composed of n words, each word w_i is represented as a word embedding $\vec{\boldsymbol{w}}_i \in \mathbb{R}^{d_{em}}$ using the pre-trained FastText embeddings (Bojanowski et al., 2016). A single-layered CNN is then modeled on this input sequence S (Kim, 2014). First, a convolutional layer is applied having three filters $F_{[1,2,3]} \in \mathbb{R}^{d_{em}\times h_{[1,2,3]}}$ of heights $h_{[1,2,3]}$, respectively. For each $k \in \{1, 2, 3\}$, filter F_k slides across S and extracts h_k-gram features at each instance. This creates a feature map vector $\vec{\boldsymbol{m}}_k$ of size $\mathbb{R}^{|S|-h_k+1}$, whose each entry $m_{k,j}$ is obtained as:

$$m_{k,j} = \alpha(\; F_k \cdot S_{[j:j+h_k-1]} + b_k \;) \tag{5}$$

here, $b_k \in \mathbb{R}$ is the bias and $\alpha(\cdot)$ is a non-linear activation function.

M feature maps are created from each filter F_k giving a total of $3M$ feature maps as output. Following this, a max-pooling operation is performed across the length of each feature map. Thus, for all M feature maps computed from F_k, output $\vec{\boldsymbol{o}}_{\boldsymbol{k}}$ is calculated as, $\vec{\boldsymbol{o}}_{\boldsymbol{k}} = [\; max(\vec{\boldsymbol{m}}^1_1), ..., \; max(\vec{\boldsymbol{m}}^{\mathrm{M}}_1) \;]$. Overall the max-pooling output is calculated by concatenation of each $\vec{\boldsymbol{o}}_{\boldsymbol{k}}$ to get $\vec{\boldsymbol{o}} = [\vec{\boldsymbol{o}}_{\boldsymbol{1}} \oplus \vec{\boldsymbol{o}}_{\boldsymbol{2}} \oplus \vec{\boldsymbol{o}}_{\boldsymbol{3}}] \in \mathbb{R}^{3M}$, where $\oplus$ represents concatenation. Finally, $\vec{\boldsymbol{o}}$ is projected onto a dense layer with d_p neurons followed by the final sigmoid-prediction layer with 5 classes denoting the five personality traits (Matthews et al., 2003). We use sigmoid instead of softmax to facilitate multi-label classification. This is calculated as:

$$\vec{\boldsymbol{q}} = \alpha(\; W_1\vec{\boldsymbol{o}} + \vec{\boldsymbol{b}}_1 \;) \tag{6}$$

$$\hat{\boldsymbol{y}} = \sigma(\; W_2\vec{\boldsymbol{q}} + \vec{\boldsymbol{b}}_2 \;) \tag{7}$$

$W_1 \in \mathbb{R}^{d_p\times 3M}, W_2 \in \mathbb{R}^{5\times d_p}, \vec{\boldsymbol{b}}_1 \in \mathbb{R}^{d_p}$ and $\vec{\boldsymbol{b}}_2 \in \mathbb{R}^5$ are parameters and $\alpha(.)$ represents non-linear activation.

3.4.3 Fusion

We take a multi-view learning approach to combine both stylometric and personality features into a comprehensive embedding for each user. We use CCA to perform this fusion. CCA captures maximal information between two views and creates a combined representation (Hardoon et al., 2004; Benton et al., 2016). In the event of having more than two views, fusion can be performed using an extension of CCA called *Generalized* CCA (see Appendix).

Canonical Correlation Analysis: Let us consider the learnt stylometric embedding matrix $D \in \mathbb{R}^{d_s\times N_u}$ and personality embedding matrix $P \in \mathbb{R}^{d_p\times N_u}$ containing the respective embedding vectors of user u_i in their i^{th} columns. The matrices are then mean-centered and standardized across all user columns. We call these new matrices as X_1 and X_2, respectively. Let the correlation matrix for X_1 be $R_{11} = X_1{X_1}^T \in \mathbb{R}^{d_s\times d_s}$, for X_2 be $R_{22} = X_2{X_2}^T \in \mathbb{R}^{d_p\times d_p}$ and the cross-correlation matrix between them be $R_{12} = X_1{X_2}^T \in \mathbb{R}^{d_s\times d_p}$. For each user u_i, the objective of CCA is to find the linear projections of both embedding vectors that have a maximum correlation. We create K such projections, i.e., K-canonical variate pairs such that each pair of projection is orthogonal with respect to the previous pairs. This is done by constructing:

$$W = X_1^T A_1 \text{ and } Z = X_2^T A_2 \tag{8}$$

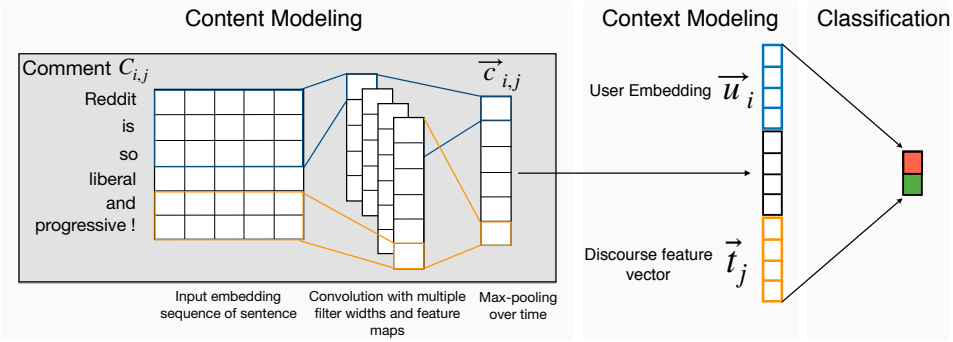


Figure 2: Overall hybrid network of CASCADE. For the comment $C_{i,j}$, its content-based sentential representation $\vec{c}_{i,j}$ is extracted using a CNN and appended with context vectors $\vec{u}_i$ and $\vec{t}_j$.

where, $A_1 \in \mathbb{R}^{d_s \times K}$, $A_2 \in \mathbb{R}^{d_p \times K}$ and $W^T W = Z^T Z = I$. To maximize correlation between W and Z, optimal A_1 and A_2 are calculated by performing singular value decomposition as:

$$R_{11}^{-\frac{1}{2}} R_{12} R_{22}^{-\frac{1}{2}} = A \Lambda B^\top \quad \text{, where} \quad A_1 = R_{11}^{-\frac{1}{2}} A \text{ and } A_2 = R_{22}^{-\frac{1}{2}} B \tag{9}$$

It can be seen that,

$$W^T W = {A_1}^T R_{11} A_1 = A^T A = I \quad \text{and} \quad Z^T Z = {A_2}^T R_{22} A_2 = B^T B = I \tag{10}$$

$$\text{also,} \quad W^T Z = Z^T W = \Lambda \tag{11}$$

Once optimal A_1 and A_2 are calculated, overall user embedding $\vec{u}_i \in \mathbb{R}^K$ of user u_i is generated by fusion of $\vec{d}_i$ and $\vec{p}_i$ as:

$$\vec{u}_i = (\vec{d}_i)^T A_1 + (\vec{p}_i)^T A_2 \tag{12}$$

3.5 Discourse Features

Similarly to how a user influences the degree of sarcasm in a comment, we assume that the discourse of comments belonging to a certain discussion forum contain contextual information relevant to the sarcasm classification. They embed topical information that selectively incur bias towards degree of sarcasm in the comments of a discussion. For example, comments on political leaders or sports matches are generally more susceptible to sarcasm than natural disasters. Contextual information extracted from the discourse of a discussion can also provide background knowledge or cues about the topic of that discussion.

To extract the discourse features, we take a similar approach of document modeling performed for stylometric features (Section 3.4.1). For all N_t-discussion forums, we compose each forum's document by appending the comments within them. As before, *ParagraphVector* is employed to generate discourse representations for each document. We denote the learnt feature vector of j^{th} forum t_j as $\vec{t}_j \in \mathbb{R}^{d_t}$.

3.6 Final Prediction

Following the extraction of text representation $\vec{c}_{i,j}$ for comment $C_{i,j}$ and retrieval of user embedding $\vec{u}_i$ for author u_i and discourse feature vector $\vec{t}_j$ for discussion forum t_j, we concatenate all three vectors to form the unified text representation $\hat{c}_{i,j} = [\vec{c}_{i,j} \oplus \vec{u}_i \oplus \vec{t}_j]$. Here, $\oplus$ refers to concatenation. The CNN used for extraction of $\vec{c}_{i,j}$ has the same design as the CNN we used to extract personality features described in Section 3.4.2. Finally, $\hat{c}_{i,j}$ is projected to the output layer having two neurons with a softmax activation. This gives a softmax-probability over whether a comment is sarcastic or not. This probability estimate is then used to calculate the categorical cross-entropy which is used as the loss function:

$$Loss = \frac{-1}{N} \sum_{i=1}^{N} \sum_{j=1}^{2} \boldsymbol{y}_{i,j} \, log_2(\hat{\boldsymbol{y}}_{i,j}) \quad \text{, where} \quad \hat{\boldsymbol{y}} = softmax(W_o \hat{\boldsymbol{c}}_{i,j} + \vec{\boldsymbol{b}}_o) \tag{13}$$

Here, N is the number of comments in the training set, y_i is the one-hot vector ground truth of the i^{th} comment and $\hat{\boldsymbol{y}}_{i,j}$ is its predicted probability of belonging to class j.

4 Experimental Results

4.1 Dataset

We perform our experiments on a large-scale self-annotated corpus for sarcasm, SARC[3] (Khodak et al., 2017). This dataset contains more than a million examples of sarcastic/non-sarcastic statements made on Reddit. Reddit comprises of topic-specific discussion forums, also known as subreddits, each titled by a post. In each forum, users communicate either by commenting to the titled post or other's comments, resulting in a tree-like conversation structure. This structure can be unraveled to a linear format, thus creating a discourse of the comments by keeping the topological constraints intact. Each comment is accompanied with its author details and parent comments (if any) which is subsequently used for our contextual processing. It is important to note that almost all comments in SARC are composed of a single sentence. We consider three variants of the SARC dataset in our experiments.

- **Main balanced:** This is the primary dataset which contains a balanced distribution of both sarcastic and non-sarcastic comments. The dataset contains comments from 1246058 users (118940 in training and 56118 in testing set) distributed across 6534 forums (3868 in training and 2666 in testing set).
- **Main imbalanced:** To emulate real-world scenarios where the sarcastic comments are typically fewer than non-sarcastic ones, we use an imbalanced version of the Main dataset. Specifically, we maintain a $20:80$ ratio (approx.) between the sarcastic and non-sarcastic comments in both training/testing sets.
- **Pol:** To further test the effectiveness of our user embeddings, we perform experiments on a subset of Main, comprising of forums associated with the topic of politics. Table 1 provides the comment distribution of all the dataset variants mentioned.

		Training set				Testing set			
		no. of comments		avg. no. of words per comment		no. of comments		avg. no. of words per comment	
		non-sarc	*sarc*	*non-sarc*	*sarc*	*non-sarc*	*sarc*	*non-sarc*	*sarc*
Main	balanced	77351	77351	55.13	55.08	32333	32333	55.55	55.01
	imbalanced	77351	25784	55.13	55.21	32333	10778	55.55	55.48
Pol	balanced	6834	6834	64.74	62.36	1703	1703	62.99	62.14

*non-sarc: non-sarcastic, sarc: sarcastic

Table 1: Details of comments in SARC.

The choice of using SARC for our experiments comes with multiple reasons. First, this corpus is the first of its kind that was purposely developed to investigate the necessity of contextual information in sarcasm classification. This characteristic aligns well with the main goal of this paper. Second, the large size of the corpus allows for statistically-relevant analyses. Third, the dataset annotations contain a small false-positive rate for sarcastic labels thus providing reliable annotations. Also, its self-annotation scheme rules out the annotation errors induced by third-party annotators. Finally, the corpus structure provides meta-data (e.g., user information) for its comments, which is useful for contextual modeling.

4.2 Training details

We hold out 10% of the training data for validation. Hyper-parameter tuning is performed using this validation set through RandomSearch (Bergstra and Bengio, 2012). To optimize the parameters, Adam optimizer (Kingma and Ba, 2014) is used, starting with an initial learning rate of $1e^{-4}$. The learnable parameters in the network consists of $\theta = \{U_d, D, W_{[1,2,o,s]}, F_{[1,2,3]}, \vec{\mathbf{b}}_{[1,2,o,d]}, b_{[1,2,3]}\}$. Training termination is decided using early stopping technique with a patience of 12. For the batched-modeling of comments in CNNs, each comment is either restricted or padded to 100 words for uniformity. The optimal hyper-parameters are found to be $\{d_s, d_p, d_t, K\} = 100$, $d_{em} = 300$, $k_s = 2$, $M = 128$, and $\alpha = ReLU$.

We manually analyze the effect in validation performance for different sizes of user-embedding dimension K (Figure 3a) and discourse feature vector size d_t (Figure 3b). In both cases, the performance trend suggests the optimal size to be approximately 100.

[3] http://nlp.cs.princeton.edu/SARC

Models	Main				Pol	
	balanced		imbalanced			
	Accuracy	*F1*	*Accuracy*	*F1*	*Accuracy*	*F1*
Bag-of-words	0.63	0.64	0.68	0.76	0.59	0.60
CNN	0.65	0.66	0.69	0.78	0.62	0.63
CNN-SVM (Poria et al., 2016)	0.68	0.68	0.69	0.79	0.65	0.67
CUE-CNN (Amir et al., 2016)	0.70	0.69	0.73	0.81	0.69	0.70
CASCADE (no personality features)	0.68	0.66	0.71	0.80	0.68	0.70
CASCADE	**0.77**†	**0.77**†	**0.79**†	**0.86**†	**0.74**†	**0.75**†
Δ_{SOTA}	$\uparrow$ 7%	$\uparrow$ 8%	$\uparrow$ 6%	$\uparrow$ 5%	$\uparrow$ 5%	$\uparrow$ 5%

†:significantly better than CUE-CNN (Amir et al., 2016).

Table 2: Comparison of CASCADE with state-of-the-art networks and baselines on multiple versions of the SARC dataset. We assert significance when $p < 0.05$ under paired-t test. Results comprise of 10 runs with different initializations. The bottom row shows the absolute difference with respect to the CUE-CNN system.

For modeling the *ParagraphVector*, we use the open-sourced implementation provided by *Gensim*[4]. The CNNs used in the model are implemented using Tensorflow library[5].

4.3 Baseline Models

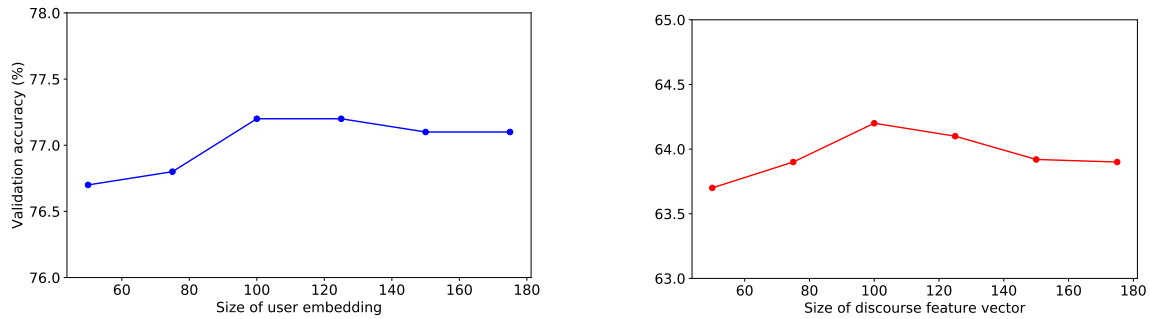
Here, we describe the state-of-the-art methods and baselines that we compare CASCADE with.

- **Bag-of-words:** This model uses an SVM classifier whose input features comprise of a comment's word-counts. The size of the vector is the vocabulary size of the training dataset.
- **CNN:** We compare our model with this individual CNN version. This CNN is capable of modeling only the *content* of a comment. The architecture is similar to the CNN used in CASCADE (see Section 3.2).
- **CNN-SVM:** This model proposed by Poria et al. (2016) consists of a CNN for content modeling and other pre-trained CNNs for extracting sentiment, emotion and personality features from the given comment. All the features are concatenated and fed into an SVM for classification.
- **CUE-CNN:** This method proposed by Amir et al. (2016) also models user embeddings with a method akin to *ParagraphVector*. Their embeddings are then combined with a CNN thus forming the CUE-CNN model. We compare with this model to analyze the efficiency of our embeddings as opposed to theirs. Released software[6] is used to produce results on the SARC dataset.

4.4 Results

Table 2 presents the performance results on SARC. CASCADE manages to achieve major improvement across all datasets with statistical significance. The lowest performance is obtained by the bag-of-words approach whereas all neural architectures outperform it. Amongst the neural networks, the CNN baseline

[4] http://radimrehurek.com/gensim/models/doc2vec.html
[5] http://github.com/dennybritz/cnn-text-classification-tf
[6] http://github.com/samiroid/CUE-CNN



(a) CASCADE with only user embeddings.

(b) CASCADE with only discourse features.

Figure 3: Exploration of dimensions for user embedding and discourse feature vector.

receives the least performance. CASCADE comfortably beats the state-of-the-art neural models CNN-SVM and CUE-CNN. Its improved performance on the Main imbalanced dataset also reflects its robustness towards class imbalance and establishes it as a real-world deployable network.

We further compare our proposed user-profiling method with that of CUE-CNN, with absolute differences shown in the bottom row of Table 2. Since CUE-CNN generates its user embeddings using a method similar to the *ParagraphVector*, we test the importance of personality features being included in our user profiling. As seen in the table, CASCADE without personality features drops in performance to a range similar to CUE-CNN. This suggests that the combination of stylometric and personality features are indeed crucial for the improved performance of CASCADE.

4.5 Ablation Study

We experiment on multiple variants of CASCADE so as to analyze the importance of the various features present in its architecture. Table 3 provides the results of all the combinations. First, we test performance for the *content*-based CNN only (row 1). This setting provides the worst relative performance with almost 10% lower accuracy than optimal. Next, we include contextual features to this network. Here, the effect of discourse features is primarily seen in the Pol dataset getting an increase of 3% in F1 (row 2). A major boost in performance is observed ($8 - 12\%$ accuracy and F1) when user embeddings are introduced (row 5). Visualization of the user embedding cluster (Section 4.6) provides insights for this positive trend. Overall, CASCADE consisting of CNN with user embeddings and contextual discourse features provides the best performance in all three datasets (row 6).

We challenge the use of CCA for the generation of user embeddings and, hence, replace it with simple concatenation. This, however, causes a significant drop in performance (row 3). Improvement is not observed even when discourse features are used with these concatenated user embeddings (row 4). We assume the increase in parameters caused by concatenation for this performance degradation. CCA, on the other hand, creates succinct representations with maximal information, giving better results.

4.6 User Embedding Analysis

We investigate the learnt user embeddings in more detail. In particular, we plot random samples of users on a 2D-plane using t-SNE (Maaten and Hinton, 2008). The users who have greater sarcastic comments (atleast 2 more than the other type) are termed as sarcastic users (colored red). Conversely, the users having fewer sarcastic comments are called non-sarcastic users (colored green). Equal number of users from both the categories are plotted. We aim to analyze the reason behind the performance boost provided by the user embeddings as shown in Table 3. We see in Figure 4 that both the user types belong to similar distributions. However, the sarcastic users have a greater spread than the non-sarcastic ones (red belt around the green region). This is also evident from the variances of the distributions where the sarcastic distribution comprises of 10.92 variance as opposed to 5.20 variance of the non-sarcastic distribution. From this observation, we can infer that the user embeddings belonging to this non-overlapping red-region provide discriminative information regarding the sarcastic tendencies of their users.

	CASCADE			Main				Pol	
	user		dis-	balanced		imbalanced			
	cca	concat.	course	*Acc.*	*F1*	*Acc.*	*F1*	*Acc.*	*F1*
1.	-	-	-	0.65	0.66	0.69	0.78	0.62	0.63
2.	-	-	✓	0.66	0.66	0.68	0.78	0.63	0.66
3.	-	✓	-	0.66	0.66	0.69	0.79	0.62	0.61
4.	-	✓	✓	0.65	0.67	0.71	0.85	0.63	0.66
5.	✓	-	-	0.77	0.76	**0.80**	**0.86**	0.70	0.70
6.	✓	-	✓	**0.78**	**0.77**	0.79	**0.86**	**0.74**	**0.75**

Table 3: Comparison with variants of the proposed CASCADE network. All combinations use content-based CNN

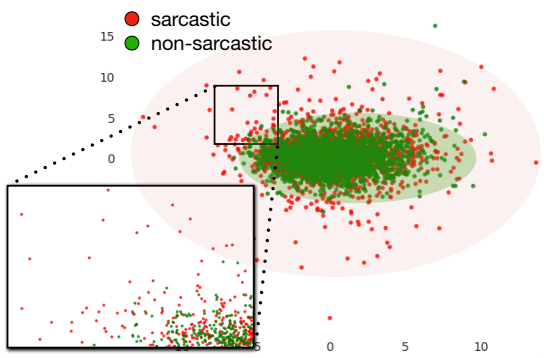


Figure 4: 2D-Scatterplot of the user embeddings visualized using t-SNE (Maaten and Hinton, 2008).

4.7 Case Studies

Results demonstrate that discourse features provide an improvement over baselines, especially on the Pol dataset. This signifies the greater role of the contextual cues for classifying comments in this dataset over the other dataset variants used in our experiment. Below, we present a couple of cases from the Pol dataset where our model correctly identifies the sarcasm which is evident only with the neighboring comments. The previous state-of-the-art CUE-CNN, however, misclassifies them.

- For the comment *Whew, I feel much better now!*, its sarcasm is evident only when its previous comment is seen *So all of the US presidents are terrorists for the last 5 years.*
- The comment *The part where Obama signed it.* doesn't seem to be sarcastic until looked upon as a remark to its previous comment *What part of this would be unconstitutional?*.

Such observations indicate the impact of discourse features. However, sometimes contextual cues from the previous comments are not enough and misclassifications are observed due to lack of necessary commonsense and background knowledge about the topic of discussion. There are also other cases where our model fails despite the presence of contextual information from the previous comments. During exploration, this is primarily observed for contextual comments which are very long. Thus, sequential discourse modeling using RNNs may be better suited for such cases. Also, in the case of user embeddings, misclassifications were common for users with fewer historical posts. In such scenarios, potential solutions would be to create user networks and derive information from similar users within the network, e.g., by means of community embeddings (Cavallari et al., 2017). These are some of the issues which we plan to address in future work.

5 Conclusion

In this paper, we introduced CASCADE, a Contextual Sarcasm Detector, which leverages both content and contextual information for the classification. For contextual details, we perform user profiling along with discourse modeling from comments in discussion threads. When this information is used jointly with a CNN-based textual model, we obtain state-of-the-art performance on a large-scale Reddit corpus. Our results show that discourse features along with user embeddings play a crucial role in the performance of sarcasm detection.

References

Silvio Amir, Byron C Wallace, Hao Lyu, and Paula Carvalho Mário J Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. *arXiv preprint arXiv:1607.00976*.

Adrian Benton, Raman Arora, and Mark Dredze. 2016. Learning multiview embeddings of twitter users. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 14–19.

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305.

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

Erik Cambria, Soujanya Poria, Alexander Gelbukh, and Mike Thelwall. 2017. Sentiment analysis is a big suitcase. *IEEE Intelligent Systems*, 32(6):74–80.

Paula Carvalho, Luís Sarmento, Mário J Silva, and Eugénio De Oliveira. 2009. Clues for detecting irony in user-generated contents: oh...!! it's so easy;-. In *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, pages 53–56. ACM.

Sandro Cavallari, Vincent Zheng, Hongyun Cai, Kevin Chang, and Erik Cambria. 2017. Learning community embedding with community detection and node embedding on graphs. In *CIKM*, pages 377–386.

Na Cheng, Rajarathnam Chandramouli, and KP Subbalakshmi. 2011. Author gender identification from text. *Digital Investigation*, 8(1):78–88.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in twitter and amazon. In *Proceedings of the fourteenth conference on computational natural language learning*, pages 107–116. Association for Computational Linguistics.

Roberto González-Ibánez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers-Volume 2*, pages 581–586. Association for Computational Linguistics.

David R Hardoon, Sandor Szedmak, and John Shawe-Taylor. 2004. Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, 16(12):2639–2664.

Harold Hotelling. 1936. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 757–762.

Aditya Joshi, Pushpak Bhattacharyya, and Mark J Carman. 2017. Automatic sarcasm detection: A survey. *ACM Computing Surveys (CSUR)*, 50(5):73.

Anupam Khattri, Aditya Joshi, Pushpak Bhattacharyya, and Mark Carman. 2015. Your sentiment precedes you: Using an author's historical tweets to predict sarcasm. In *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 25–30.

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2017. A large self-annotated corpus for sarcasm. *arXiv preprint arXiv:1704.05579*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Roger J Kreuz and Gina M Caucci. 2007. Lexical influences on the perception of sarcasm. In *Proceedings of the Workshop on computational approaches to Figurative Language*, pages 1–4. Association for Computational Linguistics.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Navonil Majumder, Soujanya Poria, Alexander Gelbukh, and Erik Cambria. 2017. Deep learning-based document modeling for personality detection from text. *IEEE Intelligent Systems*, 32(2):74–79.

Gerald Matthews and Kirby Gilliland. 1999. The personality theories of hj eysenck and ja gray: A comparative review. *Personality and Individual differences*, 26(4):583–626.

Gerald Matthews, Ian J Deary, and Martha C Whiteman. 2003. *Personality traits*. Cambridge University Press.

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

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, pages 246–252. Citeseer.

Soujanya Poria, Erik Cambria, Devamanyu Hazarika, and Prateek Vij. 2016. A deeper look into sarcastic tweets using deep convolutional neural networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1601–1612.

Ashwin Rajadesingan, Reza Zafarani, and Huan Liu. 2015. Sarcasm detection on twitter: A behavioral modeling approach. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 97–106. ACM.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 704–714.

Ranjan Satapathy, Claudia Guerreiro, Iti Chaturvedi, and Erik Cambria. 2017. Phonetic-based microtext normalization for twitter sentiment analysis. In *ICDM*, pages 407–413.

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the Association for Information Science and Technology*, 60(3):538–556.

Joseph Tepperman, David Traum, and Shrikanth Narayanan. 2006. " yeah right": Sarcasm recognition for spoken dialogue systems. In *Ninth International Conference on Spoken Language Processing*.

Oren Tsur, Dmitry Davidov, and Ari Rappoport. 2010. Icwsm-a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In *ICWSM*, pages 162–169.

Michel van de Velden. 2011. On generalized canonical correlation analysis. In *Proceedings of the 58th World Statistical Congress*.

Byron C Wallace, Laura Kertz, Eugene Charniak, et al. 2014. Humans require context to infer ironic intent (so computers probably do, too). In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 512–516.

Byron C Wallace, Eugene Charniak, et al. 2015. Sparse, contextually informed models for irony detection: Exploiting user communities, entities and sentiment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1035–1044.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Tweet sarcasm detection using deep neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2449–2460.

A Generalized Canonical Correlation Analysis

For user profiling with more than two views, we can use *Generalized* CCA (GCCA) as the multiview-fusion approach. In GCCA, the input data consists of I different views, $X_i \in \mathbb{R}^{d_i \times N} \quad \forall\ i \in [1, I]$, where, N is the total number of data points and d_i is the dimension of the ith view. Also, X_i represent the mean centered matrix of the data. We find a common representation $G \in \mathbb{R}^{N \times K}$ for all the input points. The *canonical covariates* $\vec{\boldsymbol{w}}_{\boldsymbol{i}} = X_i^T \vec{\boldsymbol{a}}_{\boldsymbol{i}}$ are chosen in such a way that the sum of the squared correlations between them and the group configuration is maximum:

$$\max R^2 = \sum_{i=1}^{N} r(\vec{\boldsymbol{g}}, X_i^T \vec{\boldsymbol{a}}_{\boldsymbol{i}})^2 \quad \text{s.t. } \vec{\boldsymbol{g}}^T \vec{\boldsymbol{g}} = 1 \tag{14}$$

For K-canonical variate pairs, the GCCA objective function can be formulated as follows:

$$\underset{G, A_i}{\operatorname{argmax}} \left\| G - X_i^T A_i \right\|_F^2 \quad \text{s.t. } G^T G = I \tag{15}$$

where $A_i \in \mathbb{R}^{d_i \times K}$. G can be obtained using the eigen equation:

$$(\sum_{i=1}^{N} P_i) G = G\Gamma \quad \text{, where,} \quad P_i = X_i^T (X_i X_i^T)^{-1} X_i \tag{16}$$

The matrices A_i can then be calculated as:

$$A_i = (X_i X_i^T)^{-1} X_i^T G \tag{17}$$

It is to be noted that GCCA with two views is equivalent to CCA (van de Velden, 2011).