A Transformer and Prototype-based Interpretable Model for Contextual Sarcasm Detection

Ximing Wen
Drexel University,
Philadelphia, USA
xw384@drexel.edu

Rezvaneh Rezapour
Drexel University,
Philadelphia, USA
sr3563@drexel.edu

Abstract

Sarcasm detection, with its figurative nature, poses unique challenges for affective systems designed to perform sentiment analysis. While these systems typically perform well at identifying direct expressions of emotion, they struggle with sarcasm's inherent contradiction between literal and intended sentiment. Since transformer-based language models (LMs) are known for their efficient ability to capture contextual meanings, we propose a method that leverages LMs and prototype-based networks, enhanced by sentiment embeddings, to conduct interpretable sarcasm detection. Our approach is intrinsically interpretable without extra post-hoc interpretability techniques. We test our model on three public benchmark datasets and show that our model outperforms the current state-of-the-art. At the same time, the prototypical layer enhances the model's inherent interpretability by generating explanations through similar examples in the reference time. Furthermore, we demonstrate the effectiveness of incongruity loss in the ablation study, which we construct using sentiment prototypes.

1 Introduction

The task of automatically detecting sarcasm introduces a complex challenge in natural language processing (NLP). This nuanced task bridges the gap between sentiment analysis and text interpretation, highlighting the complexity of understanding and analyzing sarcasm in written language (Ilavarasan et al., 2020). Sarcasm, characterized by a sharp, often humorous contrast between literal and intended meanings of statements, poses unique difficulties for computational models. These challenges stem from sarcasm's deep reliance on contextual clues, tone, and common human experiences. This complexity is further amplified in digital communication, where non-verbal cues are largely absent, making it essential to develop advanced models capable of interpreting such subtleties with high accuracy.

Deep learning models, especially transformer-based language models (LMs), have significantly contributed to advancements in NLP, offering powerful tools for sentiment analysis (Bu et al., 2024), emotion detection (Tu et al., 2024), and, by extension, sarcasm detection (Helal et al., 2024). More specifically, to study sarcasm detection, there is a trend to leverage LMs and a variety of features generated by different models to improve prediction accuracy (Cai et al., 2019; Bedi et al., 2021). However, a persistent critique of deep learning models is their "black-box" nature, which obscures the decision-making process and hinders their interpretability. Current approaches usually adopt post-hoc interpretability methods Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al., 2016), and SHapley Additive exPlanations (SHAP) (Lundberg and Lee, 2017) or attention mechanisms to explain a model's decision (Ribeiro et al., 2016; Mardaoui and Garreau, 2021; Kumar et al., 2021b). However, those explanations are still word-level and can only tell which part of the input they are looking at. As for sarcasm detection, when the text does not contain words that convey strong sentiment and instead uses other ways, such as analogy to express sarcasm, the word-level explanations could set similar weights to words in the sentence, and humans are ill-equipped to interpret them.

To address this challenge, our paper presents an intrinsically interpretable NLP framework that integrates prototype classification networks (Li et al., 2018) with multi-view of semantic embedding and sentiment embedding from large-scale pre-trained transformer language models. To the best of our knowledge, our study is the first to apply a prototype-based network in sarcasm detection. This is achieved through a unique training regimen that enables the network to learn a collection of pro-

The Proceedings for the 15th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis (WASSA 2026), pages 278–288

totype tensors, which encapsulate latent clusters of training samples. At the point of inference, the model makes classification judgments solely based on the similarity to these prototypes, allowing for the model's decisions to be transparently explained by referencing the training examples most closely aligned with the top-matched prototypes. Together with a sentiment-prototype-based incongruity loss that captures the difference between implicit and explicit sentiment, our approach not only provides clear, human-understandable explanations for its predictions but also achieves state-of-the-art performance. The key contributions of our methods can be summarized as follows:

1. We propose a novel interpretable framework for sarcasm detection. Our framework is built upon a prototype-based network leveraging semantic embedding and sentiment embedding from pre-trained transformer-based language models.

2. Extensive experiments on three public benchmark datasets show that our approach achieves state-of-the-art performance while being interpretable. We also conduct an ablation study to analyze the influence of incongruity loss in our model.

3. We conduct case studies to show that our model can generate human-readable, sentence-level explanations for the model's reasoning process at the reference time. Our model and training code are available here: `https://github.com/social-nlp-lab/Sarcasm-Detection`.

2 Related Work

2.1 Contextual Sarcasm Detection

Initial research in sarcasm detection primarily relied on simple lexical and syntactic features, and the classifiers are categorized as **Content-based Models** (Carvalho et al., 2009; Davidov et al., 2010; González-Ibáñez et al., 2011), leveraging features like n-grams, and part-of-speech tags (Riloff et al., 2013; Tepperman et al., 2006; Tsur et al., 2010).

With the increase in the usage of sarcasm on online platforms in recent years, the performance of the sarcasm detection model is usually compromised in terms of robustness when faced with texts plagued by grammatical inaccuracies (Švelch, 2015). Moreover, these texts are usually a series of posts and comments that are highly temporal and contextual. As a result, relying solely on linguistic cues has become inadequate, prompting researchers to develop **Context-based Models**. A prominent strand of this research involves mining sentiment incongruity in sarcastic texts with attention-machenism (Pan et al., 2020; Najafabadi et al., 2024) to improve models' performance. Diverging from these methods, our approach adopts sentiment prototypes to discern both implicit and explicit sentiments within texts, enhancing the interpretability of the reasoning process. Furthermore, other scholars are exploring the modeling of user interactions via Graph Convolutional Networks (GCN) (Mohan et al., 2023) or employing commonsense knowledge transformers like COMET (Yu et al., 2023).

2.2 Explainability of Transformer Language Models

Post hoc Interpretability: When leveraging LM's ability to understand context, the model's complexity prevents people from understanding the model's reasoning process. In the field of XAI, post-hoc explainable approaches, such as LIME (Ribeiro et al., 2016), and SHAP (Lundberg and Lee, 2017), are used to generate explanations for transformers' results (Kumar et al., 2021a) through analyzing weights for each input word representation. However, these methods are now facing the challenge of being faithful and robust. Various studies show the sensitivity of the models to small perturbations in the target model's inputs or parameters (Ivankay et al., 2022; Mardaoui and Garreau, 2021). Other researchers, such as Akula and Garibay (2021) and Rishabh Misra and Prahal Arora (2018), have explored the use of attention mechanisms to interpret prediction outcomes, focusing on how attention scores are allocated across individual words. However, these explanations remain at the word level, indicating only which parts of the input the model attends to. In the context of sarcasm detection, where sarcasm may be conveyed through mechanisms such as analogy rather than explicit sentiment, word-level explanations can assign similar weights to different words in a sentence. This makes it challenging for humans to interpret the underlying sarcastic intent effectively. To address this, we aim for more intuitive and sparse explanations: well-descriptive but short-sequence prototypes.

Prototype-based Reasoning in Deep Neural Networks: Prototype-based methods emphasize that visualizing the reasoning process through prototypes can significantly improve the intuitiveness of interpretation. This approach, leveraging prototype-based reasoning, has been a core aspect of interpretability in classical models for decades, as evidenced by research from Cupello and Mishelevich (1988), Fikes and Kehler (1985), and Kim et al. (2015). A pioneering example of incorporating prototypical learning into deep neural networks is the work by Chen et al. (2019), who introduced a novel neural network design for image classification. By inserting a prototype layer following the convolutional layers, the model compares convolution responses across different locations in the predicted image with predefined prototypes. Furthermore, this allows users to grasp why an image is classified in a certain way, such as understanding why a bird is identified as a 'red-bellied woodpecker' due to its distinct red belly and head, along with black and white wing stripes. Following this work, researchers explored incorporating prototype layer with transformer-based encoders, such as Universal Sentence Encoder, BERT, BART (Bidirectional and Auto-Regressive Transformers) in fake news detection and hotel review classification (Das et al., 2022; Hong et al., 2024; Wen, 2024). Sarcasm, due to its nature, can benefit from such reasoning provided by prototype-based models. However, this approach is still underexplored.

3 Methodology

Our approach, as shown in Figure 1, first encodes semantic embedding and sentiment embedding with the Semantic View encoder and the Sentiment View encoder separately, and then both the encoded semantic embedding and sentiment embedding are fed into two separate prototype layers. Finally, the output distance vectors from the two prototype layers are concatenated and sent to the output layer to make the prediction.

3.1 Semantic View

The Semantic View focuses on capturing contextual meanings from text input. We use pre-trained language models (LMs) from Hugging Face (Wolf et al., 2020) as encoders to extract sentence-level embeddings. When the input comment has ancestor posts (*e.g.*, SARC Dataset by Khodak et al. (2018)), which help set up the stage for the conversation and provide more context information, we concatenate the nearest ancestor with the comment as a whole and encode the embeddings with a pre-trained language model (*e.g.* Sentence-BERT (SBERT) by Reimers and Gurevych, 2019, RoBERTa by Liu et al., 2019b).

$$\mathbf{e}^{ct} = \mathbf{Encoder}(\mathbf{x}) \tag{1}$$

where x denotes the input text, $\mathbf{e^{ct}}$ is the semantic information representation vector.

3.2 Sentiment View

For sentiment feature extraction, we decompose the text into two parts: **Explicit Part**, which are sentiment words extracted following (Joshi et al., 2015), and **Implicit Part**, which is the rest of the text capturing the implicit sentiments.

Both segments are processed through the Sentiment Encoder, specifically the SiEBERT model (Hartmann et al., 2023), to separately obtain the vector representations $\mathbf{e}^{st,ep}$ and $\mathbf{e}^{st,ip}$ of the CLS token from the final hidden state. For brevity, we use $\mathbf{e}^{st}$ to represent both of them in the following equation:

$$\mathbf{e}^{st} = \mathbf{SiEBERT}(\mathbf{x}) \tag{2}$$

where $\mathbf{x}$ denotes the input text, $\mathbf{e}^{st}$ is the sentiment information representation vector. The explicit representation $\mathbf{e}^{st,ep}$ is labeled as $\mathbf{z}^{st,ep}$ with SiEBERT. The implicit representation $\mathbf{e}^{st,ip}$ is labeled as $\mathbf{z}^{st,ip}$ identically to $\mathbf{z}^{st,ep}$ for non-sarcastic inputs, whereas it is labeled oppositely for sarcastic inputs.

Fallback Strategy The implementation incorporates a fallback strategy for cases where no explicit sentiment cues are identified in the input text. Specifically, when the sentiment word detector returns an empty set, a predefined neutral-to-positive sentiment token sequence is used as a surrogate explicit sentiment representation. This surrogate input is encoded using SiEBERT following the same preprocessing and inference pipeline applied to detected sentiment words, ensuring architectural consistency.

3.3 Prototypical Layer

As shown in Figure 1, after the input is encoded into a latent semantic representation $\mathbf{e}^{ct} \in \mathbb{R}^{d_s}$ through Semantic View and two latent sentiment representation $\mathbf{e}^{st,ep} \in \mathbb{R}^{d_m}$, $\mathbf{e}^{st,ip} \in \mathbb{R}^{d_m}$ through

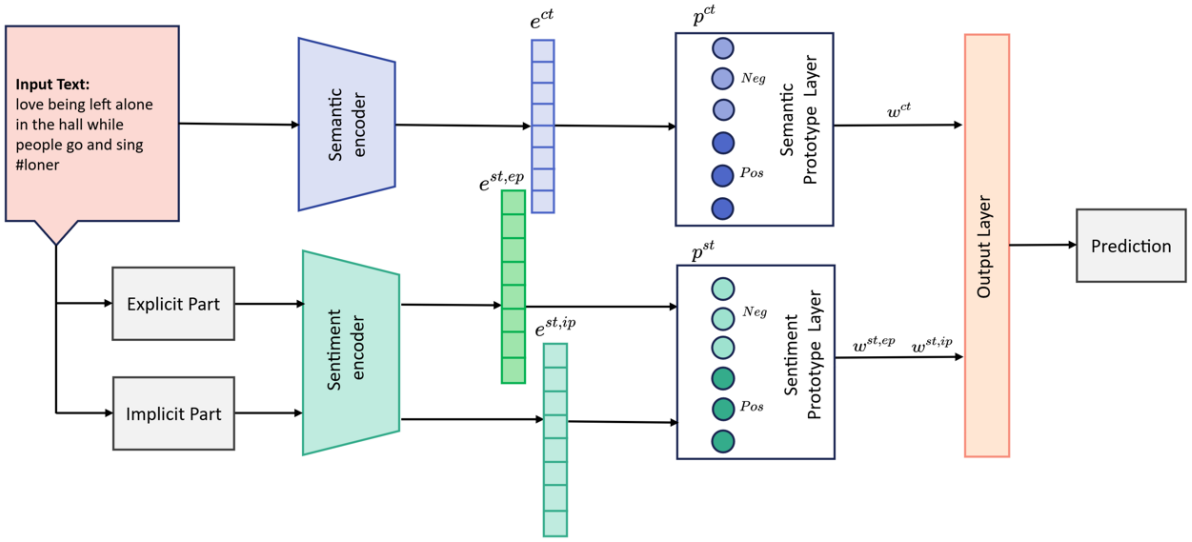

Figure 1: Diagram of the proposed model architecture and workflow.

Sentiment View separately, each representation is fed into a prototype layer respectively.

Semantic Prototype: The Semantic Prototype Layer consists of k_a prototypes $\mathbf{P}^{ct} = \{\mathbf{p}_j^{ct}\}_{j=1}^{k_a}$, where $\mathbf{p}_j^{ct} \in \mathbb{R}^{d_s}$, the same dimension of the encoded latent semantic feature $\mathbf{e}^{ct}$. Each prototype, represented as a tensor, encapsulates a cluster of training examples. To ensure that both sarcastic and non-sarcastic classes are effectively represented by the learned prototypes, we allocate a fixed number of prototypes to each class. Utilizing k-means clustering (Hartigan and Wong, 1979), we segment the training data of each class into multiple clusters and subsequently initialize the prototypes for each class with these cluster centers. These prototypes are trained through loss terms $\mathcal{L}_{cls}^{ct}$ and $\mathcal{L}_{sep}^{ct}$ defined in §3.5. This layer calculates the similarity between embedding $\mathbf{e}^{ct}$ and each prototype $\mathbf{p}_j^{ct}$ with a Radial basis function (RDF) kernel function as follows:

$$sim(\mathbf{e}^{ct}, \mathbf{p}_j^{ct}) = exp\left(-\frac{\|\mathbf{e}^{ct} - \mathbf{p}_j^{ct}\|_2^2 + \varepsilon}{\sigma^2}\right) \quad (3)$$

This similarity score increases monotonically as the Euclidean distance increases. σ is a user-specified value that determines how quickly the similarity score increases as the distance between $\mathbf{e}^{ct}$ and $\mathbf{p}_j^{ct}$ decreases. A small σ makes the kernel function more sensitive to changes in distance, leading to a kernel matrix with more localized information about the data points. This can make the model more sensitive to noise in the data. A large σ, on the other hand, produces a smoother kernel function that is less sensitive to the exact distance between data points, potentially making the model more robust. We also add a small value ε set as $1e-4$ for numerical stability. We get vector $\mathbf{w}^{ct}$ by calculating the similarity score between $\mathbf{e}^{ct}$ and each prototype vector within $\mathbf{P}^{ct}$.

Sentiment Prototype Layer: The Sentiment Prototype Layer comprises k_b prototypes $\mathbf{P}^{st} = \{\mathbf{p}_j^{st}\}_{j=1}^{k_b}$, where $\mathbf{p}_j^{st} \in \mathbb{R}^{d_m}$. These prototypes are categorized into positive prototypes where $\mathbf{p}^{st} \in \mathbf{P}_1^{st}$ and negative prototypes where $\mathbf{p}^{st} \in \mathbf{P}_0^{st}$. We initialize $\mathbf{P}_1^{st}$ using the k-means cluster centers computed from the positive training data labeled by SiEBERT and $\mathbf{P}_0^{st}$ using those derived from the negative training data. When clustering, we only use non-sarcastic training samples, without dividing the text into implicit and explicit parts.

We calculate the similarity between embedding $\mathbf{e}^{st}$ and each prototype $\mathbf{p}_j^{st}$ as follows:

$$sim(\mathbf{e}^{st}, \mathbf{p}_j^{st}) = exp\left(-\frac{\|\mathbf{e}^{st} - \mathbf{p}_j^{st}\|_2^2 + \varepsilon}{\sigma^2}\right) \quad (4)$$

We calculate similarity scores for the explicit and implicit representations, $\mathbf{e}^{st,ep}$ and $\mathbf{e}^{st,ip}$, against each prototype vector within $\mathbf{P}^{st}$, yielding the similarity vectors $\mathbf{w}^{st,ep}$ and $\mathbf{w}^{st,ip}$, respectively.

3.4 Output Layer

The output layer is a fully connected layer followed by a sigmoid layer. It takes the concatenation of the

extracted similarity vectors $\mathbf{w}^{ct}$, $\mathbf{w}^{st,ep}$ and $\mathbf{w}^{st,ip}$ from prototype layers as input and predict the likelihood of a text being sarcastic.

We intentionally employ concatenation to fuse these representations in order to preserve the independence of prototype similarity vectors, which is essential for interpretability in prototype-based models, as it allows direct attribution of the final prediction to individual semantic or sentiment prototypes. More complex fusion mechanisms, such as cross-attention or gating, may introduce stronger interactions but would entangle prototype signals and obscure their individual contributions. Interactions between explicit and implicit sentiment are therefore not modeled at the fusion layer; instead, they are explicitly captured by the incongruity loss (Eq. 12), which encourages disagreement between explicit and implicit sentiment predictions for sarcastic inputs.

3.5 Loss and Training Algorithm

We construct the loss function with four different terms to ensure both accuracy and interpretability.

Accuracy Loss: The first term is accuracy loss, and it uses cross-entropy loss to optimize the predictive power of the network. In equation 5, f is the output classifier, $\mathcal{P}_i$ is the predicted probability distribution, y_i is the label, and n is the total number of training data points. θ refers to the trainable weights in the classifier.

$$\mathcal{P}_i = f([\mathbf{w}_i^{ct}, \mathbf{w}_i^{st,ep}, \mathbf{w}_i^{st,ip}]) \tag{5}$$

$$\mathcal{L}_{acc} = -\frac{1}{n}\sum_{i=1}^{n} \mathbf{y}_i \log P(\mathcal{P}_i = \mathbf{y}_i | \mathbf{x_i}; \boldsymbol{\theta}) \tag{6}$$

Division Loss: To distribute prototypes in the embedding space as much as possible, we design the $\mathcal{L}_{div}$ indicated in equation 7 for both semantic and sentiment prototypes. This loss uses cosine similarity to measure the difference between any two prototypes $\mathbf{p}_m$ and $\mathbf{p}_n$ in P and penalizes it if their similarity is larger than λ. It is particularly beneficial when there are multiple prototypes for a single class, as it promotes the representation of diverse aspects of that class.

$$\mathcal{L}_{div} = \sum_{\mathbf{p}_j, \mathbf{p}_q \in \mathbf{P}, j \neq q} max(0, cos(\mathbf{p}_j, \mathbf{p}_q) - \lambda) \tag{7}$$

Clustering and Seperation Loss: The clustering loss $\mathcal{L}_{cls}$ and $\mathcal{L}_{sep}$ are inspired by previous work, ProtoPNet (Chen et al., 2019). The clustering loss $\mathcal{L}_{cls}$ ensures each embedding is close to at least one prototype in its own class, and separation loss $\mathcal{L}_{sep}$ encourages each embedding to be distant from prototypes not of its class. Together, $\mathcal{L}_{cls}$ and $\mathcal{L}_{sep}$ push each prototype to focus more on training examples from the same class and less on training examples from other classes.

For **semantic prototypes**, they are defined as follows:

$$\mathcal{L}_{cls}^{ct} = \frac{1}{n}\sum_{i=1}^{n} \min_{\mathbf{p}_j^{ct} \in \mathbf{P}_{y_i}^{ct}} \|\mathbf{e}_i^{ct} - \mathbf{p}_j^{ct}\|_2^2 \tag{8}$$

$$\mathcal{L}_{sep}^{ct} = -\frac{1}{n}\sum_{i=1}^{n} \min_{\mathbf{p}_j^{ct} \notin \mathbf{P}_{y_i}^{ct}} \|\mathbf{e}_i^{ct} - \mathbf{p}_j^{ct}\|_2^2 \tag{9}$$

For **sentiment prototypes**, we promote proximity between positive prototypes and training data segments labeled positive ($z_i = 1$) by SiEBERT, and likewise align negative prototypes with training data segments labeled negative ($z_i = 0$) through losses defined as follows:

$$\mathcal{L}_{cls}^{st} = \frac{1}{n}\sum_{i=1}^{n} \min_{\mathbf{p}_j^{st} \in \mathbf{P}_{z_i}^{st}} \|\mathbf{e}_i^{st} - \mathbf{p}_j^{st}\|_2^2 \tag{10}$$

$$\mathcal{L}_{sep}^{st} = -\frac{1}{n}\sum_{i=1}^{n} \min_{\mathbf{p}_j^{st} \notin \mathbf{P}_{z_i}^{st}} \|\mathbf{e}_i^{st} - \mathbf{p}_j^{st}\|_2^2 \tag{11}$$

The final Clustering and Separation Loss is $\mathcal{L}_{cls\_sep} = \mathcal{L}_{cls}^{ct} + \mathcal{L}_{sep}^{ct} + \mathcal{L}_{cls}^{st} + \mathcal{L}_{sep}^{st}$

Incongruity Loss: We hypothesize the presence of incongruity between the explicit and implicit sentiments within a sarcastic comment. For instance, in the sarcastically labeled sentence, "Oh no, a rainy day again! This is great!" the explicit sentiment conveyed by the word "great" appears positive. However, upon closer examination of the context, it becomes evident that the speaker does not favor rainy days, revealing an underlying negative sentiment. Based on this observation, we introduce the incongruity loss defined with cross-entropy to effectively capture this disparity between explicit and implicit sentiment:

$$\mathcal{L}_{inco} = -\frac{1}{n}\sum_{i=1}^{n}\Big(\mathbf{z}_i^{ep}\log P(h(\mathbf{w}_i^{st,ep}) = \mathbf{z}_i^{ep}|\boldsymbol{\theta}) + \mathbf{z}_i^{ip}\log P(h(\mathbf{w}_i^{st,ip}) = \mathbf{z}_i^{ip}|\boldsymbol{\theta})\Big) \tag{12}$$

where h is a MLP based classifier that predicts a probability distribution.

We construct the final loss function L by combining the previously defined loss components, each weighted by their respective coefficient λ. Additionally, we incorporate an L1 regularization term $\|\theta\|$ to promote sparsity in the weights of the output layer. The final loss $\mathcal{L}$ is defined as follows:

$$\mathcal{L} = \mathcal{L}_{acc} + \lambda_1\mathcal{L}_{div} + \lambda_2\mathcal{L}_{cls\_sep} + \lambda_3\mathcal{L}_{inco} + \lambda_4\|\theta\| \tag{13}$$

3.6 Prototype Projection

For improved interpretability, we visualize the semantic prototypes by projecting a prototype vector onto its closest datapoint in the training dataset, measured by Euclidean distance. For the large training dataset with over 120k comments, we did a random pre-sampling of one-tenth of the comments for computation efficiency. Each prototype's embedding is replaced with the nearest comment's embedding in the training data point. The alignment of prototypes with training set samples provides an intuitive and easily understandable interpretation for humans.

3.7 Prototype Initialization

We use k-means cluster centers on the training data to initialize prototypes, with a fixed small number of prototypes per class (20). We observed the model to be robust to prototype counts within a reasonable range; extremely small counts reduce coverage, and extremely large counts increase redundancy but yield similar performance due to the division loss.

4 Experimental Setup

In this section, we introduce the datasets used in the experiment and the baseline models.

4.1 Data

We evaluate our methods on the following three public benchmark datasets: (1) **SARC 2.0** [1] (Khodak et al., 2018), a corpus comprising 1.3 million comments on Reddit. Each comment is self-annotated, and we focus on the primary main balanced variation of it, with 118,940 comments in training and 56,118 comments in the testing set. (2) **Twitter** (Riloff et al., 2013) dataset is sourced from the Twitter platform. Sarcastic tweets are identified using the hashtag #sarcasm, while non-sarcastic tweets lack this hashtag. The dataset consists of 1,368 training examples and 588 test examples. (3) **Sarcasm Corpus V2 Dialogues** (Oraby et al., 2017) is a diverse and richly annotated corpus of sarcasm in dialogue. This dataset is collected from a variety of dialogue sources to capture sarcasm in different conversational contexts, moving beyond traditional social media platforms like Twitter.

[1]Link: https://nlp.cs.princeton.edu/old/SARC/2.0/

4.2 Models and Settings

We employed 5-fold cross-validation to evaluate our model's performance and fine-tuned the hyperparameters on the validation data. The Optimizer for all neural networks is Adam, and the learning rate is $1e-4$. We used one single GTX 3090 for each model's training, and due to the limitation of GPU RAM, when training with LM encoders, we chose a batch size of 60 and an accumulated gradient step of 30. We use early stopping (Fomin et al., 2020) based on the loss of validation data.

Semantic Encoder: In prototype-based architectures, it is a common practice to use Euclidean distance to measure sentence similarity between training examples and prototypes. However, embeddings from transformer-based language models are not typically trained with contrastive loss that leverages Euclidean distance for measuring sentence similarity. To further explore whether this influences the model's interpretability, we selected RoBERTa-large and SBERT as the encoders for comparison in our experiments:

(1) RoBERTa-large: RoBERTa-large is a transformer-based language model developed to improve upon the BERT architecture. We used the [CLS] token embedding from the last hidden states directly for the downstream task.

(2) SBERT: SBERT is post-trained on BERT. It uses siamese and triplet network structures to derive semantically meaningful sentence embeddings that can be compared using cosine similarity.

We used the pre-trained model all-mpnet-base-v2 [2] developed by Microsoft from Hugging Face.

Sentiment Encoder: For sentiment feature extraction, we employed the SiEBERT model (Hartmann et al., 2023), which is a fine-tuned version of the RoBERTa-large model optimized for sentiment classification tasks.

Baseline : For comparative evaluation with our approach, we selected the following methodologies as baselines: Fracking Sarcasm (Ghosh and Veale, 2016), GRNN (Zhang et al., 2016), CNN-LSTM-DNN (Ghosh and Veale, 2016), SIRAN (Tay et al., 2018), MIRAN (Tay et al., 2018), ELMo-BiLSTM (Ilic et al., 2018), A2Text-Net (Liu et al., 2019a), SARC 2.0 (Khodak et al., 2018), and CASCADE (Hazarika et al., 2018). Additionally, we incorporated GRU-Attention model (Akula and Garibay, 2021), which offers intrinsic interpretability through attention scores. We further considered a graph-based method, BERT-GCN (Mohan et al., 2023), alongside ensemble strategies such as Fuzzy-Logic (Dai, 2024) and MULE (Vitman et al., 2023). BiGRU (Najafabadi et al., 2024) is included for its ability to measure sentiment incongruity.

Metrics: We used accuracy, recall, and F1-Score as metrics to evaluate models' performance. Since we use 5-fold cross-validation, we calculated the average of 5 experiment results on the test dataset for each metric as the final result.

5 Results & Discussion

We mainly focus on answering the two questions through our experiments: **Q1:** What is the performance of our white box framework compared to other baselines, and **Q2:** How well does the given explanation represent the true reasoning of the model? We discuss them separately in the following sections.

5.1 Overall Performance

We represent our experiments on three public benchmark datasets in Table 2, Table 3, and Table 5 separately. Overall, our model achieved the highest accuracy, recall, and F1 scores on the Sarcasm Corpus V2 and SARC datasets, outperforming all other baselines. On the Twitter dataset, our model obtained the best accuracy and F1, while GRU-Attention yielded the best recall. Additionally, to compare our approach with CASCADE on the SARC dataset, we incorporated personality features into our model. Of the two versions of our model, the one using a RoBERTa encoder generally outperformed its SBERT-based counterpart.

[2]The model can be downloaded here: https://huggingface.co/sentence-transformers/all-mpnet-base-v2

Models	Accuracy	Recall	F1
Fracking Sarcasm	89.2	87.9	88.1
GRNN	66.4	64.7	65.4
ELMo-BiLSTM	76.2	75.0	75.9
ELMo-BiLSTM FULL	77.4	73.5	75.3
ELMo-BiLSTM AUG	68.6	70.8	69.4
A2Text-Net	91.3	91.0	90.0
GRU-Attention	97.9	**99.6**	98.7
BERT-GCN	88.3	87.1	87.3
MULE	93.5	94.1	93.8
Our Model (SBERT)	98.0	97.3	**98.7**
Our Model (RoBERTa)	**98.3**	98.6	98.4

Table 2: Results on Twitter dataset.

Models	Accuracy	Recall	F1
GRNN	64.4	61.8	61.2
CNN-LSTM-DNN	67.3	66.7	65.7
SIARN	71.9	71.8	71.8
MIARN	74.2	72.9	72.7
ELMo-BiLSTM	74.6	74.7	74.7
ELMo-BiLSTM FULL	76.2	76.0	76.0
GRU-Attention	77.3	77.2	77.2
BiGRU	79.3	81.4	80.2
Fuzzy-Logic	81.8	80.3	81.0
Our Model (SBERT)	82.1	82.5	82.4
Our Model (RoBERTa)	**83.6**	**83.7**	**83.6**

Table 3: Results on Sarcasm Corpus V2 Dialogues dataset

5.2 Case Study for Explanations

In Table 1, we present trained semantic prototypes after projection and the distance score between prototypes and the input comment. The prototypes exhibit a similar topic as the input text, and the keywords are highlighted in yellow by humans for easy illustration (The highlighted words in Table 1 were only meant for reader guidance to aid comparison). For instance, in Case 1, the SBERT prototypes consist of comments skeptical of the efficacy of specific government regulations on cigarettes, and drugs, showing significant overlap with the input comments—a pattern that is also observed in Case 2. Our case study demonstrates that prototype-based models provide more intuitive and human-readable explanations for sarcasm than analyzing a distribution of scores, especially in the absence of strong sarcasm cue words.

When comparing the explanations generated by SBERT and RoBERTa, we did not see a significant

	Input text or similar prototypes	Distance
Case 1	*Post:* Australian state to try and ban cigarettes to anyone born after 2000. *Comments*: yes , because drug prohibition always work so well.	
SBERT	*Post:* France proposes banning the use of underweight models as part of a campaign to stop anorexia. *Comment*: Finally, a problem government can solve.	1.1187
	Post: After kicking out the dea , the UN says Bolivia has successfully lowered its coca cultivation in each of the last four years. *Comment:* if the drug agency is literally a drug agency , can I use 1984 as a reference	1.1244
	Post: Gun control debate reignited by plan to import new shotgun into Australia. *Comment*: I don't understand why Australia has the gun bans that they do considering everything on the freaking continent is trying to kill them	1.1588
RoBERTa	*Post:* Next friday, the United Kingdom will join Australia and become the second country in the world to introduce plain packaging to cigarettes , removing all brands , logos , and colors and replacing them with standardized dark green packages with graphic health warnings. *Comment:* I'm sure that 'll stop people from smoking.	0.9662
	Post: Gunman shoots himself dead in sydney factory siege. *Comment:* Never happened because Australia has gun control	1.0821
	Post: I really want some people to die but I don't want to go to prison so i'm not going to do anything illegal. *Comment:* The system works.	1.1359
Case 2	*Post:* Campaigns must work very hard to pretend that presidential policy choices matter a great deal relative to the actions of the federal reserve. *Comment:* Yes , it 's a real stretch to imagine the fiscal stance of the federal government matters	
SBERT	*Post:* Looks like the Clinton campaign just pressed the big red button. *Comment:* Hey guys , wrap it up , campaign is over , we just got the memo saying that we can't run against a woman , it 's sexist	1.1428
	Post: Ron and Rand Paul now is the time to pass audit the fed. *Comment:* Yeah , thats really the big problem facing America right now .	1.1187
	Post: Bernie Sanders announces bill to abolish private prisons , hints at marijuana policy platform. *Comment:* So really important issues then.	1.1589
RoBERTa	*Post:* Senator Bernie Sanders and several other democratic lawmakers are urging the Obama administration to hold a public hearing to determine whether the national institutes of health should override the patent on a prostate cancer drug in an effort to drive down prices. *Comment:* Lol like the corrupt corporate structure is going to do anything for the greater good.	1.1005
	Post: Chris Christie being voted as potential VP pick for Trump. *Comment:* Because American needs more traffic jams.	1.1573
	Post: Senators are trying to rush through a massive online sales tax hike. *Comment:* New York state residents don't see what all the commotion is about.	1.5731

Table 1: Case examples of input texts, semantic prototypes of SBERT and RoBERTa after projection, and the Euclidean distance score between the prototype embeddings and input sentence vector. **Note:** The highlighted words are only meant for reader guidance and were manually annotated by us to aid comparison.

	w			w/o		
	Acc.	**Rec.**	**F1**	**Acc.**	**Rec.**	**F1**
Twitter	98.3 (+1.1%)	98.6 (-0.1%)	98.4 (+0.9%)	97.2	98.7	97.5
Dialogues	83.6 (+1.7%)	83.7(+1.3%)	83.6 (+1.3%)	82.2	82.6	82.5
SARC	82.4 (+2.2%)	85.8 (+2.4%)	83.0 (+2.9%)	80.2	83.4	81.1

Table 4: Ablation study on the effect of incongruity loss, where 'w' denotes models trained with incongruity loss, and 'w/o' refers to models trained without it. The percentage reflects the variation in performance when training with incongruity loss compared to training without it.

Models	**Accuracy**	**Recall**	**F1**
CASCADE	77.0	84.0	77.0
SARC 2.0	75.0	-	76.0
ELMo-BiLSTM	72.0	-	-
ELMo-BiLSTM FULL	76.0		76.0
GRU-Attention	81.0	82.1	81.0
BiGRU	69.4	68.6	69.0
MULE	75.2	83.8	80.1
Our Model (SBERT)	80.1	**86.2**	82.2
Our Model (RoBERTa)	**82.4**	85.8	**83.0**

Table 5: Results on Reddit dataset SARC 2.0

Method	**Twitter**	**Dialogues**	**SARC**
E_{ct}	97.3	82.7	81.02
$E_{ct} + E_{st}$ (w/o L_{inco})	97.5	82.5	81.1
$E_{ct} + E_{st}$ (with L_{inco})	98.4	83.6	83.0

Table 6: Ablation study on different modules

difference between them, indicating that although, unlike SBERT, RoBERTa is not additionally trained to represent semantic similarity, it still works well with the prototype structure.

5.3 Ablation Study

To fully evaluate our model, we conducted an ablation study evaluating the influence of our **Incongruity Loss**. We experimented with and without the incongruity loss with the RoBERTa encoder on all three datasets and the result is shown in Table 4.

Upon analyzing the results for the Dialogues and SARC datasets, we observed performance improvements ranging from -0.1% to 2.9% when training model with incongruity loss, with SARC dataset showing a particularly notable increase of over 2% on recall, accuracy, and F1. In contrast, the Twitter dataset demonstrated minimal improvement. We hypothesize that this is due to the already high baseline performance on the Twitter dataset, which constrains the extent of further performance gains.

We further perform an ablation study to examine the effect of incorporating the sentiment encoder module in addition to the semantic encoder. The corresponding F1 scores are presented in Table 6. Without the incongruity loss, adding the sentiment encoder yields a comparable F1 score to using only the semantic encoder. However, when the incongruity loss is introduced, the model's performance improves substantially, achieving up to a 2.9% increase in F1.

6 Conclusion

We proposed a novel approach that leverages state-of-the-art LM encoders and prototype-based networks to build an intrinsically interpretable model for sarcasm detection. Our approach achieved state-of-the-art performance on three public benchmark datasets. By representing prototypes with the closest training sentences, our method can explain sarcasm detection with sentence-level, human-readable explanations.

7 Limitation

Our data primarily comes from English-speaking populations from specific platforms, which may not be generalizable. Also, our work does not provide examples to show reasoning with sentiment prototypes. Future research could investigate generating explanations by analyzing the attention scores associated with both negative and positive sentiment prototypes.

8 Ethics Statement

The development and deployment of sarcasm detection models present several ethical considerations. First, our study recognizes the potential biases inherent in training data, particularly those that stem from subjective interpretations of sarcasm across different linguistic, cultural, and social contexts. We used three different datasets from diverse platforms to ensure generalizability. In addition, while our prototype-based explainability framework enhances interpretability, we stress that sarcasm detection remains an inherently complex and nuanced task. We encourage the responsible use of our model to enhance human understanding rather than replace human judgment in sensitive contexts. Therefore, we do not recommend over-reliance on automated sarcasm detection in complex tasks and advocate for human oversight in critical decision-making processes. Moreover, we are committed to transparency in our research by making our model and code publicly available to foster reproducibility. We also adhere to ethical guidelines and ensure that all datasets used in this study comply with proper licensing.

References

Ramya Akula and Ivan Garibay. 2021. Explainable detection of sarcasm in social media. In *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 34–39, Online. Association for Computational Linguistics.

Manjot Bedi, Shivani Kumar, Md Shad Akhtar, and Tanmoy Chakraborty. 2021. Multi-modal sarcasm detection and humor classification in code-mixed conversations. *IEEE Transactions on Affective Computing*, 14(2):1363–1375.

Kun Bu, Yuanchao Liu, and Xiaolong Ju. 2024. Efficient utilization of pre-trained models: A review of sentiment analysis via prompt learning. *Knowledge-Based Systems*, 283:111148.

Yitao Cai, Huiyu Cai, and Xiaojun Wan. 2019. Multi-modal sarcasm detection in Twitter with hierarchical fusion model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2506–2515, Florence, Italy. Association for Computational Linguistics.

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*, TSA '09, page 53–56, New York, NY, USA. Association for Computing Machinery.

Chaofan Chen, Oscar Li, Chaofan Tao, Alina Jade Barnett, Jonathan Su, and Cynthia Rudin. 2019. This

looks like that: Deep learning for interpretable image recognition. *Preprint*, arXiv:1806.10574.

James M. Cupello and David J. Mishelevich. 1988. Managing prototype knowledge/expert system projects. *Commun. ACM*, 31(5):534–550.

Jiakun Dai. 2024. A bert-based with fuzzy logic sentimental classifier for sarcasm detection. In *2024 7th International Conference on Advanced Algorithms and Control Engineering (ICAACE)*, pages 1275–1280. IEEE.

Anubrata Das, Chitrank Gupta, Venelin Kovatchev, Matthew Lease, and Junyi Jessy Li. 2022. ProtoTEx: Explaining model decisions with prototype tensors. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2986–2997, Dublin, Ireland. Association for Computational Linguistics.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcasm in Twitter and Amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 107–116, Uppsala, Sweden. Association for Computational Linguistics.

Richard Fikes and Tom Kehler. 1985. The role of frame-based representation in reasoning. *Commun. ACM*, 28(9):904–920.

V. Fomin, J. Anmol, S. Desroziers, J. Kriss, and A. Tejani. 2020. High-level library to help with training neural networks in pytorch. `https://github.com/pytorch/ignite`.

Aniruddha Ghosh and Tony Veale. 2016. Fracking sarcasm using neural network. In *Proceedings of the 7th workshop on computational approaches to subjectivity, sentiment and social media analysis*, pages 161–169.

Roberto González-Ibáñez, 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*, pages 581–586, Portland, Oregon, USA. Association for Computational Linguistics.

John A Hartigan and Manchek A Wong. 1979. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108.

Jochen Hartmann, Mark Heitmann, Christian Siebert, and Christina Schamp. 2023. More than a feeling: Accuracy and application of sentiment analysis. *International Journal of Research in Marketing*, 40(1):75–87.

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. Cascade: Contextual sarcasm detection in online discussion forums. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1837–1848. Association for Computational Linguistics.

Nivin A Helal, Ahmed Hassan, Nagwa L Badr, and Yasmine M Afify. 2024. A contextual-based approach for sarcasm detection. *Scientific Reports*, 14(1):15415.

Dat Hong, Tong Wang, and Stephen Baek. 2024. Protorynet - interpretable text classification via prototype trajectories. *J. Mach. Learn. Res.*, 24(1).

E Ilavarasan et al. 2020. A survey on sarcasm detection and challenges. In *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pages 1234–1240. IEEE.

Suzana Ilic, Edison Marrese-Taylor, Jorge Balazs, and Yutaka Matsuo. 2018. Deep contextualized word representations for detecting sarcasm and irony. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 2–7.

Adam Ivankay, Ivan Girardi, Chiara Marchiori, and Pascal Frossard. 2022. Fooling explanations in text classifiers. *Preprint*, arXiv:2206.03178.

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)*, pages 757–762.

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2018. A large self-annotated corpus for sarcasm. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Been Kim, Cynthia Rudin, and Julie Shah. 2015. The bayesian case model: A generative approach for case-based reasoning and prototype classification. *Preprint*, arXiv:1503.01161.

Akshi Kumar, Shubham Dikshit, and Victor Albuquerque. 2021a. Explainable artificial intelligence for sarcasm detection in dialogues. *Wireless Communications and Mobile Computing*, 2021:1–13.

Akshi Kumar, Shubham Dikshit, and Victor Hugo C Albuquerque. 2021b. Explainable artificial intelligence for sarcasm detection in dialogues. *Wireless Communications and Mobile Computing*, 2021(1):2939334.

Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. 2018. Deep learning for case-based reasoning through prototypes: a neural network that explains its predictions. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium*

on Educational Advances in Artificial Intelligence, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Liyuan Liu, Jennifer Lewis Priestley, Yiyun Zhou, Herman E Ray, and Meng Han. 2019a. A2text-net: A novel deep neural network for sarcasm detection. In *2019 IEEE first international conference on cognitive machine intelligence (CogMI)*, pages 118–126. IEEE.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *Preprint*, arXiv:1907.11692.

Scott M. Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874.

Dina Mardaoui and Damien Garreau. 2021. An analysis of lime for text data. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 3493–3501. PMLR.

Anuraj Mohan, Abhilash M Nair, Bhadra Jayakumar, and Sanjay Muraleedharan. 2023. Sarcasm detection using bidirectional encoder representations from transformers and graph convolutional networks. *Procedia Computer Science*, 218:93–102.

Maryam Khanian Najafabadi, Thoon Zar Chi Ko, Saman Shojae Chaeikar, and Nasrin Shabani. 2024. A multi-level embedding framework for decoding sarcasm using context, emotion, and sentiment feature. *Electronics*, 13(22):4429.

Shereen Oraby, Vrindavan Harrison, Lena Reed, Ernesto Hernandez, Ellen Riloff, and Marilyn Walker. 2017. Creating and characterizing a diverse corpus of sarcasm in dialogue. *arXiv preprint arXiv:1709.05404*.

Hongliang Pan, Zheng Lin, Peng Fu, and Weiping Wang. 2020. Modeling the incongruity between sentence snippets for sarcasm detection. In *ECAI 2020*, pages 2132–2139. IOS Press.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *Preprint*, arXiv:1908.10084.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. *CoRR*, abs/1602.04938.

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, Seattle, Washington, USA. Association for Computational Linguistics.

Rishabh Misra and Prahal Arora. 2018. Sarcasm detection using hybrid neural network.

Jaroslav Švelch. 2015. Excuse my poor english: language management in english-language online discussion forums. *International journal of the sociology of language*, 2015(232):143–175.

Yi Tay, Luu Anh Tuan, Siu Cheung Hui, and Jian Su. 2018. Reasoning with sarcasm by reading in-between. *arXiv preprint arXiv:1805.02856*.

Joseph Tepperman, David R. Traum, and Shrikanth S. Narayanan. 2006. "yeah right": Sarcasm recognition for spoken dialogue systems. In *Interspeech*.

Oren Tsur, Dmitry Davidov, and Ari Rappoport. 2010. Icwsm - a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. *Proceedings of the International AAAI Conference on Web and Social Media*.

Geng Tu, Tian Xie, Bin Liang, Hongpeng Wang, and Ruifeng Xu. 2024. Adaptive graph learning for multimodal conversational emotion detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19089–19097.

Oxana Vitman, Yevhen Kostiuk, Grigori Sidorov, and Alexander Gelbukh. 2023. Sarcasm detection framework using context, emotion and sentiment features. *Expert Systems with Applications*, 234:121068.

Ximing Wen. 2024. Language model meets prototypes: Towards interpretable text classification models through prototypical networks. *arXiv preprint arXiv:2412.03761*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhe Yu, Di Jin, Xiaobao Wang, Yawen Li, Longbiao Wang, and Jianwu Dang. 2023. Commonsense knowledge enhanced sentiment dependency graph for sarcasm detection. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 2423–2431.

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.