Humor and Jokes

Bucă Mihnea-Vicențiu and **Căpățînă Răzvan-Nicolae** and **Ciobanu Dragoș** and **Petre-Șoldan Adela**
mihnea-vicentiu.buca@s.unibuc.ro
razvan-nicolae.capatina@s.unibuc.ro
dragos.ciobanu@s.unibuc.ro
adela.petre-soldan@s.unibuc.ro

Abstract

Humor and jokes are pervasive in human communication, yet remain a significant challenge for Natrual Language Processing (NLP) due to their reliance on nuanced lexical cues, pragmatic context, and implicit world knowledge. In this paper, we present a unified study of two core tasks in computational humor: joke detection and open-ended joke generation. We curate and leverage a large-scale, diverse corpus comprising six humor datasets, contrasted with *News Articles* as non-humorous examples. For joke classification, we evaluate multiple methodologies: a baseline `TF-IDF` model, pretrained `spaCy` embeddings paired with a neural regressor, and a bidirectional transformer (`BERT`) trained to predict humor scores, assigning higher values to jokes and near-zero scores to non-jokes. For generation, we propose a hybrid approach, fine-tuning autoregressive models (`GPT-2`, `BART`) and a recurrent neural network (`RNN`), while also introducing a **custom transformer-based architecture** optimized for humor generation through dataset-driven learning. Our results highlight the connection between computational methods, and the layered nature of humor, showing how models can link low-level linguistic cues, such as wordplay and semantic shifts, with higher-level contextual understanding. By jointly addressing detection and generation, we illustrate the potential of unified approaches for modeling humor's complexity, and emphasize the importance of stylistic diversity in approximating real-world humor.

1 Introduction

Computational humor has emerged as a multidisciplinary field at the intersection of linguistics, psychology, and artificial intelligence, aiming to model, understand, and generate humor through computational means. Unlike many language tasks governed by explicit syntactic or semantic rules, it draws on linguistic theories of wordplay and ambiguity, psychological models of incongruity and the subversion of expectations (Attardo, 2000), and machine learning techniques for pattern recognition and generation. Jokes often require context-awareness and commonsense reasoning-areas. This convergence makes humor a uniquely challenging Natrual Language Processing (NLP) problem, as it requires not just syntactic or semantic understanding, but also pragmatic reasoning, cultural awareness, and often shared background knowledge. Yet, despite its ubiquity and importance, modeling humor computationally remains one of the most persistent challenges in NLP (Kalloniatis and Adamidis, 2024b; Diya et al., 2025). For example, resolving lexical ambiguity or detecting irony depends heavily on context and shared cultural knowledge. Similarly, modeling the unexpected shifts that produce humor demands a nuanced understanding of both language and real-world references (Carmen Curcos Cobos, 1997). Despite notable progress in isolated detection systems (Mihalcea and Strapparava, 2005; Romanowski et al., 2025) and standalone generation frameworks (Turano and Strapparava, 2022), these two strands have rarely been evaluated together or trained on the same diverse dataset. As a result, detection models miss insights into generative patterns, and joke generators lack rigorous benchmarks for humor quality. In this paper, we close this gap by unifying joke classification and generation under a common large-scale dataset, enabling comparative analysis and mutual improvement of both tasks. By evaluating both humor detection and generation within a shared computational framework, we aim to uncover synergistic relationships between these traditionally siloed tasks. For instance, we hypothesize that models trained to assign continuous humor scores, rather than binary labels, can provide fine-grained supervision for generative systems. These scores can then guide the generation of more contextually appropriate punchlines, in contrast to prior work that treats humor detection as

a static binary filter. This joint setup allows us to systematically probe the linguistic and pragmatic structures that underlie humor, while assessing the extent to which shared representations can support both understanding and generation.

2 Related Work

Early work in humor detection explored rule-based approaches and shallow linguistic features. For example (Mihalcea and Strapparava, 2005) explored the feasibility of computationally distinguishing humorous from non-humorous text. They collected a dataset of 16,000 humorous one-liners and compared them against non-humorous examples from Reuters titles, proverbs, and the British National Corpus. Their approach involved supervised learning with decision trees trained on these handcrafted features, illustrating the potential of computational models to detect humor-relevant patterns. Subsequent studies expanded on this foundation by incorporating additional cues like incongruity, phonetic similarity, and semantic shift (Valitutti, 2018).

The advent of deep learning has significantly advanced the field of humor detection by enabling models to learn complex patterns and contextual nuances inherent in humorous text (Chen and Soo, 2018). Traditional machine learning approaches often relied on handcrafted features and shallow models, which limited their ability to capture the intricacies of humor. Deep learning models, particularly Convolutional Neural Networks (CNNs) (Bertero and Fung, 2016) and Recurrent Neural Networks (RNNs) (Kalloniatis and Adamidis, 2024a), have demonstrated superior performance by leveraging their capacity to process sequential data and extract hierarchical features. Parallel to detection efforts, humor generation initially relied on template-based systems constrained by rigid schemas (e.g., 'Why did the chicken cross the road?'). These approaches, while interpretable, struggled to generalize beyond formulaic joke structures (Binsted et al., 2006). (Amin and Burghardt, 2020) systematically reviewed generative humor systems, evaluating them against linguistic theories like incongruity-resolution and superiority theory. They highlighted key limitations, particularly in creativity and adaptability.

The emergence of large language models (LLMs) like GPT-2 and BART (Lewis et al., 2019) has significantly advanced the field of natural language processing (NLP), enabling data-driven joke generation through fine-tuning on diverse corpora. These models have demonstrated the ability to generate coherent and contextually relevant humor, though challenges remain in achieving human-like comedic quality. Recent studies, however, critique these models for over-reliance on surface-level patterns (e.g., question-answer formats) and inadequate handling of cultural or pragmatic context (Jentzsch and Kersting, 2023; Horvitz et al., 2024).

Systematic reviews, such as, (Kalloniatis and Adamidis, 2024a), underscore the persistent divide between detection and generation research. While detection models increasingly adopt transformer architectures like `BERT` to capture humor's contextual nuances, generative systems remain siloed, rarely incorporating feedback from detection frameworks to iteratively refine output quality.

While prior work often separates humor detection (Mihalcea and Strapparava, 2005; Turano and Strapparava, 2022) and generation, we propose a unified framework where humor detection scores directly inform generative model training. By training `BERT` to predict continuous humor scores instead of binary labels, we provide a nuanced signal that prioritizes high-quality examples for fine-tuning generative models like `GPT-2`. This contrasts with previous studies that treat detection as isolated or train generators on unfiltered datasets. Additionally, we address the stylistic limitations of humor datasets, which often overrepresent short-form puns and one-liners. Our diverse datsets, spanning dad jokes, narrative humor and some satire, reveals model limitations in handling humor's contextual breadth. This study highlights the need for humor-specific inductive biases in generative models to bridge low-level linguistic cues with high-level narrative surprise.

3 Method

Our methodology consists of two interconnected components: **a humor detection task** framed as a regression problem, and a **humor generation task** targeting diverse joke synthesis. For detection, we train models to assign continuous humor scores to text samples, allowing for more nuanced predictions than binary classification. We implement and compare two approaches: a classical machine learning pipeline using `TF-IDF` features with a neural regressor, and a transformer-based regression model fine-tuned from `BERT`. For generation, we fine-tune several language models, including

GPT-2, BART, and **a custom transformer**, using only high-scoring examples as judged by the detection models. This bidirectional setup allows us to investigate how detection feedback can improve generative quality and, conversely, how generation models respond to stylistic variance and contextual demands across joke types.

3.1 Dataset

To support both classification and generation tasks, we curate a large-scale and stylistically diverse humor corpus composed of six major joke datasets and a non-humorous control set of news articles. The preprocessed datasets share a unified schema with the following columns: ID, Title, Category, Body, and Rating. While fields like ID, Title, and Rating may be missing in some cases, the Body—containing the full joke text—is always present and used across all experiments.

- **Short Jokes**[1]: over 200k brief, one-line jokes.
- **Jester**[2]: a 1.7M joke rating dataset from a joke recommendation system.
- **Dadjokes**[3]: Puns and dad jokes.
- **Stupidstuff, Wocka, and Reddit Jokes**[4]: scraped joke corpora emphasizing community-rated humor.

These datasets differ in structure and humor genre, ranging from short puns to multi-line narratives and satirical setups. We also include a balanced set of non-humorous *News Articles* [5] to serve as negative examples in the classification task. The variety in form, source, and humor style provides a robust testbed for analyzing both linguistic and contextual dimensions of computational humor.

3.2 Preprocessing

To ensure the integrity and quality of the input data, we apply a series of preprocessing steps focused on filtering inappropriate content, cleaning text, and structurally formatting the jokes for model consumption.

First, we discard any entries with missing or empty Body fields and remove duplicates to eliminate redundancy. We then filter out jokes that contain profanity or hate speech using a keyword-based blacklist and simple regular expressions. Only "positive" jokes—those free of offensive language—are retained for both classification and generation tasks.

Next, we apply a structural normalization step to prepare jokes for models that benefit from explicit semantic segmentation (especially during generation). Each joke is heuristically split into two components: the *setup* and the *punchline*. Using a rule-based function, we locate the final sentence-ending punctuation mark (e.g., ., !, or ?) and split the joke at that point. If no such punctuation is detected, we fall back to a midpoint split based on character count. This method helps preserve the narrative structure of humor and allows models to better align the buildup and punchline during training.

3.3 TF-IDF Vectorization with Neural Regressor

We implement a simple regression model that predicts humor scores based on TF-IDF[6] features extracted from joke bodies. The text is lowercased, cleaned, and vectorized using unigrams and bigrams with a fixed feature size.

The model is a feed-forward neural network with three hidden layers and ReLU activations. It is trained using mean squared error and optimized with Adam. After training, we apply a sigmoid function to the outputs and select a threshold on the validation set to convert scores into binary predictions.

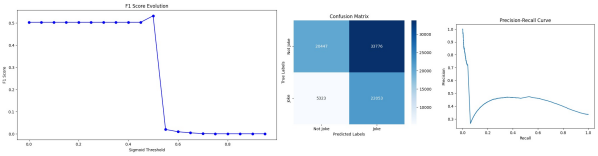

Figure 1: TF-IDF Vectorization with Neural Regressor

This approach serves as a straightforward and interpretable baseline.

[1] https://www.kaggle.com/datasets/abhinavmoudgil95/short-jokes

[2] https://www.kaggle.com/datasets/vikashrajluhaniwal/jester-17m-jokes-ratings-dataset

[3] https://huggingface.co/datasets/shuttie/dadjokes

[4] https://github.com/taivop/joke-dataset

[5] https://www.kaggle.com/datasets/rmisra/news-category-dataset

[6] https://shorturl.at/2f4zK

3.4 Transformer-Based Humor Regression with BERT

We fine-tune a pre-trained `bert-base-uncased`[7] model to predict continuous humor scores. After minimal cleaning (lowercasing, punctuation removal, digit replacement), each joke is tokenized into 64 subword tokens. We replace BERT's classification head with a single linear output and train for 3–5 epochs using AdamW (lr=5e-5) to minimize MSE. Post-training, a sigmoid function is applied and an optimal threshold is selected on validation data to yield binary joke/non-joke predictions. This pipeline leverages contextual embeddings to capture semantic and pragmatic humor cues that sparse models cannot.

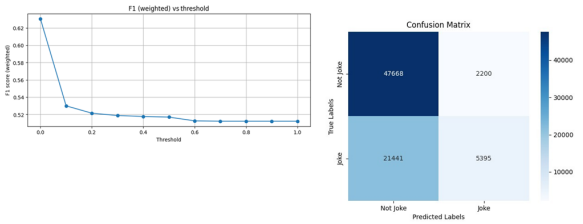

Figure 2: BERT Regression

3.5 Custom Transformer for Next-Token Humor Generation

We train a word-level transformer[8] from scratch using TensorFlow/Keras for next-token prediction. The model uses token embeddings and a stack of transformer encoder blocks with self-attention and feed-forward layers. It is trained using sparse categorical cross-entropy to predict the next word in input sequences. At inference time, it generates humorous text autoregressively from a given prompt.

3.6 BART Fine-Tuning for Joke Generation

We fine-tune a pretrained BART[9] language model using HuggingFace Transformers for sequence-to-sequence joke generation. Input prompts and corresponding punchlines are tokenized using a BART tokenizer, and the model is trained to reconstruct punchlines from prompts. The architecture consists of an encoder-decoder transformer with shared embeddings and learned positional encodings. During training, we use teacher forcing with cross-entropy loss on the decoder outputs. At inference time, the model generates jokes autoregressively from a given prompt using beam search or greedy decoding.

3.7 GPT-2 Fine-Tuning for Joke Generation

We fine-tune a pretrained GPT-2[10] language model for humor generation using HuggingFace Transformers. A corpus of jokes is tokenized using a GPT-2 tokenizer with the EOS token as padding. The architecture is a unidirectional decoder-only transformer with causal self-attention. Each joke is framed as a single language modeling task, where the model learns to predict the next token in the sequence. During training, we minimize the causal language modeling loss over full joke texts. At inference time, the model generates jokes autoregressively from an initial seed using greedy or sampling-based decoding strategies.

3.7.1 GPT-2 Fine-Tuning with Humor Detection Feedback

We fine-tune a GPT-2[11] model using the HuggingFace `Trainer` API on a combined dataset of jokes from Jester, Reddit, and Stupidstuff, normalized and filtered using previously defined rating-based methods. Only clean samples with a normalized rating above 0.5 are retained. Prompts are masked during loss computation to direct learning toward punchlines. Training is run for 3 epochs using mixed precision and saved for later use.

4 Future Work

While our unified framework for humor detection and generation demonstrates promising synergies, several directions remain open:

- **Humor-Specific Architectures.** Incorporate inductive biases for phonetic ambiguity and incongruity (e.g., specialized attention heads or hierarchical encoders) to better capture punchline dynamics.
- **Cross-Cultural and Multilingual Humor.** Extend datasets and models to other languages and cultural contexts, and explore transfer learning to handle idiomatic expressions and wordplay in multiple tongues.
- **Interactive Systems.** Integrate detection and generation pipelines into a conversational

[7] https://shorturl.at/PtpOh
[8] https://shorturl.at/nlB2G
[9] https://shorturl.at/Vvq5V
[10] https://shorturl.at/mHT4C
[11] https://shorturl.at/7HoQD

agent or writing assistant, allowing real-time humor feedback and on-the-fly joke suggestions.

- **Reinforcement Learning for Quality.** Use human or classifier feedback as reward signals to fine-tune generators, optimizing for metrics like perceived funniness or coherence.

- **Multi-Modal Humor.** Combine textual models with image or audio inputs (memes, stand-up clips) to study how visual and auditory cues contribute to humorous effect.

- **Bias and Safety.** Systematically audit models for offensive or biased content and develop filtering or steering methods to ensure safe, inclusive humor.

5 Conclusion

We have presented a unified framework for joke detection and generation, benchmarking both a TF-IDF + neural regressor and a fine-tuned BERT model on a diverse humor corpus.

Detection. The TF-IDF pipeline (Figure 1, left) achieves high recall (0.806) but moderate precision (0.395), yielding an F1 score of 0.530. This makes it well suited as a broad joke filter but prone to false positives. In contrast, BERT (Figure 2, right) attains higher overall balance, with precision 0.71, recall 0.20 on the "joke" class, and a weighted F1 of 0.63. BERT's contextual embeddings clearly improve precision at the cost of reduced sensitivity.

Generation. We compare four generation strategies:

- **Custom Transformer:** "My friend asked me if you have a dad … the bartender says sir cumference" (indicates limited long-range coherence).

- **BART:** "*the bartender asks him what he wants to drink … im sorry but you cant have a beer*" (coherent dialogue but minimal punch).

- **GPT-2 (baseline):** "*Bob: What's the difference between a woman and a man? … I don't know what you're talking about.*" (grammatical but unfunny).

- **GPT-2 with detection feedback:**

  1. "Yo mamma so fat she stuck her head out of the window on a mule!" said the young lady. "what?"
  2. "Yo mamma so fat she lost her weight! … damn! she lost it!"
  3. "Yo mamma so fat she walked into a limo … and got in the way of a car."
  4. "Yo mamma so dumb she … and got out her cell phone."
  5. "Yo mamma so stupid, that she threw a birthday party … and all the guests were invited to join her."

  (diverse, stylistically varied punchlines with recognizable humor).

The feedback-informed GPT-2 clearly produces the most engaging and varied jokes, demonstrating the value of using detection scores to filter and prioritize training examples.

Outlook. Building on these results, future systems can combine TF–IDF's broad recall with BERT's precision in a cascade, and leverage detection-guided feedback loops to refine generative models. Jointly optimizing both tasks promises more robust, contextually aware humor agents capable of both spotting and crafting jokes in real time.

Limitations

Our work has several limitations that suggest avenues for improvement:

- **Language and Cultural Scope.** We train exclusively on English-language jokes and news articles, limiting applicability to other languages and cultural humor traditions.

- **Dataset Biases.** Public joke corpora overrepresent short puns and one-liners; longer narrative or situational humor remains underexplored, and rating distributions reflect community preferences rather than objective funniness.

- **Compute Requirements.** Fine-tuning transformer models (BERT, GPT-2, BART) requires GPUs with substantial memory, which may not be accessible for smaller research groups or real-time deployment.

- **Real-Time Integration.** The current pipelines are not optimized for low-latency

inference, constraining their use in interactive applications (e.g., chatbots) without further model compression or pruning.

- **Safety and Bias.** Although we filter profanity, the models can still generate offensive or culturally insensitive content; robust bias mitigation and safety filters are needed before production deployment.

- **Evaluation Metrics.** Our reliance on regression MSE and binary F1 overlooks subjective dimensions of humor (e.g., novelty, appropriateness), motivating the development of richer, human-centered evaluation frameworks.

Ethical Statement

Humor generation and detection carry potential risks if misused or deployed without care:

Unethical Uses Adversaries could leverage our generative models to create targeted disinformation or propaganda wrapped in humorous framing, increasing virality and lowering readers' critical guard. Automated joke generation could also produce offensive, hateful, or culturally insensitive content at scale, amplifying harmful stereotypes.

Bias and Fairness Our models are trained on community-sourced joke corpora that overrepresent certain demographic groups, styles (e.g., "yo mamma" jokes), and cultural references. As a result, the system may perpetuate gender, ethnic, or ageist biases present in the training data. Moreover, the binary mask (joke vs. non-joke) and regression ratings reflect crowd preferences rather than universal standards of humor, potentially marginalizing niche or underrepresented comedic forms.

Mitigation Measures To reduce overtly harmful content, we filter profanity and known slurs before training. We recommend:

- **Data Auditing:** Continuously inspect generated outputs for bias or toxicity and refine filters.

- **Human-in-the-Loop:** Deploy humor systems with human moderators or reviewers in high-stakes domains (e.g., educational tools).

- **Diverse Data:** Incorporate balanced, multicultural joke datasets and invite contributions from underrepresented groups.

Responsible Use We advise practitioners to treat our models as assistive tools—providing suggestions rather than automated publication. In interactive applications (chatbots, writing aids), always include a user override and clear disclaimers about potential inaccuracies or offensiveness.

Personal Stance Humor is a powerful engagement mechanism, but it must be wielded responsibly. I believe that transparent reporting of model limitations, proactive bias mitigation, and ongoing community feedback are essential to harness humor NLP ethically and inclusively.

References

Miriam Amin and Manuel Burghardt. 2020. A survey on approaches to computational humor generation. In *Proceedings of the 4th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 29–41, Online. International Committee on Computational Linguistics.

Salvatore Attardo. 2000. Humorous texts: A semantic and pragmatic analysis.

Dario Bertero and Pascale Fung. 2016. Deep learning of audio and language features for humor prediction. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 496–501, Portorož, Slovenia. European Language Resources Association (ELRA).

Kim Binsted, Anton Nijholt, Oliviero Stock, Carlo Strapparava, Graeme Ritchie, Ruli Manurung, Helen Pain, Annalu Waller, and Dave O'Mara. 2006. Computational humor. *Intelligent Systems, IEEE*, 21:59–69.

Maria Carmen Curcos Cobos. 1997. The pragmatics of humorous interpretations.

Peng-Yu Chen and Von-Wun Soo. 2018. Humor recognition using deep learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 113–117, New Orleans, Louisiana. Association for Computational Linguistics.

Diya, Arunima Jaiswal, Labanti Purty, Smitanna Mandal, and Nitin Sachdeva. 2025. A study on humor detection on social media data using machine learning and explainable ai.

Zachary Horvitz, Jingru Chen, Rahul Aditya, Harshvardhan Srivastava, Robert West, Zhou Yu, and Kathleen McKeown. 2024. Getting serious about humor: Crafting humor datasets with unfunny large language models.

Sophie Jentzsch and Kristian Kersting. 2023. Chatgpt is fun, but it is not funny! humor is still challenging large language models.

Antonios Kalloniatis and Panagiotis Adamidis. 2024a. Computational humor recognition: a systematic literature review. *Artificial Intelligence Review*, 58(2):43.

Antonios Kalloniatis and Panagiotis Adamidis. 2024b. Computational humor recognition: A systematic literature review - artificial intelligence review.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Rada Mihalcea and Carlo Strapparava. 2005. Making computers laugh: Investigations in automatic humor recognition. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 531–538, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Adrianna Romanowski, Pedro H. V. Valois, and Kazuhiro Fukui. 2025. From punchlines to predictions: A metric to assess llm performance in identifying humor in stand-up comedy.

Beatrice Turano and Carlo Strapparava. 2022. Making people laugh like a pro: analysing humor through stand-up comedy.

Alessandro Valitutti. 2018. Humor facilitation of polarized events. In *Distributed, Ambient and Pervasive Interactions: Technologies and Contexts*, pages 337–347, Cham. Springer International Publishing.

A Example Appendix

BERT (Bidirectional Encoder Representations from Transformers)[12] BERT is a transformer-based model designed for understanding the context of words in a sentence.

BART (Bidirectional and Auto-Regressive Transformers)[13] BART is a transformer model that combines the bidirectional context of BERT with the auto-regressive capabilities of GPT. It's particularly effective for text generation and summarization tasks.

GPT-2 (Generative Pre-trained Transformer 2)[14] GPT-2 is a transformer-based model that generates human-like text. It uses a unidirectional approach, predicting the next word in a sequence based on the previous words.

[12] https://huggingface.co/docs/transformers/en/model_doc/bert

[13] https://huggingface.co/docs/transformers/en/model_doc/bart

[14] https://huggingface.co/docs/transformers/en/model_doc/gpt2