Automating Humor: A Novel Approach to Joke Generation Using Template Extraction and Infilling

Mayank Goel **Parameswari Krishnamurthy** **Radhika Mamidi**
Language Technologies Research Centre
IIIT Hyderabad
mayank.goel@research.iiit.ac.in, {param.krishna, radhika.mamidi}@iiit.ac.in

Abstract

This paper presents a novel approach to humor generation in natural language processing by automating the creation of jokes through template extraction and infilling. Traditional methods have relied on predefined templates or neural network models, which either lack complexity or fail to produce genuinely humorous content. Our method introduces a technique to extract templates from existing jokes based on semantic salience and BERT's attention weights. We then infill these templates using advanced techniques, through BERT and large language models (LLMs) like GPT-4, to generate new jokes. Our results indicate that the generated jokes are novel and human-like, with BERT showing promise in generating funny content and GPT-4 excelling in creating clever jokes. The study contributes to a deeper understanding of humor generation and the potential of AI in creative domains.

1 Introduction

Humor Generation in NLP is the task of generating jokes through a computer system. It is particularly challenging, and can be seen as a short creativity task. Critical to solving this task is understanding humor theory, which attempts to explain the structure and mechanism of jokes. Ideas from (Raskin, 1985) and then (ATTARDO and RASKIN, 1991), building up from various theories from antiquity, have talked about how humor is based on the incongruity mechanism - the punchline is funny because it deviates from the expectations after the setup.

The task of humor generation is important because of how culturally significant Humor is, and how ingrained it is in our society, and yet we don't have any clear mechanism to produce a joke. Applying computational methods on this task can lead us closer to understanding Humor, as well as to Language itself, since humor can be seen as a short creative task that is currently not understood well.

Most attempts at humor generation in the past have used template-based methods. The primary limitation in template-based methods is that they are restricted in the template they are infilling, and for neural methods is that they don't generate humorous content. Chaudhary et al (Chaudhary et al., 2021) attempted to bridge this gap by generating templates from existing jokes and then infilling them, thus aiming to combine the creativity of neural outputs with the structured humor of template-based methods. For extraction, they used a score that took two heuristics into account:

1. Using the role of a word in the dependency tree, with manually assigned weights to each role
2. Using the frequency of the word in common usage, with lesser frequent words having a higher score.

However, this approach is contingent upon the universal applicability of the assigned weights and the accuracy of the dataset used for frequency analysis.

In our paper, we propose enhancements to this model, introducing more robust and scalable methods that aim to generalize beyond the limitations of manual weight assignments and dataset validity. These methods use contextual embeddings to understand the semantic role of a word.

2 Methodology

2.1 Overview

The process of generating a new joke computationally involves dissecting an existing joke into two distinct parts: style and content. 'Content' encompasses the specific elements that give the joke its meaning and humor, while 'style' refers to the syntax and structure that make the joke coherent and understandable. Our approach seeks to isolate these two aspects by extracting a template that captures the style of the joke. We then fill in this template with new content. This technique is designed to retain the original structural humor of the joke while introducing new thematic material, thus creating a new joke that resonates with the familiarity of the original's format.

2.2 Template Extraction

For template extraction, the model should ideally extract an ideal set of words such that the resulting template has both - scope for creativity and generation of a new joke, while being sufficiently restrictive so that the generation is based on the "funny" structure of the original joke. Given a template, the infilling model should create a new joke by filling in the blanks. The task of separating style and content in a joke is not straightforward. Consider this classic joke:

"Why did the chicken cross the road? To get to the other side"

An ideal template to generate would be:

"Why did the [MASK] cross the [MASK]? To get to [MASK]"

This template has enough of the original joke to maintain the core syntactic style, yet leaves scope for new words to be creatively filled in.

To achieve this, our model identifies content words to be masked while preserving style words. We filter potential words for masking based on their part-of-speech tags, selecting nouns, adjectives, and adverbs as likely candidates. Verbs, while semantically significant, tend to contribute more to the joke's style; hence, we retain them to preserve the structural integrity of the humor.

The extraction algorithm processes a given sentence and evaluates each word based on two metrics: its semantic importance to the overall sentence and the attention weights as determined by BERT (Devlin et al., 2019). Each word is assigned a score by combining the two metrics above, and half the words in each of the setup and the punchline are masked.

2.2.1 Attention-based Importance Scoring

In order to understand which words in a joke are most pivotal to its meaning, we use the power of the attention mechanism found in transformer models, specifically the BERT model. The attention mechanism offers a granular understanding of contextual interdependencies between words, making it a robust method for word importance estimation.

Given a sentence and a set of target words, the function procures the attention weights for each token from the model's outputs. Recognizing the significance of the latter layers in a transformer model, which often capture higher-level semantics, we opted to focus on the attention weights from the last four layers. These are weighted progressively with values [0.1, 0.2, 0.3, 0.4] to give more emphasis to the final layer, assuming that the later layers encode more semantically relevant information.

As the BERT tokenizer breaks words into subwords or tokens, we sum the attention weights of subwords corresponding to a single word. Once we obtain an attention matrix, each word's importance score is computed as the sum of its attention weights across all tokens.

The major justification for using attention-based importance scoring is its inherent capability to capture complex relationships and dependencies between words. Unlike traditional methods that might solely rely on static word properties, attention scores derive their strength from understanding the contextual interplay of words within a sentence. This ensures that words are not evaluated in isolation but in terms of their role and contribution to the overall meaning of the sentence, making this approach particularly suited for discerning the nuanced humor elements in jokes.

2.2.2 Semantic Weight Scoring

The methodology hinges on the premise that the removal of a semantically significant chunk from a sentence would induce a considerable shift in its overall meaning. By measuring this shift, we can quantify the importance of the chunk in relation to the sentence.

To operationalize this idea, we use sentence transformers to encode a sentence, which calculates a sentence embedding by mean pooling the word embeddings. For each word in the candidates list initially identified, we generate two sets of embeddings:

1. **Original Embedding**: The embedding for the entire sentence, capturing the collective semantic essence with the chunk in context.

2. **Reduced Embedding**: The embedding for the sentence sans the chunk in question, presenting a semantic landscape where the chunk's contribution is absent.

The divergence between these embeddings, quantified using cosine similarity, serves as a proxy for the chunk's importance. A substantial divergence implies that the removed chunk was semantically pivotal, thus commanding a higher importance score. By systematically evaluating all identified chunks in the sentence, we can rank them based on their computed importance scores.

2.2.3 Template Generation

We use a dataset of short jokes (Gulrajani, 2024) as the base set of jokes, which consists of 230k jokes. We finetune BERT on the train split of our dataset (80% of the dataset) to make it suited for the domain, and use the test split as candidates for template extraction. We combine the scores derived from semantic weights and BERT's attention metrics to assign a composite score to each word within a sentence. For jokes structured as a question and answer—or a setup followed by a punchline—we apply a masking strategy that alters half of the words in both the setup and the punchline, rounding up if necessary. This ensures that both components of the joke undergo modification.

The decision to mask half of the words is a deliberate one, informed by qualitative assessments that suggest this proportion adeptly balances the need for novelty with the retention of the original joke's framework. Masking too few words might leave the joke overly intact and familiar, while masking too many could strip away the essential humor structure.

2.3 Joke Generation

2.3.1 BERT Infilling

The [MASK] tokens require contextually suitable words to complete the humor inherent in the joke. The mask-filling task is in line with how BERT has been originally trained — to predict and fill in missing words in a sentence. For each '[MASK]' token, the function prompts a pipeline that proposes potential fill-ins. These candidates are evaluated by inserting them into the sentence and measuring the semantic shift from our initial benchmark through cosine similarity. The goal is to find the candidate whose inclusion results in the least deviation from the original sentence's semantic embedding.

This chosen candidate is likely the most appropriate filler, as it maintains the semantic coherence of the original sentence most closely. The candidate then replaces the '[MASK]', and the function moves on to the next placeholder. Employing BERT for this context-driven infilling leverages its strength in understanding language context and structure, as it was extensively trained on vast amounts of text data, including being finetuned on a dataset of jokes.

2.3.2 LLM Infilling

To get a comprehensive view of current LLM capabilities, we use two models - Zephyr-7B-Beta (Tunstall et al., 2023) and GPT-4 (OpenAI, 2023). Zephyr-7B-Beta is a LLM finetuned on the Mistral-7B (Jiang et al., 2023) model on a mix of datasets, and the authors "found that removing the in-built alignment of these datasets boosted performance on MT Bench and made the model more helpful" (Tunstall et al., 2023). The authors claim that the model outperforms Llama2-Chat-70B. On the other hand, GPT-4 has been recognized as the highest-performing LLM across multiple evaluative leaderboards (llm, 2024). Our intent was to compare LLMs of different capabilities to get a clearer idea of the suitability of infilling to generate jokes. We prompted the LLMs to generate a joke by filling in the blanks.

3 Evaluation

3.1 Experiment Setup

Bisten and Ritchie (Binsted and Ritchie, 1994) suggest the Turing Test to evaluate jokes: asking the evaluators whether the joke is human-generated, regardless of how funny it is. We use this as our evaluation task, with the hypothesis being that if humans perform badly on this task, it means the model is able to generate human-like jokes. We evaluate our BERT model and GPT-4 model, leaving Zephyr aside as GPT-4 is a better performing LLM, and BERT's performance is indicative of how good the template generation process is, as GPT-4 is significantly more powerful as a generative model.

We crowd-source the evaluation, sending the evaluators a script to run. At each instance, the model shows either a human or a computer generated joke, with one of BERT or GPT-4 chosen randomly. We get about 500 human annotations from 15 annotators, who were instructed to annotate as many jokes as they like. We removed some users with signifantly bad scores (possibly caused due to misunderstanding the instructions). They are asked to score a joke on 1-5, with 1 being likely human-generated, and 5 as likely computer-generated. For our evaluation, we consider a score of 1,2 as human and 3,4,5 as computer - since 3 implies they are not convinced it's a human generated joke.

3.2 Quality and Novelty of Generated Jokes

The generated jokes were novel across all three models and were also able to generate coherent output, even if not necessarily jokes. This is a direct result of the template extraction and infilling process. The metrics are shown in Table 1.

Due to the finetuning process, BERT is able to generate funny jokes, simply by filling relevant funny content words in a suitable template. Zephyr arguably performs worse than BERT, as the loss in using relevant humorous content is not compensated enough by the gain in linguistic capability. GPT-4, however, is able to generate clever jokes based on the template. Selected examples are there in Appendix A.

3.3 Results

Table 1: Performance Metrics Comparison

Metric	**GPT-4**	**BERT**
Accuracy	0.633	0.713
Precision	0.437	0.596
Recall	0.544	0.776
F1 Score	0.484	0.674

The results (shown in Table 1) match well with our initial hypotheses, based on around 200 annotations for each model. Annotators were provided a random joke and told to predict whether it is human-generated or computer-generated. Precision shows how many jokes were computer-generated, among all jokes that were labeled as computer-generated. The precision for both the models is somewhat low, which is indicative that the evaluators were assuming low quality jokes are computer made. Recall is a more interesting metric, as it shows how many model-generated jokes managed to convince the evaluator as being human-generated, where GPT-4 performs very well, but BERT shows promise, specially given how the model is not suitable for generation. These results show that our models are able to make convincing and novel jokes, based on simple heuristics that we applied before, and is able to create suitable templates for joke generation.

4 Discussion

4.1 Limitations and Future Work

Through this paper, we attempt to show that jokes have a distinct style that can be build upon to create novel jokes. Future studies could include a more rigorous evaluation involving control templates to validate the effectiveness of the proposed template extraction and infilling method.

This paper shows insight into the nature of related content words, which are used to infill on templates - but it will require further study through which we can understand the mechanism of jokes more directly. Combining such analysis with existing theories of humor can be useful for understanding humor.

4.2 Conclusion

Our research demonstrates the feasibility of generating novel and convincing jokes through a combination of template extraction and advanced infilling techniques. The use of BERT and GPT-4 models has shown that it is possible to automate humor while maintaining the structural integrity of jokes. This approach opens new avenues for humor generation in AI, highlighting the intricate balance between creativity and the preservation of comedic structure.

References

2024. Llm leaderboard. https://toloka.ai/llm-leaderboard/. Accessed: 2024-05-30.

SALVATORE ATTARDO and VICTOR RASKIN. 1991. Script theory revis(it)ed: joke similarity and joke representation model. *HUMOR*, 4(3-4):293–348.

Kim Binsted and Graeme Ritchie. 1994. An implemented model of punning riddles.

Tanishq Chaudhary, Mayank Goel, and Radhika Mamidi. 2021. Towards conversational humor analysis and design.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Aman Gulrajani. 2024. Short jokes dataset. https://github.com/amoudgl/short-jokes-dataset. Accessed: 2024-05-30.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

OpenAI. 2023. Gpt-4 technical report.

Victor Raskin. 1985. *Semantic Mechanisms of Humor*, volume 5.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023. Zephyr: Direct distillation of lm alignment.