Humour Conversation Generation: Exploiting Incongruity-Based Features and Deep Learning Methods

Jingrong Chen, Yubo Xie, and Pearl Pu
School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne
Lausanne, Switzerland
{jingrong.chen,yubo.xie,pearl.pu}@epfl.ch

***Abstract*—This paper introduces the methods to detect and generate humorous conversation based on conversational data gathered from different resources. In order to gather and filter the humorous conversation data, we combined Incongruity-Based Features and a neural-based method to filter the data. To find humorous conversation data, our method is useful. In the later part of our paper, we implemented three main types of text generation models to compare with our finetuned GPT-J model; our finetuned model outperforms the traditional methods like LSTM and Seq2Seq models; It only requires a few datasets compared to methods, but it performs better. It can generate humorous replies according to the context question, just like a human.**

***Index Terms*—Text generation, Humour, natural language processing, Incongruity-Based, Deep learning**

I. Introduction

Humour recognition and humour generation are both important topics in computational Humour. At the current stage of technological evolution, chatbots are already widely used by many companies and organizations [2]. There is a growing demand for an emotional chatbot for users' higher satisfaction in various commercial fields [3]. Humour is highly valued in interactions between people. It is present in social conventions, cultural artefacts, and daily life. However, computational humour has been a challenge since humour is a complex, puzzling and idiosyncratically human form of behavior [4], and a universal definition of humour is hard to achieve because different people hold different understandings of even the same sentence. Besides, humour is always situated in a broader context that sometimes requires much external knowledge to fully understand it. Moreover, some types of humour express the motivation to disparage its target, exert superiority over others, or release repressed aggressive tension [5]. There is no universal definition and theory for humour. Understanding and identifying humour has long been a goal for natural language understanding systems. Many people are trying to discover underlying features of humour, John Allen Paulos cleverly scrutinizes the mathematical structures of jokes, puns, paradoxes, spoonerisms, riddles, and other forms of humour [6], Taylor and Mazlack recognized wordplay jokes based on statistical language recognition techniques, where they learned statistical patterns of text in N-grams and provided a heuristic focus for a location of where wordplay may or may not occur [7]. Valitutti A and Doucet A automated generation of humorous texts by substituting a single word in a given short text [8], and Purandare and Litman analyzed acoustic-prosodic and linguistic features to recognize humour during spoken conversations [9] automatically. Moreover, many attempts in the deep learning methods have also become popular in recent years. Chen and Soo implemented a Convolutional Neural Network (CNN) with an extensive filter size to recognition of different types of humour with benchmarks collected in both English and Chinese languages [10], Mao and Liu utilize BERT, a multi-layer bidirectional transformer encoder which can help learn deep bi-directional representations for automatic humour detection and scoring [11].

This paper aims to generate a humorous response to a given question or sentence, which could be used in chatbot and daily life conversations. We do not restrict our subject to investigations to only jokes, which consists of set-up and punchline [12]. Consider the example: "Why haven't aliens visited our solar system?". The question is considered as the set-up and provides the context for this joke. The punchline, "They looked at the reviews, and we only have one star.", is usually at the end of a joke and produces a laugh. We also try to cover the humour in daily casual conversations. Compare the Figure1 with the following example: Person A: "Do you ever just feel really stupid?" and Person B replies: "Happens to me way too often.". In the structure of the joke shown in Figure1, the punchline usually violates the expectation structure and as for the later example, the conversation is more casual.

In this paper, we first collect our humorous conversation dataset by using assembling methods mixing incongruity-based features and deep learning methods. Then we finetune the collected dataset in GPT-J [14], which has 6 billion parameters, making it one of the most advanced open-source Natural Language Processing models in this writing. GPT-J is

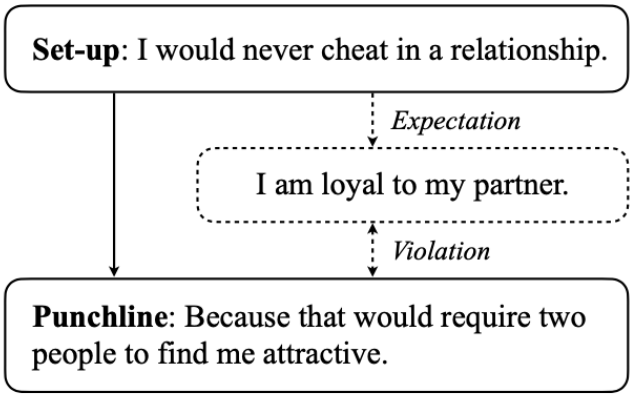

Fig. 1. A joke example consisting of a set-up and a punchline. A violation can be observed between the punchline and the expectation [13]

a natural alternative to OpenAI's proprietary GPT-3, the most potent AI model ever released for text understanding and text generation [15]. We also compare our method with RNN and sequence to sequence model [16]. We use BLEU score, which measure the similarity between the output and the reference. We also conduct a 6-point Likert-type scale human evaluation method.

II. Literature Review

Humour Theory Humour is a fundamental feature of social life and can be used with distinct motivations and targets. Humour is often used in a positive interpersonal manner and in contexts perceived to be playful, safe, non-serious, or, in other words, benign [15]. Humour occurs in all cultures and all individuals throughout the world, nearly every type of interpersonal relationship. The experience of humour appears to be predicated on two cognitive-perceptual processes activated by characteristics of a humour stimulus and the social context in which it is encountered: (1) perception of incongruity and (2) appraisal of incongruity in a non-serious humour mindset [17]. Although there are many debates over humour's characteristics, most contemporary investigators would agree that the perception of "incongruity" is at the heart of the humour experience [17]. Many linguists and psychologists have proposed numerous theories to humour [18]. Rod A. Martin grouped the classic theories into three critical differences: relief, superiority and incongruity. Relief theories focus primarily on interpersonal needs, namely the relief of tension in the form of physiological arousal [19] or "forbidden" sexual and aggressive impulses [20]. Superiority theories propose that the fulfilment lies in the sense of sudden triumph or feeling of superiority over another person or group of people [21]. Moreover, incongruity theories focus instead on the formal object of amusement, often a stimulus event. It is the perception of something incongruous that violates our mental patterns and expectations [22]. Incongruity is now the dominant theory of humour in philosophy and psychology.

Humour recognition The task of humour recognition refers to determining whether a sentence or a conversation contains a certain degree of humour. The challenging part is that, there are different types of humour. Raz Y classifies them into 11 types of humour [23] such as irony, jokes and wordplay. It is almost impossible to design a general algorithm that classifies them [24]. Many scholars are trying to find the underlying features to achieve humour recognition, and many humour Feature Extractions have been proposed. Bucaria C develops his finding on lexical and syntactic ambiguity that result in sources of voluntary or involuntary humour. [24]. Stylistic features include alliteration, rhyming, negative sentiment, adult slang [25] and other features like ambiguity and preliminaries [26] [27] lexical centrality [28]. Moreover, recently, Uncertainty and Surprisal [13] have been used for the humour recognition task. As well as deep learning methods, Chen and Soo [10] implemented a Convolutional Neural Network (CNN) for their humour recognition task. Yubo also used GPT-2 in his humour recognition task. Furthermore, Mao [11] proposed a BERT-based approach for humour detection. Annamoradnejad and Zoghi also proposed Using BERT sentence embedding for humour detection [29].

Humour generation Since the early 1990s, computer scientists and linguists have been trying to create jokes using algorithms [30]. Although there are some attempts to generate humour, no system has been able to imitate human humour and generate jokes like a human today. We can see humour generation as a particular case of automatic text generation. As traditional text generation methods, we can categorize them into two approaches, neural-based and non-neural-based methods: templates and neural networks. Until Yang and Sheng's publication [31] on an LSTM(RNN) for joke production in 2017, all of the previously published systems were template-based [30].

III. Construct the humour generation Dataset

The dataset we used consists of data collected from various sources and measures. We only considered positive examples with precisely two sentences to adapt the dataset for our purpose. Our dataset contains not only the structured dataset like a joke with the set-up and the punchline. It also contains the data form like the daily conversation and comments, which do not necessarily have a joke-like format. We crawled the data from different subreddit such as r/jokes, r/askreddit and r/showerthoughts r/casual conversations. Moreover, we also used collected datasets from humorous conversation screenshots that people posted on-line and we also adopt the data from Orion Weller [32], Issa and Gohar [29]. In total, we have 198,000 row datasets with two sentences and token length less than 156 token size lengths. Then we divided the dataset into 7:2:1 train:validation:test set with random sampling.

To the humorous reply, we filter the dataset using the non-neural-based method combined with the neural method. We proposed using humour indicator method, which means if

the comment's comment has humorous keywords, then this comment is more likely to be a humorous reply.

A. Data Cleaning

For the data we crawl from subreddits, we adopt some methodology from Yubo's paper [13]. We clean the data with the following rules. (1). We restrict the length of the submission and comment to be under 50 (by counting several tokens) and bigger than 2 (2). We only kept the comments whose percentage of alphabetical letters is equal or greater than 75% (3). We discarded punchlines that do not begin with an alphabetical letter (4).We discard the comment if the comment does not have a parent submission or comment (5). We filter our data according to humour comment indicators—the process showed in Figure 2

B. Humour indicator

Humour indicator is the comment of a comment which reflects whether this comment is humorous or not. It consists of the keywords or emoticons or emojis. We crawled r/jokes comments between the date of 2020.0101-2020.1231. We used bi-grams and trig-grams to find some indicators from them. This way, it could be found some patterns like "ha ha" and "ha ha ha". Besides supplements, we searched from the internet and asked native English speakers what a possible reply could be when something is considered funny. To filter the humorous data, we kept the comments that have the child humor indicator comment and they also should equal or above 40 percent of the comments' comment.

C. Neural-based detecting

For the neural-based method, we use two models to filter our data. The first model we used is the Colbert [29] . It is already a pre-trained neural network with eight layers. Then three hidden layers are concatenated in the fourth layer and continue sequentially to predict the single target value. The model structure is shown in Fig 3

As for the second model we used the score of surprisal valued calculated from GPT-2 model [13], a Transformer-based architecture pre-trained model trained on massive data and publicly available online. It is domain independent, thus suitable for modeling various styles of English text. We calculate the surprisal value [13] defined as:

$$\begin{aligned} S(x, y) &= -\frac{1}{|y|} \log p(y \mid x) \\ &= -\frac{1}{|y|} \sum_{i=1}^{n} \log v_i^{y_i} \end{aligned} \tag{1}$$

The following text denotes the question/submission as x and the reply as y. We are interested in quantity regarding the probability distribution p(y|x): surprisal, which is elaborated in Equation 1.

For the final filtered data, we examined humans. We found out that our method could filter out 27/50 percentage humorous comments compared to random searching, in which only 3/50 percent of the data is humorous. Our method is effective in finding humorous datasets for later stages for humour generation.

D. Ocr data collecting

We also collected the data from the screenshots shown in Figure 4 that people published online from different resources, including Twitter, Tumblr and Facebook. For convenience, we only focused on those with two-sentence sentences by using the OCR technique to enrich our dataset. The data is from various sources such as Twitter, iMessage, Facebook and Instagram. These data are already tagged as humorous, so we do not need to examine them. Besides, they are from daily conversations and interactions, so they are of good quality.

IV. Deep Learning Methods

Here we compared and examined the two popular deep learning models as the baselines to compare our model.

A. LSTM

First, we check on the traditional deep learning framework: text generation tasks based on a particular type of recurrent neural network (RNN) [41] called LSTM [42], like traditional deep learning model CNN [43] GNN [44]. These traditional methods tend to seek stronger feature extractors to achieve higher steganalysis effects. RNN helps sequential model data; each timestep generates new output based on the current input and past output. We can use the characteristic of RNN to do humour generation.

However, it is challenging to train standard RNNs to solve problems that require learning long-term temporal dependencies. The LSTMs are called Long Short Term Memory (LSTM), a special type of RNN capable of learning long-term dependency problems. It includes a 'memory cell' that can maintain information in memory for long periods. Then it uses gates to deal with the vanishing and exploding gradient problem in RNNs. So, LSTM can handle information in memory for an extended period compared to RNN. Their structures are shown in Figure 5 and Figure 6.

All recurrent neural networks have the form of a chain of repeating modules of a neural network. In standard RNNs, this repeating module will have a straightforward structure, such as a single tanh layer. LSTM also has a chain of repeating modules but four layers in each module, with serval sigmoid gates to protect and control the cell state. Then it has two data flows to the next unit. The top straight-line runs down the entire chain, with only minor linear interactions. Then the other flow to the next module contains the current state interacted with the first flow.

We implemented a simple LSTM chain structure model with 128 recurrent layers. The model was trained with GPU with 30k epochs.

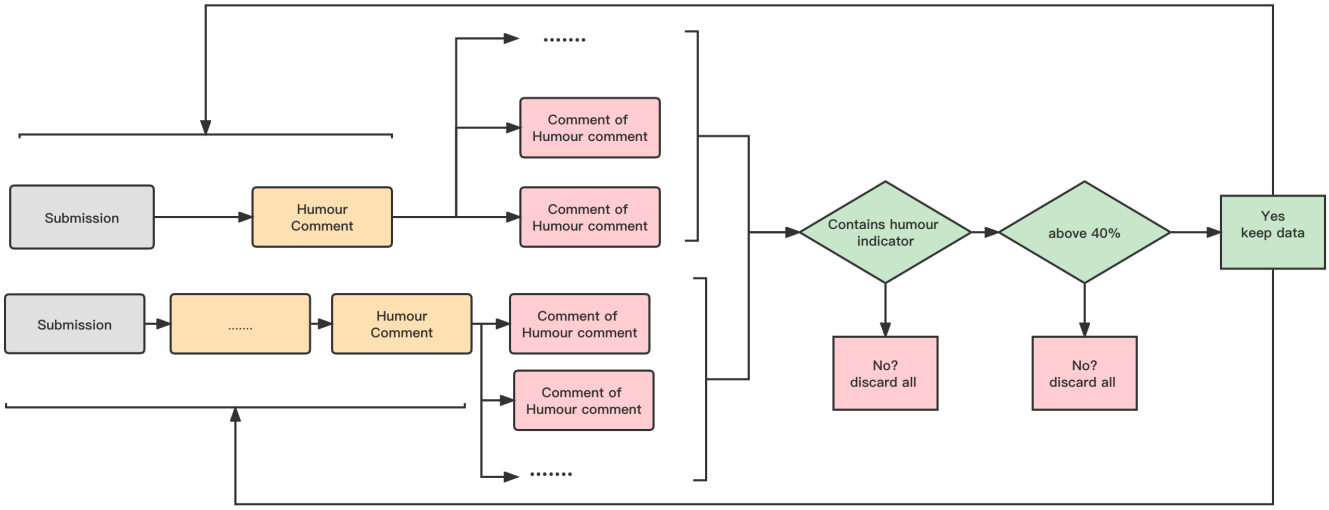

Fig. 2. Indicator filter process

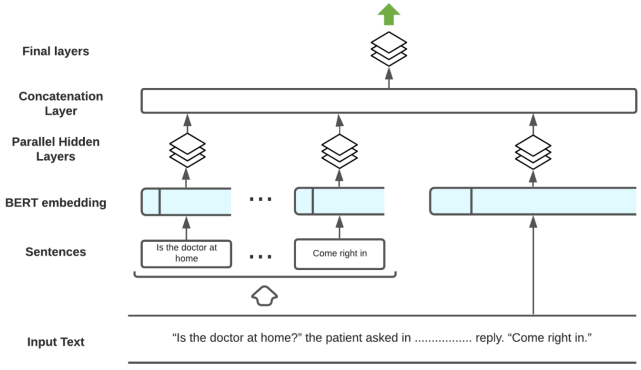

Fig. 3. Colbert model Components [29]

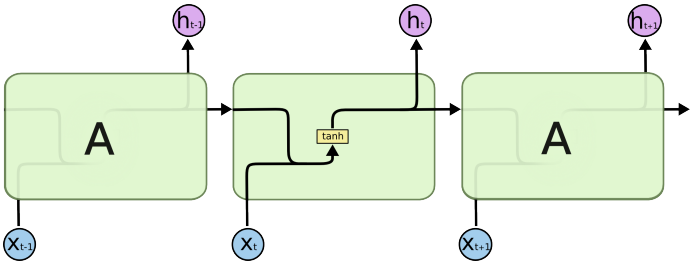

Fig. 5. RNN

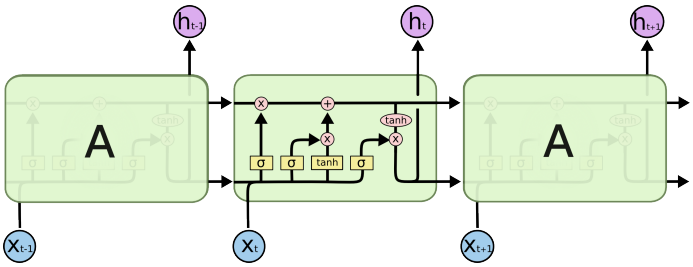

Fig. 6. LSTM

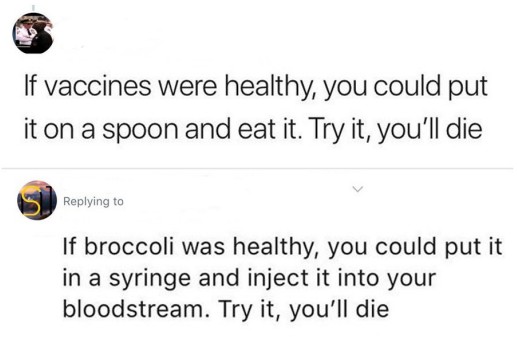

Fig. 4. OCR data example

B. *Seq2Seq*

The seq2seq (sequence to sequence) model is a type of encoder-decoder deep learning model that uses recurrent neural networks like LSTM to generate output. seq2seq networks have an encoder accepting language as input and outputting state vectors and a decoder accepting the encoder's final state and outputting possible translations. It is commonly used for text translations, but it is also good for text generation since it has a good general ability. There are two main components [49] for the network:

Encoder: It uses deep neural network layers and converts

the input words to corresponding hidden vectors. Each vector represents the current word and the context of the word.

Decoder: It takes the output of the encoder as input. It also uses its hidden states and the current word to produce the next hidden vector and predict the next word.

The model we implemented has a structure like Figure 8. We use 4 GRU units as encoders instead of LSTM units. GRU structure in Figure 7 performs similar to LSTM but computational cheaper. Compared to LSTM, GRU has one gate less than LSTM. It is only three times the number of parameters as RNN, while LSTM is four times. Considering a large number of GRU units and layers will effectively reduce the number of parameters.

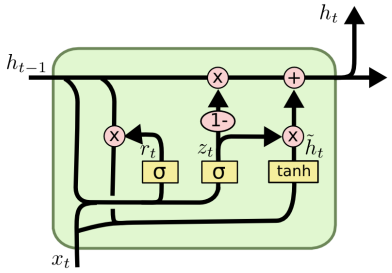

Fig. 7. GRU

C. Fine-tuning GPT j-6b

For the last two models above, the data limits the applicability of language models. Collecting a sizeable supervised training dataset for many tasks is difficult, especially when the process has to be repeated for each new task. pre-traine plus finetune method is becoming popular in natural language processing. The model is usually designed to be very large to learn more information and have a good generalization power during pre-training; for example, GPT2 has 1.5 billion parameters. Then it can be finetuned according to a specific task. One good advantage of the pre-traine plus finetune method is that limited data can be outperformed by the traditional NLP model, which requires a hundred times of data.

The new GPT model is GPT-3, the third Generative Pre-trained Transformer that uses 175 billion parameters. It is trained on Microsoft's Azure's AI supercomputer (Scott 2020). It is a costly training , estimated around $12 million. However, OpenAI licenses its GPT-3 exclusively. We chose the alternative transformer [35] model GPT-J6b [33] trained on the pile dataset, an 825 GiB English text corpus targeted at training large-scale language models. [34], GPT-J may be the most potent open-source Natural Language Processing model today with 6 billion parameters and it is the only open-source alternative competing with GPT-3. GPT-J outperforms GPT3 in code generation tasks. Then GPT-J is the best-performing publicly available Transformer LM in terms of zero-shot performance on various down-streaming tasks.

The model consists of 28 layers with a model dimension of 4096 and a feedforward dimension of 16384. The model dimension is split into 16 heads, each with a dimension of 256. Rotary position encodings (RoPE) [37] were applied to 64 dimensions of each head. The model is trained with a tokenization vocabulary of 50257, using the same set of BPE's as GPT-2 [36] and GPT-3 [15].

We finetuned the model with our dataset using Google Cloud TPU [38] v3-8 and using Google Cloud bucket to store our weights data and pre-trained model.

1) Dataformat: Before each question (submission) and answer we added prompt text before each text and separate our text using $< |endoftext| >$ the dataset shall be looked like the example on below:

User: Can I tell you a joke about vegetables?

Humorous reply: Only if you say peas.$< |endoftext| >$

Then the dataset is tokenized using GPT2Tokenizer [39], to construct a "Fast" GPT-2 BPE tokenizer (backed by HuggingFace's tokenizers library), using byte-level Byte-Pair-Encoding. Then we pad our token array to the same length and save them into tfrecord, a simple format for storing a sequence of binary records.

Instead of training on the original 2048 sequence length data, which includes several data rows on each token sequence, we trained max token length of 156 size length and padded the ones that are not 156 lengths. We trained our data for 10k epochs with 8 cores TPU.

V. Measure matrix

Although many systems have been developed so far, a standardized methodology for assessing the quality of such systems is still missing [30]. The majority of approaches were evaluated using user studies where actual humans rated the generated jokes on a Likert scale that typically ranges from 0 (not a joke) to 5 (really funny) [40]. Here we introduce N-gram-based matrix BLEU score and Model-based metric BERT score.

1) BLEU: Bilingual Evaluation Understudy [45], although it is made for comparing a candidate translation of text to one or more reference translations, it can be used to evaluate text generated for a suite of natural language processing tasks. BLEU is a weighted geometric mean of N-gram precision scores, N-gram precision defined as

$$\mathsf{p}_n = \frac{\sum_{\mathbf{s}} \min\left(C(\mathbf{s},\mathbf{y}), C\left(\mathbf{s},\mathbf{y}\ \right)\right)}{\sum_{\mathbf{s}} C(\mathbf{s},\mathbf{y})} \tag{2}$$

Then the BLEU score define as:

$$\mathsf{BLEU} = \mathsf{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{3}$$

where BP is the brevity penalty, which penalizes sequences that are too short.

$$\mathbf{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1\ \ r=c)} & \text{if } c \leq r \end{cases} \tag{4}$$

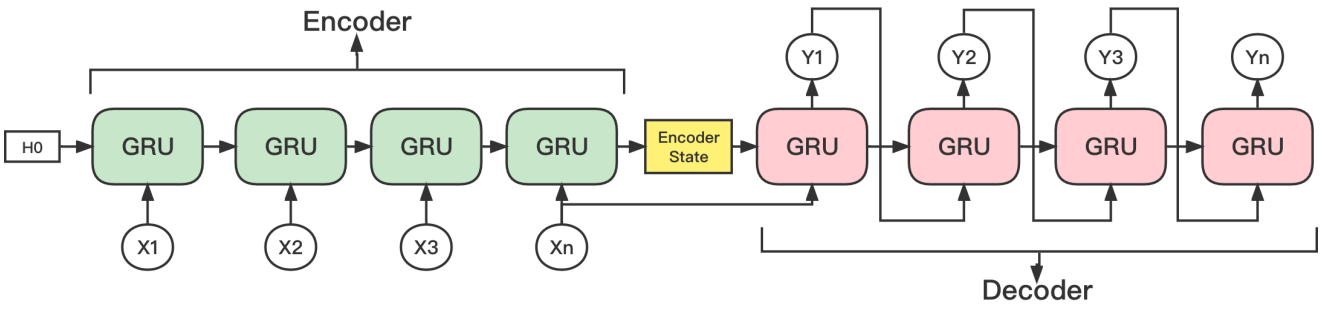

Fig. 8. Seq2Seq model structure

c is the length of the candidate translation and r be the effective reference corpus length.

2) BERTscore: BLEU and other N-gram-based metrics like ROUGE Score [46] and METEOR [47] have a critical disadvantage which only uses the sole characteristic of word matching. To retrieve the actual semantics of a sentence, BERTScore leverages the pre-trained contextual embeddings from BERT and matches words in candidate and reference sentences by cosine similarity [48]. The complete BERT score metrics define as:

$$R_{\text{BERT}} = \frac{1}{|x|} \sum_{x_i 2 x} \max_{\hat{x}_j 2 \hat{x}} x_i^{\top} \hat{x}_j \tag{5}$$

$$P_{\text{BERT}} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j 2 \hat{x}} \max_{x_i 2 x} x_i^{\top} \hat{x}_j \tag{6}$$

$$F_{\text{BERT}} = 2 \frac{P_{\text{BERT}} \cdot R_{\text{BERT}}}{P_{\text{BERT}} + R_{\text{BERT}}} \tag{7}$$

X is the reference, and $\hat{x}$ is the candidate, each token is matched to the most similar token in the other sentence.

3) Human rate metric: To get a fair evaluation, we asked five English speakers of different cultural backgrounds to rate our model generated result sample from the test data sets. The raters were asked to rate each joke on a 6-point Likert-type scale: 0 (nonsense syntactically), 1 (not relate), 2 (not funny), 3 (somewhat funny), 4 (funny), 5 (really funny)

VI. Experiments

A. Result

When we examined the output of each model, we found out all three models did not generate a close answer to the ground truth answer. Here, we only chose BLEU and human rate metrics as our measurement metrics.

From the table I, we can see that all three models are not close to the ground truth. The results for LSTM and Seq2Seq model do not generate a humorous reply, even though we train on the pre-trained model, which trained with 304 GB Reddit comments. Then for the LSTM and Seq2Seq, we need more dataset, at least dozens of GB data, to generate syntactically correct data. We trained on data we collected (16MB size of the train.txt). Then the results of LSTM and Seq2Seq model were usually syntactically wrong and non-related to the answer. As for the Finetuned GPT-J model, we can see that the model has great generalizing power. With only 16MB of data, we can train and let the model generate the correct and topic-related data. From the human rate result, we only have an average of 2 syntactically not correct and 5 topics not related out of 40. While the LSTM and Seq2Seq model has more than 17 and 14 syntactically incorrect result with 19 and 13 unrelated replies to the question. The finetuned GPT-J model outperforms the other two models in all ways.

TABLE I
Model Score Results

Table Head	**Table Column Head**	
	BLEU	***Human rate***
LSTM	1.37	0.3433
Seq2Seq	1.40	0.8788
FineTuned GPT-j	1.68	2.2682

B. Case study

Here we look at the reply generated by different models, the question is “User: Why will a Tesla scandal be exciting?”. We can see that LSTM generated a syntactically wrong reply. seq2seq model is better but the reply is too general. It is not humorous. With the finetuned GPT-J model. The answer is quite humorous and relates to the topic very well.

TABLE II
GroundTruth and Generated Replies

Model	reply
GroundTruth	elongate
LSTM	The explosion is not the property of the explosion.
Seq2Seq	Hahaha, I do not know what you are talking about
FineTuned GPT-j	Because Elon Musk will be charged with battery.

VII. Conclusion

In this paper, we present several methods to detect humorous conversations. We combined neural and non-neural methods to detect and classify humorous conversations. Then, we crawled and collected our dataset from different kinds of resources, mainly from Reddit submissions and comments. We also gather data in different forms, such as pictures and screenshots with a wide range of daily topics. After preprocessing and cleaning, we combined incongruity-based features such as superficial value and neural-based method value to filter the data in order to gather the data we needed. After we gathered humorous conversation data, we also discussed the possible solution of humour generation evaluation metrics and proposed a novel human rating metrics. We implement the three main types of text generation models and compare their performances. The finetuned GPT-j model outperforms the LSTM and Seq2Seq models in all aspects. It requires less training data but with better results. It has excellent generalizing power and can be customized to complete different topic-specific tasks. We can see that the pre-train plus finetune method is very promising in text generation or, even more, humour generation field. In future work, the finetuned model could be customized to fit the conversational form of text generation. We could also gather more data from different resources. Also, Prompt Training of our finetuned GPT-J model will be a good idea.

Acknowledgment

Much thanks to professor Pearl Pu and Yubo Xie's help, as well as the advice from the HCI PHDs. We would also like to thank Google Cloud platform for providing one-month free access to their TPUs.

References

[1] Attardo, Salvatore. Linguistic theories of humour. Vol. 1. Walter de Gruyter, 2010.

[2] Eleni Adamopoulou, Lefteris Moussiades, Chatbots: History, technology, and applications, Machine Learning with Applications, Volume 2, 2020, 100006, ISSN 2666-8270,

[3] D. Lee, S. -H. Han, K. -J. Oh, and H. -J. Choi, "A Temporal Community Contexts Based Funny Joke Generation," 2017 18th IEEE International Conference on Mobile Data Management (MDM), 2017, pp. 360-365, DOI: 10.1109/MDM.2017.62.

[4] Ritchie, G. (2003). The Linguistic Analysis of Jokes (1st ed.). Routledge. https://doi.org/10.4324/9780203406953

[5] Chan YC, Hsu WC, Liao YJ, Chen HC, Tu CH, Wu CL. Appreciation of different styles of humour: An fMRI study. Sci Rep. 2018;8(1):15649. Published 2018 Oct 23. doi:10.1038/s41598-018-33715-1

[6] Paulos J A. Mathematics and humour [M]. University of Chicago Press, 2008.

[7] Taylor J M, Mazlack L J. Computationally recognizing wordplay in jokes[C]//Proceedings of the Annual Meeting of the Cognitive Science Society. 2004, 26(26).

[8] Valitutti A, Doucet A, Toivanen J M, et al. Computational generation and dissection of lexical replacement humour[J]. Natural Language Engineering, 2016, 22(5): 727-749.

[9] Purandare A, Litman D. humour: Prosody analysis and automatic recognition for f* r* i* e* n* d* s[C]//Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing. 2006: 208-215.

[10] Chen P Y, Soo V W. humour recognition using deep learning[C]//Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: Human language technologies, volume 2 (short papers). 2018: 113-117.

[11] Mao J, Liu W. A BERT-based Approach for Automatic humour Detection and Scoring[C]//IberLEF@ SEPLN. 2019: 197-202.

[12] Bright W. International encyclopedia[J]. Psychology, 1992, 9: 151.

[13] Xie Y, Li J, Pu P. Uncertainty and surprisal jointly deliver the punchline: Exploiting incongruity-based features for humour recognition[J]. arXiv preprint arXiv:2012.12007, 2020.

[14] Wang B, Komatsuzaki A. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model[J]. 2021.

[15] Florida L, Chiriatti M. GPT-3: Its nature, scope, limits, and consequences[J]. Minds and Machines, 2020, 30(4): 681-694.

[16] Sriram A, Jun H, Satheesh S, et al. Cold fusion: Training seq2seq models together with language models[J]. arXiv preprint arXiv:1708.06426, 2017.

[17] Martin R A, Ford T. The psychology of humour: An integrative approach[M]. Academic Press, 2018.

[18] Keith-Spiegel P. Early conceptions of humour: Varieties and issues[J]. The psychology of humour: Theoretical perspectives and empirical issues, 1972: 4-39.

[19] Spencer H. The social organism[J]. Westminster Review, 1860, 73(143): 90-121.

[20] Freud S. Three essays on the theory of sexuality (1905)[M]//The standard edition of the complete psychological works of Sigmund Freud, Volume VII (1901-1905): A case of hysteria, three essays on sexuality and other works. 1953: 123-246.

[21] Lintott S. Superiority in humour theory[J]. The Journal of Aesthetics and Art Criticism, 2016, 74(4): 347-358.

[22] Morreall J. Philosophy of humour [J]. 2012.

[23] Raz Y. Automatic humour classification on Twitter[C]//Proceedings of the NAACL HLT 2012 student research workshop. 2012: 66-70.

[24] Yang D, Lavie A, Dyer C, et al. humour recognition and humour anchor extraction[C]//Proceedings of the 2015 conference on empirical methods in natural language processing. 2015: 2367-2376.

[25] Mihalcea R, Strapparava C. Making computers laugh: Investigations in automatic humour recognition[C]//Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing. 2005: 531-538.

[26] Reyes A, Rosso P, Buscaldi D. From humour recognition to irony detection: The figurative language of social media[J]. Data & Knowledge Engineering, 2012, 74: 1-12.

[27] Cattle A, Ma X. Recognizing Humour using word associations and humour anchor extraction[C]//Proceedings of the 27th international conference on computational linguistics. 2018: 1849-1858.

[28] Radev D, Stent A, Tetreault J, et al. humour in collective discourse: Unsupervised funniness detection in the new yorker cartoon caption contest[J]. arXiv preprint arXiv:1506.08126, 2015.

[29] Annamoradnejad I, Zoghi G. Colbert: Using bert sentence embedding for humour detection[J]. arXiv preprint arXiv:2004.12765, 2020.

[30] Amin M, Burghardt M. A survey on approaches to computational humour generation[C]//Proceedings of the 4th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature. 2020: 29-41.

[31] Ren, He and Quan Yang. "Neural Joke Generation." (2017).

[32] Weller O, Seppi K. humour detection: A transformer gets the last laugh[J]. arXiv preprint arXiv:1909.00252, 2019.

[33] Wang B, Komatsuzaki A. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model[J]. 2021.

[34] Gao L, Biderman S, Black S, et al. The pile: An 800GB dataset of diverse text for language modelling [J]. arXiv preprint arXiv:2101.00027, 2020.

[35] Vaswani A, Shazier N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.

[36] Lagler K, Schindelegger M, Böhm J, et al. GPT2: Empirical slant delay model for radio space geodetic techniques[J]. Geophysical research letters, 2013, 40(6): 1069-1073.

[37] Su J, Lu Y, Pan S, et al. Reformer: Enhanced transformer with rotary position embedding[J]. arXiv preprint arXiv:2104.09864, 2021.

[38] Wikipedia contributors. "Tensor Processing Unit." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 5 Jun. 2022. Web. 8 Jun. 2022.

[39] [Transformers: State-of-the-Art Natural Language Processing](https://aclanthology.org/2020.emnlp-demos.6) (Wolf et al., EMNLP 2020)

[40] Binsted K, Ritchie G. An implemented model of punning riddles[R]. The University of Edinburgh, Department of Artificial Intelligence, 1994.

[41] Cho K, Van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation[J]. arXiv preprint arXiv:1406.1078, 2014.

[42] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.

[43] Gu J, Wang Z, Kuen J, et al. Recent advances in convolutional neural networks[J]. Pattern Recognition, 2018, 77: 354-377.

[44] Li J, Li S, Zhao W X, et al. Knowledge-enhanced personalized review generation with capsule graph neural network[C]//Proceedings of the 29th ACM International Conference on Information & Knowledge Management. 2020: 735-744.

[45] Papineni K, Roukos S, Ward T, et al. Bleu: a method for automatic evaluation of machine translation[C]//Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002: 311-318.

[46] Lin C Y. Rouge: A package for automatic evaluation of summaries[C]//Text summarization branches out. 2004: 74-81.

[47] Banerjee S, Lavie A. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments[C]//Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization. 2005: 65-72.

[48] Zhang T, Kishore V, Wu F, et al. Bertscore: Evaluating text generation with bert[J]. arXiv preprint arXiv:1904.09675, 2019.

[49] Wadhwa, Mani (2018-12-05). "seq2seq model in Machine Learning". GeeksforGeeks. Retrieved 2019-12-17.