HumourSpace: A Novel Framework for Quantification and Characterisation of Humour

Midhush Manohar T.K.[1] , **Ninaad R. Rao**[1] , **Nishant Ravi Shankar**[1] and **Ramamoorthy Srinath**[2]

[1]Student, PES University, Bengaluru, India

[2]Professor, PES University, Bengaluru, India

{mimosk25, ninaadrrao, nishantravishankar, ramamoorthysrinath}@gmail.com

Abstract

Humour, a human-trait, is an integral ingredient in day-to-day communication. This paper attempts to extend the processing of this complex and highly subjective quality to digital entities. The paper does this by introducing HumourSpace, a model-agnostic framework which transforms humorous sentences to a 9-dimensional feature space, with each dimension representing a computational linguistic feature. This is followed by objective evaluation of different strains of humour using unsupervised learning. Using this framework, the paper introduces a Personalised Rating Mapper (PRM) that seeks to capture an individual's affinity towards a particular strain of humour. To validate the improvements in humour content personalisation, experiments involving user-surveys have been conducted, which conclude that 73% of users agree more with the humour content recommended by the PRM as compared to 20.36% for crowdsourced ratings. By reducing a machine learning model's dependency on average crowdsourced ratings, this framework is a promising approach to improve the quality of human-computer interaction in a predictable and quantifiable manner.

1 Introduction

Successfully capturing and characterising humour is one of the most challenging AI-complete problems that is being tackled in the field of Natural Language Processing (NLP). The notion of AI-Completeness [Raymond, 1991] captures the "hardness" of an AI problem similar to what NP-Completeness does for algorithmic tractability. Considering humour, the desirable capabilities of an AI system include both understanding the context of the conversation and interpreting specific humour preferences. Further, such a system must also have the ability to detect and react to humour. Technical advances in end-to-end training of deep neural networks have led to significant progress in various theoretical and applied domains of NLP, including Information Extraction, Ontological Engineering, Speech Processing, and Statistical NLP. Most of these domains have achieved a considerable amount of success because of their mathematical origins. However, this has not been the case with humour in Natural Language (NL) owing to its highly nuanced and subjective nature.

Computational Humour is a field of study in the branches of Computational Linguistics and AI which employs algorithmic models in humour research. The first "computer model of a sense of humour" was proposed by [Suslov, 1992]. This paper suggested that to detect humour from a piece of text, there must exist a possibility of a specific malfunction, along with the necessity that one must instantly delete the unintended interpretation of the text from one's consciousness. Simultaneously, one must also understand the interpretation from a humorous context. This malfunction created above is strongly correlated to the incongruity-resolution theory.

To explore the subjective nature of humour in NL, two different experiments were conducted, the first of which was estimating the quality of humour using supervised multi-class classification on the datasets referenced in Section 4. The quality of humour was defined as the average crowdsourced ratings given to humour content rounded off to the nearest whole number, following the Likert scale system (1-5). Likert-scale is a psychometric scale that is commonly employed in questionnaires. Pre-trained Universal Sentence Encoder [Cer *et al.*, 2018] was used for embedding the humour content. Results of various classifiers such as Support Vector Machine (SVM), Random Forest Classifier (RFC), Hierarchical Attention Network (HAN) [Yang *et al.*, 2016], and XLNet [Yang *et al.*, 2019] were similar, and the accuracy was close to the majority-class model. Here, a majority-class model is a model which predicts only the majority-class for every input. The accuracy of such a model is equivalent to the proportion of the majority class in the data it was trained on.

The second experiment was an analysis based on user ratings collected from 133 responses. These responses were compared against crowdsourced ratings, with the method of evaluation being an Inter-Annotator Agreement Percentage (IAAP). It is defined as the relative frequency of the average rating as follows:

$$Inter - AnnotatorAgreement = \frac{Freq(A)}{T} \quad (1)$$

$$where:$$
$$T = Total\ number\ of\ responses$$
$$A = Average\ rating$$

Crowdsourced Rating	IAAP
1	24.38
2	19.82
3	24.30
4	19.29
5	14.00

Table 1: Inter-Annotator Agreement Percentage (IAAP) on a Likert scale.

This experiment showed that the overall inter-annotator agreement between the users and crowdsourced ratings was 20.36%, further depicted in Table 1. This was in line with the results from [Winters *et al.*, 2018], where an agreement of only 41.36% was obtained for the human-created jokes.

The results of these experiments imply that a high amount of subjectivity and bias goes behind evaluating humour, which ends up impacting any machine learning model's performance. To tackle this problem of subjectivity, the paper proposes a novel quality estimation pipeline that objectively evaluates humour based on various computational linguistic features. The humorous texts are transformed to a 9-dimensional vector space, which are clustered using unsupervised learning to represent the different strains of humour. This is followed by PRM, which captures an individual user's tastes in humour, thus attempting to provide better humour content to users. The objective approach also helps in establishing a ubiquitous computational ranking system for various pieces of humorous texts, to ensure a normalised method of comparison as opposed to the different schemes used by various websites. Experiments evaluating the proposed pipeline and the dependency of domain in each humour cluster have been discussed in Section 7.

2 Background

With the proliferation and widespread increase in the usage of AI and NLP, there is much research being carried out on ways to make AI systems more friendly and explainable. Products including digital assistants in smart devices (Apple's Siri and Amazon's Alexa), chatbots for social good adopted by organisations (UNICEF), and interactive agents deployed by major business entities around the world show that NLP has seen, and continues to see rapid improvements in speed and accuracy. However, most NLP solutions are confined to automating tasks, and the human-chatbot conversation is programmatic in nature, with the digital entity responding curtly in a few sentences. The Turing Test, described by [Turing, 1950; Turing, 2009], is a common baseline experiment that attempts to provide a differentiating barrier between humans and synthesised intelligence. An artificial system that subsumes human emotions in its processing to cross this barrier, would have come closer to solving the AI-complete problem. Humour, an essential part in everyday conversation, is no exception to the difficulties associated with clearing the Turing Test, a necessary but not sufficient measure for evaluating machine intelligence. The high complexity in successfully characterising humour is because of its subjective nature. Virtual assistants gaining the ability to characterise and inherently incorporate humour can drastically enhance the quality of a conversation, resulting in an improved user experience.

3 Related Work

In Computational Humour, several advancements have already been made, such as detecting one-liners and innuendos [Kiddon and Brun, 2011], generating analogies using the power of big data [Petrović and Matthews, 2013], the creation of puns using the local-global principle [He *et al.*, 2019], and more [Binsted, 1996; Venour, 2000; McKay, 2002; Stock and Strapparava, 2005; Ritchie, 2005; Shahaf *et al.*, 2015; Dybala, 2008].

To recognise humour, [Cai and Ehrhardt, 2013] considered computational linguistic features. Fifty-dimensional word embeddings were trained and used along with the Parts Of Speech (POS) tag of the previous and the next word to capture context. [Cai and Ehrhardt, 2013] used this combined feature vector and trained a single layer neural network for humour recognition. [Yang *et al.*, 2015] focused on identifying semantic structures behind humour from four perspectives, namely incongruity, ambiguity, interpersonal effect, and phonetic style. They proposed a set of features for each of the semantic structures. Random Forest Classifier (RFC) was trained to classify the text into humour vs. non-humour. To enable humour in a sentence, [Yang *et al.*, 2015] also introduced the notion of anchor extraction. [Chen and Soo, 2018] detected humour using Convolutional Neural Networks (CNNs) and Highway Networks, the latter of which was introduced by [Srivastava *et al.*, 2015] to help unimpeded information flow across several hidden layers. However, all of these papers focus on detecting humour by classifying content as humorous or non-humorous.

[Winters *et al.*, 2018] introduced an algorithm that learns humour from a set of jokes that are human-rated. A website called JokeJudger was created so that users can rate a joke for the template: "I like my X like I like my Y, Z". The rating of jokes was done on the Likert scale. The features used in [Winters *et al.*, 2018] were inspired by [Ritchie, 1999]. Even though [Winters *et al.*, 2018] focuses on recognising different levels of humour, it is dependent on crowdsourced ratings, which are highly subjective in nature. As an extension to [Winters *et al.*, 2018], [Winters *et al.*, 2019] identified useful humour metrics based on humour theory. [Winters *et al.*, 2019] used metrical schemas to associate lexical relations on words for the purpose of humour recognition.

In contrast to the work that has been previously done, our system does not learn humour from crowdsourced ratings as the underlying classification system tends to be noisy and biased. The bias could be in terms of the number of users looking at a given text or simply context based bias. To tackle this, our system uses computational linguistic features in an unsupervised manner to objectively evaluate humour. This helps computers understand humour in an unbiased manner.

4 Data Collection and Preprocessing

The prerequisites to our dataset include domains of the jokes (to distinguish between different kinds of humour), types of jokes such as one-liner, short-paragraph, and question-answer

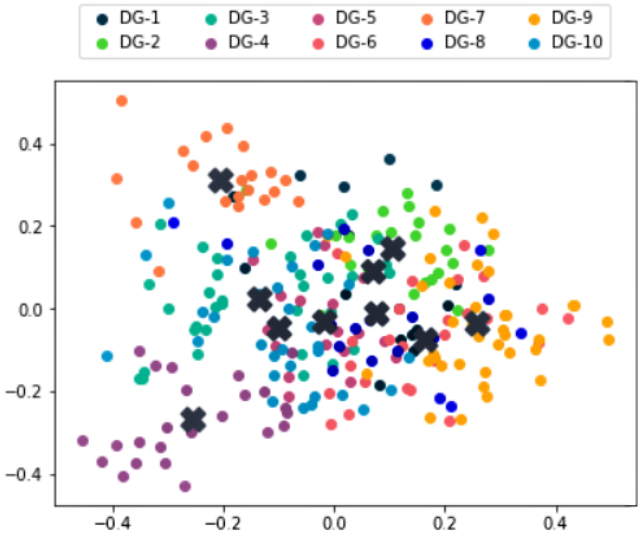


Figure 1: Data representation based on different Domain Groups.

(to include a wide range of generalized humour content), as well as ratings provided on a Likert scale (to obtain the content's humour level). To obtain a sizeable proportion of training, validation and testing inputs, the data for this paper has been taken from various sources. The humorous texts comprise of web-scrapped data from websites [1] as well as a compilation of jokes from [Pungas, 2017]. These sources have content tagged with domains and crowdsourced ratings except for Reddit [2], which lacks domain tags. The non-humorous texts have been collected from [Misra, 2018] and Wikipedia. When this data was aggregated, it resulted in 251 domains. On inspection, a considerable number of these domains were found to be similar, which necessitated their grouping. To group the overlapping domains, different approaches such as Universal Sentence Encoder (USE) based cosine similarity, GloVe based semantic similarity and ELMo embeddings based similarity were attempted. However, these techniques performed similarly and resulted in poor segregation of domains into Domain Groups (DGs), as depicted in Figure 1. Therefore, manual clustering of these domains was performed, which resulted in 12 primary domains as shown in Table 2.

Before application, the data was cleaned in the following steps:

- Removal of emojis and non-ASCII characters,
- Expansion of contractions (for example: "She's" to "She is"),
- Conversion of text into lowercase, and
- Tokenisation of text at both sentence and word levels.

Due to the presence of a large corpus of unlabelled text, domain classification has been performed for automated domain tagging. Models such as a Feed-Forward Neural Network (FFN), a Random Forest Classifier (RFC), a Support Vector Machine (SVM), and a Hierarchical Attention Network (HAN) have been compared. USE [Cer *et al.*, 2018], the state of the art sentence encoder, is used as the embedding layer for FFN, RFC and SVM, as these models require sentence representations. However, it has been found that HAN, which uses GloVe based word embeddings, gives the best results. Table 3 specifies the performance of the different models.

[1] https://www.ajokeaday.com, https://onlinefun.com, https://unijokes.com and http://www.jokesoftheday.net

[2] https://www.reddit.com

Domain	Data size
Animal	9287
Bar	9834
Event/Day	7803
Human	27579
Inappropriate	7148
Politics	43717
Profession	27362
Relationship	33284
Religion	7908
Sports	23349
Technology	9266
Transport/Location	10714

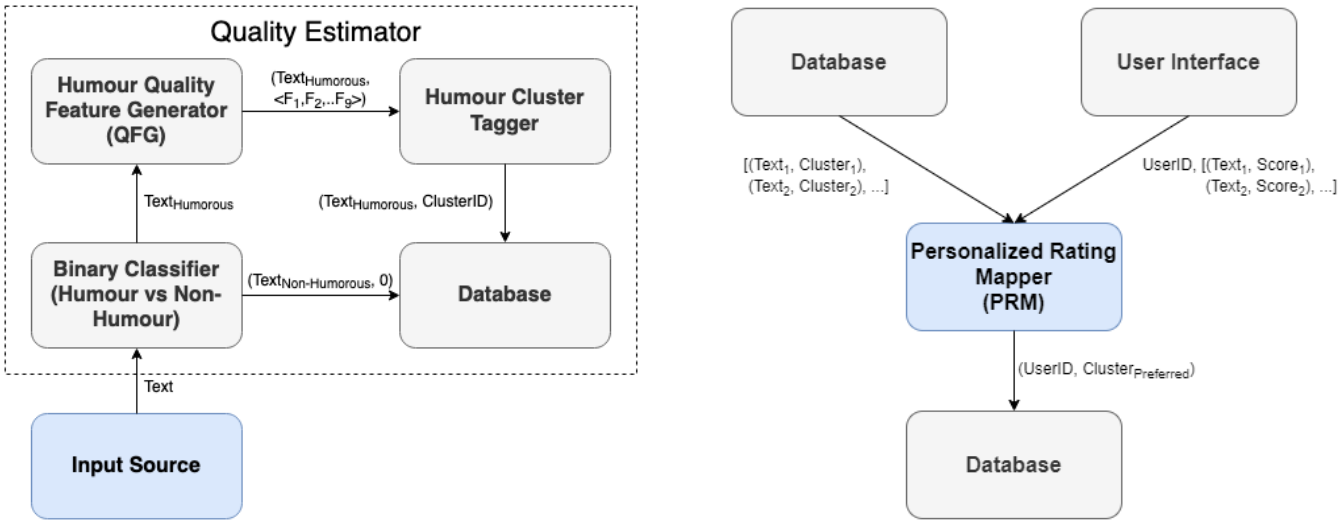
Table 2: Domain distribution of the dataset.

	Train			
	Accuracy	**Recall**	**Precision**	**F-1 Score**
FFN + USE	0.65	0.65	0.65	0.65
RFT + USE	0.63	0.63	0.64	0.63
SVM + USE	0.69	0.72	0.65	0.67
HAN + GloVe	0.78	0.76	0.78	0.77
	Test			
	Accuracy	**Recall**	**Precision**	**F-1 Score**
FFN + USE	0.54	0.64	0.43	0.44
RFT + USE	0.53	0.61	0.40	0.42
SVM + USE	0.69	0.71	0.65	0.67
HAN + GloVe	0.78	0.76	0.78	0.77

Table 3: Comparison of different classifiers for domain classification.

5 System Design

The pipeline used in this paper include linguistic feature generation, clustering the text based on the linguistic features and creating a personalised rating system that caters to an individual's humour preferences. Sections 5.1 to 5.2 describe the sub-modules depicted in the dataflow diagram (Figure 2).



(a) Quality Estimation.

(b) Personalised Rating Mapper (PRM)

Figure 2: Dataflow diagram.

5.1 Quality Estimation

Quality Estimation takes text as input and estimates the quality of humour content, which is further fine-tuned to a particular user. The first stage of this pipeline is a binary classifier which tags the text as humorous or non-humorous, as explained in Section 7.1. Texts tagged as non-humorous receive a rating of 0. The texts which get tagged as humorous, i.e., with a rating of 1, are fed into the Humour Quality Estimation module for further processing. In this module, the input is processed by the Quality Feature Generator (QFG), which generates linguistic features. These features are then used to estimate humour using the Unsupervised Quality Estimator (UQE). This has been further discussed in subsequent parts of this section.

Quality Feature Generator (QFG)

The QFG outputs 9-dimensional feature vectors, and each component of a vector specifies the value of humour-related features. Some of these features including Obviousness, Compatibility, Inappropriateness, Absurdity and Conflict are inspired by Ritchie's incongruity-resolution theory [Ritchie, 1999]. This theory stems from the principle that the process of disambiguation between the obvious interpretation and the hidden interpretation of a piece of text can be a source of humour. It states that the meaning of the punchline conflicts with the obvious and primary interpretation, but is compatible with, and even evokes the hidden interpretation. The other features, such as the HMM and N-Gram probability try to capture the "rarity" in the sequencing of words in a piece of text, which could be different for humorous and non-humorous texts. Following the computation of the 9-dimensional feature vectors, the output from QFG is then used in the unsupervised quality estimation pipeline. The following paragraphs describe the implementation of these features in detail. An example of the values of these feature for a sample sentence has been shown in Table 4.

Obviousness. This helps measure how likely it is for a given sentence to occur, based on the frequency of individual tokens in a chosen corpora. It attempts to capture the notion of using unconventional vocabulary, influencing text's humorous quality. The likelihood of a given text sequence, on the other hand, is measured by features like HMM and N-gram based probabilities.

$$Obviousness = \frac{\sum_{t=1}^{t=T} P(token_t)}{T} \tag{2}$$

$$where:$$
$$T = Total\ number\ of\ tokens/words$$
$$P = Probability$$

Compatibility. Humour is usually derived from the usage of vocabulary that can be reinterpreted with a secondary or tertiary meaning during the formation of a "punchline". Compatibility explains the difficulty in understanding the different interpretations of the content. This feature is calculated by taking the average number of meanings of words of a given humorous text. The usage of tokens with a higher number of meanings increases the probability of reinterpretation, thus influencing a text's humorous quality.

$$Compatibility = \frac{\sum_{t=1}^{t=T} \sum Meanings(token_t)}{T} \tag{3}$$

$$where:$$
$$T = Total\ number\ of\ tokens/words$$

Inappropriateness. This feature aims to capture the notion that the usage of inappropriate vocabulary can explicitly influence the humorousness of a sentence. [Sjobergh, 2006] did a study describing the influence of inappropriate vocabulary

on humour and found that it has a significant influence in the quality of humour. We take this idea and calculate the inappropriateness of a token by comparing frequencies of such tokens in a sensual domain to a normal domain. This is unlike Compatibility, which quantifies humour based on the number of different interpretations of a word or group of words.

$$Inappropriateness = \frac{\sum_{t=0}^{T} \frac{Freq_{sensual}(token_t)}{Freq_{normal}(token_t)}}{T} \quad (4)$$

$$where:$$
$$T = Total\ number\ of\ tokens/words$$

Conflict. Conflict measures the difference between the regular and the humorous interpretation of a text. This feature is different from Compatibility which compares the different humorous interpretations. The frequency of the combinations of nouns and adjectives in a humorous/non-humorous corpus is used to calculate Conflict. This calculation is similar to the one mentioned in [Winters *et al.*, 2018]. To capture the dissimilarities between the perceptions of different adjective and noun pairs in humorous and non-humorous contexts, two forms of the features have been considered - *Humorous Conflict* and *Non-Humorous Conflict*.

$$Sum = \sum Bigram_{text}(token_{adj}, token_{noun})$$
$$Conflict_{text} = \frac{Sum}{Pair} \quad (5)$$

$where:$
$$T = Total\ number\ of\ tokens$$
$$Pair = Total\ number\ of\ adjective, noun\ pairs\ in\ a\ sample$$

Adjective Absurdity. This parameter helps estimate the "Comparison" feature described in [Ritchie, 1999] by comparing the usage of different adjectives with a given noun. This feature is calculated as the ratio of the frequency of a given noun-adjective pair to the total number of noun-adjective pairs for a given noun, similar to cosine similarity. This feature is similar to the adjective vector similarity feature mentioned in [Winters *et al.*, 2018] and the noun dissimilarity feature mentioned in [Petrović and Matthews, 2013] but with a slightly modified calculation to cater to our dataset.

$$Value_A = \frac{\sum(N, A)}{\sum_{j=1}^{j=n} \sum(N, A_j)}$$
$$Adjective\_Absurdity = \frac{\sum_{i=1}^{Pair} Value_i}{Pair} \quad (6)$$

$where:$
$$A = Adjective$$
$$N = Noun$$
$$Pair = Total\ number\ of\ adjective, noun\ pairs\ in\ a\ sample$$

Text: "Have you heard that rumour about butter? I probably should not spread it ."

Quality Feature Vector Values	Value
HMM_Probability	-92.91449163
Adjective_Absurdity	0.07742355237
Obviousness	0.003758234
Compatibility	4.666666667
Inappropriateness	1.506216922
Humorous_Conflict	2
Non-humorous_Conflict	1
Noun_Absurdity	0.9123241191
N-gram_Probability	-77.62713114

Table 4: Quality feature vector values for the given sentence.

Noun Absurdity. This feature is similar to the Adjective Absurdity feature and also helps estimate the "Comparison" feature but with a different calculation. Instead of comparing adjective with a given noun, Noun Absurdity compares the usage of all nouns with a given adjective. Bi-grams, POS tags, and ConceptNet distances are used to measure this parameter. The noun vector calculation is weighted by the cosine distance of the ConceptNet embeddings between the particular noun and the list of nouns that appears with the adjective. ConceptNet [Liu and Singh, 2004] is chosen as it is optimised for making practical context-based inferences over real-world texts, and its k-line knowledge increases the connectivity of the semantic network. The usage of ConceptNet in humour processing has further been substantiated in [Labutov and Lipson, 2012].

$$Weight = Cosine\_Distance(Concept\_Embedding(N), Concept\_Embedding(A))$$
$$Value_N = \frac{\sum(N, A) * Weight}{\sum_{j=1}^{j=n} \sum(N_j, A)}$$
$$Noun\_Absurdity = \frac{\sum_{i=1}^{Pair} Value_i}{Pair} \quad (7)$$

$where:$
$$A = Adjective$$
$$N = Noun$$
$$Pair = Total\ number\ of\ adjective, noun\ pairs\ in\ a\ sample$$

HMM Probability. The HMM probability is used to calculate the probability of a particular observation sequence. This is different from the "Obviousness" feature. While Obviousness helps measure the usage of a given word/vocabulary in a text, HMM probability helps measure the likelihood of a given sequence.

$$HMM\_probability = log(P(O|\lambda)) \quad (8)$$

$where:$
$$O = O1, O2, ...On\ (Observed\ Sequence)$$
$$\lambda = HMM\ Model\ Parameters$$

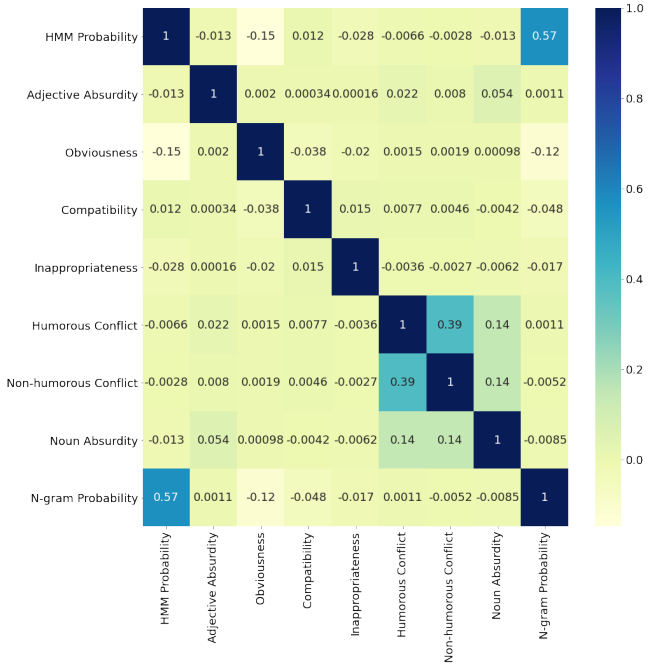


Figure 3: Correlogram between different humour features from Quality Feature Generator (QFG).

N-Gram Probability. The N-Gram probability also calculates the likelihood of observing a sequence in a corpus.

$$N - gram\_probability = log(P(O)) \quad (9)$$

$where$:

$$O = O1, O2, ...On \ (Observed\ Sequence)$$

Unsupervised Quality Estimator (UQE)

The UQE module takes 9-dimensional feature vector from QFG as input. Unsupervised Density-Based Spatial Clustering for Applications with Noise (DBSCAN) [Ester *et al.*, 1996] has been applied on the training data, identifying 5 well-separated clusters. Different clusters do not represent different quality levels of humour. Instead, they represent text groups based on objective humour characteristics. This unsupervised clustering method objectively evaluates humour based on the characteristics of the text, and not on how different users perceive the text on reading it. This unsupervised clustering is further used in designing a personalised rating system as described in Section 5.2.

5.2 Personalised Rating Mapper (PRM)

To tackle subjectivity in humour, PRM attempts to identify each user's preferences. In the first step, each user rates a few samples from the UQE clusters for a domain. Then, the average for each of the clusters is calculated. For any user, the cluster with the highest average value (from the samples initially rated) maps to rating 5 and the cluster with the lowest average value maps to rating 1. This way, the humour content is fine-tuned to every user's preferences.

Algorithm 1: PRM algorithm to find user preference with respect to UQE clusters.

Input: userRating, UQERating arrays for a given domain
Output: Clusters mapped with user's preference
PRM ($userRating$,$UQERating$);
n_1= number of UQE clusters ;
n_2= length of userRating array ;
Let $AvgScore[1 \dots n_1]$ be array with average score with index being the corresponding bucket;
for $i = 1$ *to* n_1 **do**
  $count$ = 0;
  $score$ = 0;
  for $j = 1$ *to* n_2 **do**
    if $UQERating[j]==i$ **then**
      count += 1;
      score += $userRating$[j];
    end
  end
  $AvgScore$[i] = $score$ / $count$;
  `//Average for bucket` i
end
$clusters$ = array with cluster values sorted based on $AvgScore$ array;
return $clusters$;

The PRM algorithm has been described in Algorithm 1. The results of how PRM performs better than the crowdsourced rating system are shown in Section 7.2.

6 Experimental Setup

For domain classification, mentioned in Section 4, we chose a randomized train-test-validation split with a 80:10:10 ratio. The learning rate was set to 0.001 for all the classifiers and the Adam optimization technique [Kingma and Ba, 2014] was used for its computational efficiency. For evaluating the performance of different classifiers, we used a combination of accuracy, precision, recall, and F-1 score. This provided a better insight into the model performance. The hyperparameters for the different models were obtained through a combination of nested cross-validation (CV) and grid search techniques.

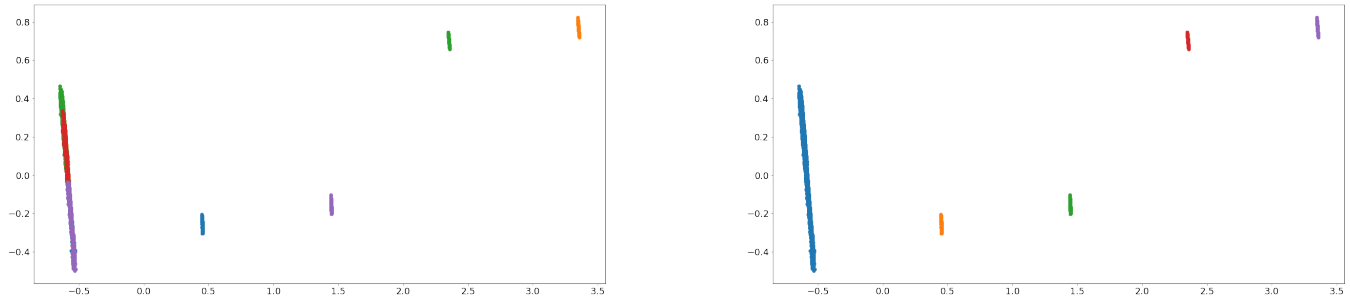
The development of this work has been done on an Ubuntu 16.04 LTS operating system with x86-64 architecture. The primary programming language used is Python 3.6, along with its associated deep learning libraries, Tensorflow and PyTorch.

For graphic intensive tasks, Google Colab and Kaggle have been used to utilise their Nvidia Tesla K80/P100 graphical processing units.

7 Implementation and Inference

7.1 Quality Estimation

Binary Classification. For segregating samples based on whether they are non-humorous or humorous (samples with rating 0 vs. samples with rating 1,2,3,4, and 5), different binary classification models such as a 1-layered FFN, a 2-layered FFN and an SVM have been compared. Here, each



(a) K-means Clustering.

(b) DBSCAN Clustering.

Figure 4: PCA reduced representation of the feature space with K-means and DBSCAN Clustering.

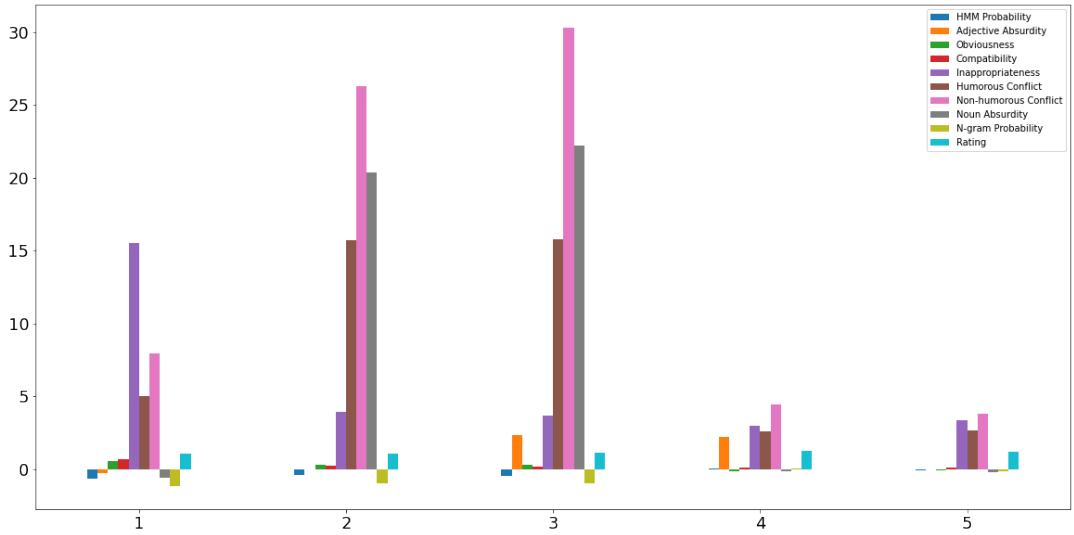


Figure 5: Bar graph representing skewness of feature values for each of the clusters.

	SVM + USE	2-layered FFN + USE	**1-layered FFN + USE**
Train			
Accuracy	0.97	0.98	0.98
Precision	0.97	0.98	0.98
Recall	0.97	0.98	0.98
Support	278489	278489	278489
Test			
Accuracy	0.97	0.98	0.98
Precision	0.97	0.98	0.98
Recall	0.97	0.98	0.98
Support	278489	278489	278489

Table 5: Comparison for Humorous vs. Non-humorous text classification.

text has been represented using 512 dimension pre-trained USE embeddings [Cer *et al.*, 2018]. Overall, the FFNs perform better than SVM. There is a slight difference in performance between the 1-layered FFN and the 2-layered FFN. However, with the 2-layered FFN slightly overfitting, the 1-layered FFN has been considered for the binary classification pipeline. Table 5 compares the results for train and test data. The results in Table 5 show that the best performing model, i.e., the 1-layered FFN, performs better than the results from the previous research [Yang *et al.*, 2015; Chen and Soo, 2018; Winters *et al.*, 2018].

Unsupervised Quality Estimator. To differentiate further between humorous samples, features generated from the QFG have been used. Figure 3 shows the correlation between the normalised features. These normalized features are then used for 2 unsupervised clustering techniques, K-means and DBSCAN. Figure 4a and Figure 4b show the Principal Component Analysis (PCA) plotted results of K-means (k=5) and DBSCAN clustering, respectively. From the figures, it is inferred that the samples are not well segregated using K-means clustering. However, the samples have been well separated into 5 different clusters using DBSCAN clustering. Hence,

Sentence	Cluster
1. My friend owns a zoo but the only animal is a tiny dog.. it's a shitzu.	1
2. Why is it hard to break up with a star trek fan ? Because they are such kling-ons	2
3. What do you get when you drop a piano on a minor ? a flat minor	3
4. Did you get that joke about the Titanic ? It took a while to sink in .	4
5. If I had only one day left to live , I would live it in my math class : it would seem so much longer .	5

Table 6: Example sentences and the corresponding cluster from Unsupervised Quality Estimator (UQE).

DBSCAN performed better that K-means for the dataset used in this paper. An example sentence for each DBSCAN cluster has been listed in Table 6.

Analysis of the clusters. The next step was to analyze the clusters formed by DBSCAN. Figure 5 shows the skewness of the features in the clusters. Skewness of a probabilistic distribution measures the asymmetry that deviates the distribution from a normal distribution. The graph shows that the clusters are segregated based on the dominance (measured in terms of skewness) of a single feature or a group of features, and thus helps distinguish between different humour strains. It is also observed that the domains are uniformly segregated in these clusters. This further proves that the new segregation proposed for estimating the quality is domain invariant and can be extended to any other domain, implying that the domain of the text does not influence the cluster distribution.

7.2 Personalised Rating Mapper

In order to test if the UQE clusters can be used to determine user preference, a follow-up survey to the one mentioned in Section 1 was sent to the same set of users to rate samples from each cluster. The results from this analysis showed that the correlation between ratings given by users and the 5 different clusters is close to 0. This result explained that the clustering mechanism based on the linguistic features is unbiased, thus reducing the noise for computational model to classify and quantify humour. This also showed that different users have different levels of liking for humour based on how the content is written linguistically.

To cater to individual users, PRM was designed in such a way that the issue of subjectivity in humour is minimized significantly. After considering a user's preference, PRM personalises content by designing a different rating mechanism for that user, as explained in Algorithm 1. In order to test this algorithm, another survey was sent with the new rating mechanism and we asked the users to rate the content. We used F1-score and RMSE values to compare whether the PRM rating system performs better than the average rating system from different websites. From the survey, 97 out of 133 user responses found PRM to be more aligned to their preferences which is close to 72.9%. From the initial survey mentioned in Section 1, only 27 of the 133 users felt that the crowdsourced rating was aligned to their preference which is close to 20.3%. This result clearly showed that PRM based system is performing close to 50% better than the crowdsourced rating mechanism.

8 Conclusions and Future Work

This paper proposes HumourSpace, a novel framework that performs unsupervised clustering of samples based on computational linguistic features for evaluating humour content. The paper then moves on to personalised ratings of humour content using PRM. Experimental analysis shows that the results from PRM were performing better than the average crowdsourced ratings by 50% (73% and 23% respectively), indicating that the improvements via personalisation are significant. The analysis of domains within each cluster showed uniformity, proving that the clusters are domain invariant and can be further extended to any new domain. As a corollary, it shows that domain does not play a significant role in objectively evaluating humour.

The role of the clusters as an evaluation metric along with existing metrics such as BLEU scores, for the task of generating new humorous content, is being studied. Since these computational linguistic features are humour specific, it can be useful for existing generators to improve humorous content. The nature of the computational linguistic features is such that it can also be applied to languages other than English, with the additional requirement of understanding the syntax and semantics of these languages. The PRM proposed is an abstract approach to identify the humour strain most suitable for an individual. The current implementation is based on the average-score model, where a humour strain is assigned to a user based on the cluster that has the highest average rating. This implementation can be further enhanced by studying the current sentiment of a user based on recent conversations, along with past behavioural trends, to get a better understanding of an individual's affinity to a particular humour strain. This could result in a more well-rounded experience with a conversational agent, thus leading to better human-computer interaction.

References

[Binsted, 1996] Kim Binsted. Machine humour: An implemented model of puns. 1996.

[Cai and Ehrhardt, 2013] Jim Cai and Nicolas Ehrhardt. Is this a joke? 2013.

[Cer *et al.*, 2018] Daniel Matthew Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder. *ArXiv*, abs/1803.11175, 2018.

[Chen and Soo, 2018] Peng-Yu Chen and Von-Wun Soo. 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, 2018.

[Dybala, 2008] Pawel Dybala. Extracting dajare candidates from the web-japanese puns generating system as a part of humor processing research. *The Proceedings of the First International Workshop on Laughter in Interaction and Body Movement (LIBM'08), Asahikawa, Japan, June*, pages 46–51, 2008.

[Ester *et al.*, 1996] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[He *et al.*, 2019] He He, Nanyun Peng, and Percy Liang. Pun generation with surprise. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1734–1744, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[Kiddon and Brun, 2011] Chloe Kiddon and Yuriy Brun. That's what she said: double entendre identification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 89–94. Association for Computational Linguistics, 2011.

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

[Labutov and Lipson, 2012] Igor Labutov and Hod Lipson. Humor as circuits in semantic networks. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 150–155, 2012.

[Liu and Singh, 2004] Hugo Liu and Push Singh. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.

[McKay, 2002] Justin McKay. Generation of idiom-based witticisms to aid second language learning. *Stock et al*, pages 77–87, 2002.

[Misra, 2018] Rishabh Misra. News category dataset. 06 2018.

[Petrović and Matthews, 2013] Saša Petrović and David Matthews. Unsupervised joke generation from big data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 228–232, 2013.

[Pungas, 2017] Taivo Pungas. A dataset of english plaintext jokes. *GitHub repository*, 2017.

[Raymond, 1991] Eric S Raymond. Jargon file version 2.8.1, 1991.

[Ritchie, 1999] Graeme Ritchie. Developing the incongruity-resolution theory. Technical report, 1999.

[Ritchie, 2005] Graeme Ritchie. Computational mechanisms for pun generation. In *Proceedings of the Tenth European Workshop on Natural Language Generation (ENLG-05)*, 2005.

[Shahaf *et al.*, 2015] Dafna Shahaf, Eric Horvitz, and Robert Mankoff. *Inside Jokes: Identifying Humorous Cartoon Captions*, page 1065–1074. Association for Computing Machinery, New York, NY, USA, 2015.

[Sjobergh, 2006] Jonas Sjobergh. Vulgarities are fucking funny, or at least make things a little bit funnier. *Proceedings of KTH CSC, Stockholm. 2006*, 2006.

[Srivastava *et al.*, 2015] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[Stock and Strapparava, 2005] Oliviero Stock and Carlo Strapparava. Hahacronym: A computational humor system. In *Proceedings of the ACL 2005 on Interactive poster and demonstration sessions*, pages 113–116. Association for Computational Linguistics, 2005.

[Suslov, 1992] IM Suslov. Computer model of a "sense of humour" i. general algorithm. *Biophysics*, 37(2):242–248, 1992.

[Turing, 1950] A. M. Turing. Computing Machinery and Intelligence. *Mind*, LIX(236):433–460, 10 1950.

[Turing, 2009] Alan M Turing. Computing machinery and intelligence. In *Parsing the Turing Test*, pages 23–65. Springer, 2009.

[Venour, 2000] CTJ Venour. The computational generation of a class of pun. 2000.

[Winters *et al.*, 2018] Thomas Winters, Vincent Nys, and Daniel De Schreye. Automatic joke generation: Learning humor from examples. In *International Conference on Distributed, Ambient, and Pervasive Interactions*, pages 360–377. Springer, 2018.

[Winters *et al.*, 2019] Thomas Winters, Vincent Nys, and Danny De Schreye. Towards a general framework for humor generation from rated examples. *http://computationalcreativity.net/iccc2019/assets/iccc_proceedings_2019.pdf*, (Proceedings of the 10th International Conference on Computational Creativit):274–281, 2019.

[Yang *et al.*, 2015] Diyi Yang, Alon Lavie, Chris Dyer, and Eduard Hovy. Humor recognition and humor anchor extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2367–2376, 2015.

[Yang *et al.*, 2016] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489, 2016.

[Yang *et al.*, 2019] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763, 2019.