ISSN 1870-4069

Development and Classification of a Chinese Humor Corpus

Yi-Ciao Gu[1], Yuen-Hsien Tseng[1], Wei-Lun Hsu[1],
Wun-Syuan Wu[1], Hsueh-Chih Chen[1,2]

[1] National Taiwan Normal University,
Taiwan

[2] National Cheng Kung University,
Taiwan

{minnie70011, chloe8599, greenteax23}@gmail.com
{samtseng, chcjyh}@ntnu.edu.tw

Abstract. In this work, we present a dataset for computational humor, which has the potential to be useful for Chinese e-commerce to enhance customer-machine/customer-clerk dialogue experience. The current humor corpus consists of 3,691 local jokes from more than 40 sources in Taiwan. Information retrieval technique is applied to remove near-duplicate jokes. The corpus is classified manually into 9 categories for potential use in proper context. Preliminary automated classification to discriminate the joke category using four traditional machine learning methods and three deep neural networks were conducted. The current results show that the performance of machine classifiers is far behind that of humans, leaving much room for research to apply them in proper context to achieve the goal of enhancing positive customer experience. Nevertheless, this developing humor corpus in traditional Chinese has indispensable value, because humor has at least subjective, cultural, regional, temporal, and linguistic characteristics, such that any local corpus has its value in the corresponding applications. Implication and potential application of this corpus are also discussed.

Keywords: Corpus annotation, joke classification, computational humor.

1 Introduction

Since Apple launched the Siri personal voice assistant in 2011, conversational user interface (CUI) has been gradually accepted by customers as the third human-machine interaction channel, in addition to the Web browsers and mobile Apps. Many companies have started to develop their own dialogue systems or chatbot platforms, allowing more businesses to provide a variety of active or passive customer services,

pp. 117–126; rec. 2019-01-17; acc. 2019-03-13

such as: booking, notification, promotion, etc. However, most CUI services are lack of humanity, which may lead to unsatisfactory experience during the interaction.

To improve customer's CUI experience, humorous dialogue can be applied as it has the potential to reduce customer complaints [1], to provide alternative effective responses outside the predefined services [2], and to win the trust from the customers. In addition, advertising, entertainment, and other industries are also the sectors to which humor is often applied for better products and services [3].

However, humor has at least five characteristics of being subjective, cultural, regional, temporal, and linguistic. To integrate humor into the conversational services, a local humor corpus is required to support the corresponding market. This work is aimed to develop a Chinese humor corpus for the mentioned purposes.

2 Related Work

In the field of natural language processing, human-computer interaction, and artificial intelligence, studies on humor identification and humor generation have been conducted since 1995, among which humor identification was considered to be more difficult than humor generation [4]. The goal of these studies is to explore humorous computational models, to enhance human-computer communication and user experience, or to assist people with communication disabilities to enhance their interpersonal interactions [5]. In recent years, advances in related technologies such as information retrieval, semantic processing, and machine learning, have further promoted the humorous dialogue research.

For example, the International Workshop on Semantic Evaluation (SemEval) held the Learning a Sense of Humor evaluation task in 2017 [6]. However, most studies focus on the humor identification research of English texts. As for the humor generation research, the use of English pun, acronym, and joke collection are the main approaches to generate humor. For humor classification study, Mihalcea and Strapparava [7] have collected 16,000 humorous and non-humorous one-liners, respectively.

They have tried stylistic features, such as alliteration, antonymy, and adult slang, to be used by decision trees, and found that alliteration is a better feature. They also use content words as features for Naive Bayes (NB) and Support Vector Machine (SVM), where SVM achieved 0.7751 accuracy in humor classification. Zhang and Liu [8] have manually collected 1,000 positive and negative humorous tweets. They use 50 features, grouped into 5 categories, for humor classification by Gradient Boosted Regression Trees, where special sign in tweets, ratio of content words, and polarity of terms are better predictors. Chen and Lee [9] use convolutional neural network (CNN) in comparison with the above traditional machine learning methods.

Their results show that CNN not only outperforms traditional methods, but also reduces the burden to design the features. In the SemEval 2017 task, the lessons learned are that humorous degree ranking is still difficult and that deep neural networks (DNN) normally perform better for the humor comparison task [6]. In humor generation study, humor and related corpora are indispensable resources. For example, Ozbal and Strapparava (2012) [10] combines English WordNet and ConceptNet resources with

Table 1. The metadata of the joke collection.

Field Name	Description
PKey	The primary key for identifying a joke
SourceTitle	Joke title from the source if any
GivenTitle	The first 10 characters of the joke content if SourceTitle is empty
Sharer	The one who share the joke, if known
Creator	The one who create the joke, if known
TextContent	The joke texts where line breaks are retained
PublicationDate	The publication date of the joke, if available
CollectingDate	The date when the joke is collected
SourceTopic	The topic/subject of the joke given from the source
GivenTopic	The topic/subject annotated by this article's authors
Length	The length of the joke content for easy of retrieving proper jokes
Language	Mostly in Chinese, a few are mixed with English and Taiwanese
SourceType	Name of the source (website, book, App)
Identifier	website URL, book ISBN, joke App name, etc.
Popularity	Humorous rating, number of likes, etc, if available

homophonic puns and metaphors to create creative names, especially humorous neologism.

Stock and Strapparava (2003) [11] used the incongruity theory to produce an interesting interpretation of the existing English acronym, e.g., reinterpreting technical words in religious terms, or providing users with concepts to generate interesting new words by the system. One of the examples is to convert IJCAI (International Joint Conference on Artificial Intelligence) into: Irrational Joint Conference on Antenuptial Intemperance.

3 Chinese Humor Corpus Development

There are a series of steps in developing this traditional Chinese humor corpus with sustainable and extendable value. These steps include: 1) select sources for collecting humorous jokes, 2) analyze the joke contents and define the fields (metadata) necessary for the corpus, 3) collect the jokes by various means, 4) remove near-duplicate jokes, 5) classify the jokes into categories for further use.

3.1 Joke Collecting and Cataloguing

To diversify the joke contents, we search and evaluate at lot of joke sources and, during a period of eight weeks, collect 3,828 jokes from 41 sources, which include 27 public websites (2777 jokes), 11 joke collection books (895 jokes), and 3 free Apps (156 jokes). After analyzing a sample of them, we decide to catalogue the joke collection based on Dublin Core Metadata Element Set [12]. The metadata finally contains 14 fields as listed in Table 1.

Some of the important fields include source identifier, source publication date, joke collecting date, sharer, author, and joke content. Note that it is hard to know the real author/creator or the copyright of a joke, as jokes may be told or revised in various ways before they are disseminated in social media or websites, and/or collected in books or Apps. These important fields are an attempt to provide as sufficient information as possible to trace back to the origin of the joke.

3.2 Duplicate Removal

As the jokes are accumulating, it is possible to collect duplicates from different sources. We then developed a full text matching tool, based on bag of words and TFxIDF term weighting, to detect near duplicates for removal [13]. This step reduced the number of jokes from 3,828 to 3,691, with each removal verified by the authors.

3.3 Manual Classification

A joke is humorous only when it is applied in a proper context. To help decide the right context, the corpus is manually classified into nine categories by at least two persons (all majored in Library and Information Science). Some of the 41 sources have their joke classification, especially for the website jokes. We adopt the most used categories and redefine their scope to yield the nine categories (the details are in the next section).

Two annotators classified each joke independently based on the category definition. When there is inconsistency (only 62 jokes with inconsistent categories), a third annotator helped label these jokes. Most of the categories among the three is then assigned to the joke. The inter-rater agreement for the two major annotators is as high as 0.97 in Cohen's kappa coefficient.

3.4 Corpus Format

The joke corpus is manually collected in an Excel file, with all field names in Chinese. We have written a Python code to convert it into an XML file with field names mapped to English as shown in Table 1. For manual update/development of the corpus, the Excel file is used as it is more efficient for manual editing. For computer processing, the XML file is recommended.

4 Discriminating Humor Categories by Machines

The high inter-rater agreement signifies the ease of discriminating the humor category by humans. The ability to automatically identify humors and their categories is of great value to later applications. As a preliminary exploration, we applied text classification techniques to evaluate the performance of machine humor discrimination. The

Table 2. Basic statistics of the joke corpus.

Joke	**Training Set**				**Testing Set**			
Category	Num	Avg	Max	Min	Num	Avg	Max	Min
C1: Lame	595	103	828	3	255	95	578	4
C2: Vocational	419	142	648	29	180	141	599	24
C3: Love-Affair	339	98	841	15	146	118	652	18
C4: Family	260	123	507	31	111	120	764	33
C5: Campus	245	123	759	14	105	116	293	18
C6: Adult	176	173	855	4	76	145	465	16
C7: Terminology	165	132	1140	14	70	130	1289	17
C8: Celebrity	155	107	518	12	67	129	713	18
C9: Others	229	107	1720	7	98	109	610	13

automated text classification follows these pipeline steps: 1) split of training/testing sets, 2) feature extraction, 3) model training, and 4) performance evaluation.

4.1 Training and Testing Sets

For training and testing machine classifiers, the corpus is split in a way that for each category 70% of the jokes are for training and the remaining 30% are for testing. Table 2 shows the nine categories with their number of jokes, and the average characters, maximum number of characters, and minimum number of characters for the jokes in a category.

4.2 Feature Extraction

To extract features for traditional machine classification, we currently exploit only the content words with four kinds of features. The text in each joke is first cleaned and standardized by tokenization, segmentation, and punctuation/stop word removal. Each text is then transformed into a feature vector with each element representing a term in the corpus. The term's value can be the term's occurring frequency (term frequency, TF) in a text, or the normalized TF multiplied by the logarithm of the inverse document frequency (IDF) of the term in the whole dataset (TFxIDF).

The term here can be a normal word, n consecutive words, or n consecutive characters. With these different values for a term and different ways to denote a term, there are four feature vectors (or feature sets) used: 1) Word Count: the term represents a normal word, and its value is the word's TF. 2) TFxIDF: the term is a normal word, and its value is the word's TFxIDF. 3) Word N-grams: the term is a N consecutive word in a text and its value is the term's TFxIDF. 4) Char N-gram: the term is a N consecutive character in a text and its value is the term's TFxIDF.

In contrast to the above sparse vectors where semantically similar terms may have no intersection in their vector representation, word embedding is a special way to transform a word into a dense vector, where semantically similar terms are also highly similar in their embedding vectors [14]. Word embeddings can be trained using the input corpus itself or can be obtained from pre-trained word embeddings such as those pro-vided by fastText [15].

In our experiments, we used those provided by fastText, because fastText considers sub-word units such that out-of-vocabulary terms still have their embedding vectors by combining the corresponding ones of their sub-word units.

4.3 Design and Validation of Machine Classifiers

Four traditional ML methods are used, which are Naive Bayes (NB), Support Vector Machine (SVM), Random Forest (RF), and single hidden-layer neural network (NN) with a softmax layer as its output layer. For the deep learning (DL) methods, the first one is based on CNN, where pretrained word embedding with trainable weights for the application task is the first hidden layer with a dropout rate of 0.25 (same for all the dropout mentioned later). This is followed by a 1-d convolution layer to convolve the embedding vectors to extract local contextual information of the input word embeddings.

A global max pooling layer follows to summarize the local contextual information. A dense hidden layer with dropout is used to summarize the convolved/contextual messages into dense information. The final layer is then used to map the summarized information to a category prediction layer, which uses softmax as its activation function and is thus called a softmax output layer.

For the second DL method (RCNN), the first hidden layer has the same embedding structure as the CNN. The next layer is a bi-directional GRU (Gated Recurrent Unit) to extract longer dependent information in the text, which is followed by a 1-d convolution layer, a max pooling layer, and then a dense hidden layer with dropout.

The final output layer is a softmax layer. The third one is fastText [15]. Although it may not fully belong to a DL method, it combines some of the most successful concepts in natural language processing, such as word embedding and machine learning. It uses a hierarchical classifier instead of a flat structure, in which the different categories are organized in a tree. FastText also exploits the fact that classes are imbalanced by using the Huffman algorithm to build the tree used to represent categories.

The depth in the tree of very frequent categories is therefore smaller than for infrequent ones, leading to further computational efficiency [15]. All the above traditional classifiers adopt the default values of the Scikit-Learn package in Python. The training epoch for DL classifiers implement by Keras package with TensorFlow backend is set to 20 (the classifier sees each of the training texts for 20 times). With these settings, the DL classifiers achieve over 95% accuracy for classifying the training set in 10 epochs.

We think this training epoch is large enough to prevent the classifier from overfitting while restraining its training time. For fastText, we follow the tutorial instructions and try on various settings until we could not obtain better result. All the above classifiers

Table 3. Performance of various models with different features.

Models	Features	MicroF1	MacroF1	Time (seconds)
NB	Word Count	0.5027	0.4068	0.77
SVM	Char bi-gram	**0.5225**	**0.4503**	0.84
RF	Char bi-gram	0.4548	0.3601	1.09
NN	Char bi-gram	0.4990	0.4429	33.96
CNN	Word Embedding	0.4954	0.4102	922.02
RCNN	Word Embedding	0.4720	0.4144	4787.52
fastText	Word bi-gram	0.5036	0.4195	5.05

have been tested and verified on a balanced binary sentiment classification corpus[1], with the DNN approaches having slightly better performance (0.9967 for CNN vs 0.9948 for SVM in MicroF1 with 30%=2,126 test tweets).

4.4 Performance Evaluation and Results

Two metrics are used for performance comparison: MicroF1 and MacroF1. For MicroF1, a single confusion matrix for all the testing documents is calculated based on the ground-truth labels and the predicted labels. From this cross-tab matrix, precision and recall rates are calculated. Their harmonic average with equal weights to evenly emphasize both precision and recall is the so-called MicroF1 measure. For MacroF1, a confusion matrix is calculated for each category.

The F1 measures from each category are then averaged into MacroF1 with equal weights to emphasize each category evenly. Based on the above calculation, MicroF1 measures overall document classification effectiveness and thus reveals more the effectiveness of a few major categories. In contrast, MacroF1 takes each category's effectiveness into consideration and thus reveals more the effectiveness of most minor categories. In Table 3, we show only the best feature set for each classifier.

The results show that SVM is most effective with efficiency (all are run under a MacBook computer) far better than the DNN methods. The DNN methods do not show advantage like those discussed in the related work, even though they utilize pre-trained embedding vectors that exploit additional language knowledge. The large difference between 0.5153 MicroF1 and 0.97 inter-rater agreement implies that there is a large room for improvement, either in the feature extraction or in the learning model.

4.5 Error Analysis and Implication

Previous studies (as discussed in the Related Work) on humor classification use humorous jokes as positive examples and non-humorous similar texts as negative examples. The classification task is thus to discriminate whether a given text is humorous or not. While in our experiment, all testing texts are humorous jokes. Our goal in this article is to explore whether a machine classifier could tell which topic the

[1] https://www.kaggle.com/c/si650winter11/data

Table 4. Confusion matrix for the SVM classifier.

	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**	
C1	**159**	25	18	8	6	5	10	6	18	255
C2	33	**121**	7	5	0	3	1	3	7	180
C3	32	7	**89**	2	2	8	2	0	4	146
C4	16	3	5	**78**	1	3	0	2	3	111
C5	8	5	2	1	**81**	2	0	2	4	105
C6	24	11	11	3	6	**11**	1	3	6	76
C7	27	6	10	3	8	0	**10**	1	5	70
C8	23	9	5	4	4	0	4	**14**	4	67
C9	31	15	18	3	4	2	4	5	**16**	98
	322	187	147	104	108	32	28	31	51	1010

Table 5. Breakdown performance for the SVM classifier.

	Precision	**Recall**	**F1-score**	**Support**
C1: Lame	0.45	0.62	0.52	255
C2: Vocational	0.60	0.67	0.63	180
C3: Love-Affair	0.54	0.61	0.57	146
C4: Family	0.73	0.70	0.72	111
C5: Campus	0.72	0.77	0.75	105
C6: Adult	0.32	0.14	0.2	76
C7: Terminology	0.31	0.14	0.2	70
C8: Celebrity	0.39	0.21	0.27	67
C9: Others	0.24	0.16	0.19	98

input joke (e.g., uttered by a user) is about, such that the subsequent response by the machine remains in the right context.

Table 4 lists the confusion matrix for the best classifier, SVM. It can be seen that most texts in small categories have been incorrectly classified into larger categories. For example, 23 texts in the C8: Celebrity are classified into C1: Lame jokes. This tendency has already been indicated by the lower MacroF value (lower than MicroF). However, there are two medium size categories that exhibit relatively high performance: C4: Family and C5: Campus jokes, as shown in Table 5, implying that the term variations are smaller between the training set and the testing set. This indicates the feasibility of applying machine classifiers to these topics. Recently, there are better deep learning models for NLP, such as BERT from Google [16] and GT2 from OpenAI [17]. They may be of great help to improve the classification performance. However, the difficulty to be able to interpret the reason (compared to the salient features revealed by SVM) may hinder the understanding of the mechanism of computational humor.

5 Potential Applications of the Corpus

In a conversational user interface (CUI), understanding the user's context, intent, or topic is valuable to create an empathetic CUI system. Despite higher prediction performance is observed in other corpora, the above experiments show that there is much to explore to achieve this goal, where topic-dependent solutions may be useful to improve the overall performance. Our corpus could contribute to this line of study. In contrast to the above humor discrimination or identification task, another line of computational humor is about humor generation.

An example of applying this corpus for this purpose is providing a CUI for users to retrieve relevant jokes. We have built a chatbot in the LINE platform (a popular mobile messaging App in Eastern Asia), named as IceBreaker, to allow a nervous-prone user to retrieve a joke for telling in front of a group of people when he/she needs to. The user may say to the IceBreaker (using the built-in voice recognition in a cell phone), like: "give me a joke about joke" (or "tell me a joke about *some_topic/some_keyword*", or simply input a keyword).

The IceBreaker would respond with some texts like: "Why can't you tell a joke at the beach? Because there would be a Tsunami!" Here Tsunami in Chinese sounds like "the Sea is laughing". The joke reminds the user to apply some homophone skills to tell a joke that may lead to some devastating consequence (which complies with the incongruity theory in humor study). The chatbot also records the feedback from the user to accumulate the average degree of the humor of the joke. The user can also feedback a new joke through the chatbot. As more information like these is accumulated, the corpus, as a seed, would be expanded and refined by its users.

6 Conclusions and Future Work

We have presented a manually collected and annotated local humor corpus, which is the first traditional Chinese one to our best knowledge. The corpus together with the tools for near-duplicate detection, category classification, and joke retrieval via LINE chatbot will be made public once we figure out the copyright issue.

We wish that it would be useful and extendable for future research and also applicable in practical application environment. For linguists and those social researchers who study humor as an important issue, we expect this manually catalogued corpus to be a valuable and sustainable dataset for further exploration.

Acknowledgments. This work is supported in part by the Ministry of Science and Technology under the grant MOST 107-2221-E-003-014-MY2 and MOST 108-2634-F-002-022, and by the "Institute for Research Excellence in Learning Sciences" of

National Taiwan Normal University from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education in Taiwan.

References

1. Binsted, K.: Computational humor. IEEE Intelligent Systems, vo. 21, no. 2, pp. 59 (2006)
2. Bellegarda, J. R.: Spoken language understanding for natural interaction: The siri experience. natural interaction with robots, knowbots and smartphones. New York, USA, Springer (2014)
3. Mihalcea, R., Strapparava, C.: Technologies that make you smile: adding humor to text-based applications. IEEE Intelligent Systems, vol. 2, no. 5, pp. 33–39 (2006).
4. Zhang, R., Liu, N.: Recognizing Humor on Twitter. In: The 23rd ACM International Conference on Information and Knowledge Management. Shanghai, China, ACM, pp. 889–898 (2014)
5. Ritchie, G., et al.: The STANDUP Interactive Riddle-Builder. IEEE Intelligent Systems, vol. 21, no. 2, pp. 67–69 (2006).
6. Potash, P., Romanov, A., Rumshisky, A.: SemEval-2017 Task 6: #HashtagWars: learning a sense of humor. In: 11th International Workshop on Semantic Evaluations, Vancouver, Canada (2017)
7. Mihalcea, R., Strapparava, C.: Learning to laugh (automatically): Computational models for humor recognition. Computational Intelligence, vol. 22, no. 2, pp. 126–142 (2006)
8. Zhang, R., Liu, N.: Recognizing humor on Twitter. In: 23rd ACM International Conference on Information and Knowledge Management, Shanghai, China (2014)
9. Chen, L., Lee, C. M.: Predicting audience's laughter using convolutional neural network. arXiv:1702.02584 (2017).
10. Ozbal, G., Strapparava, C.: Computational humour for creative naming. In: 3rd International Workshop on Computational Humor, Amsterdam, Netherlands (2012)
11. Stock, O., Strapparava, C.: Getting serious about the development of computational humor. In: 18th International Joint Conference on Artificial Intelligence, Acapulco, Mexico (2003)
12. Dublin Core Metadata Element Set, Version 1.1: Reference Description (2012) Retrieved from http://dublincore.org/documents/dces/.
13. Tseng, Y. H., Teahan, W. J.: Verifying a Chinese Collection for Text Categorization. In: 27th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '04, Sheffield, U.K (2004)
14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems (2013)
15. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. CoRR, abs/1607.01759 (2016)
16. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (2018)
17. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language Models are Unsupervised Multitask Learners (2019)