Developing Conversational Data and Detection of Conversational Humor in Telugu

Vaishnavi Pamulapati and **Radhika Mamidi**
Language Technologies Research Center
International Institute of Information Technology, Hyderabad
vaishnavi.p@research.iiit.ac.in, radhika.mamidi@iiit.ac.in

Abstract

In the field of humor research, there has been a recent surge of interest in the sub-domain of Conversational Humor (CH). This study has two main objectives. (a) develop a conversational (humorous and non-humorous) dataset in Telugu. (b) detect CH in the compiled dataset. In this paper, the challenges faced while collecting the data and experiments carried out are elucidated. Transfer learning and non-transfer learning techniques are implemented by utilizing pre-trained models such as FastText word embeddings, BERT language models and Text GCN, which learns the word and document embeddings simultaneously of the corpus given. State-of-the-art results are observed with a 99.3% accuracy and a 98.5% f1 score achieved by BERT.

1 Introduction

Humor as a phenomenon has interested scholars from diverse fields since time immemorial (Morreall, 2012). The abundance of research studies dedicated to humor is not only due to the fascinating nature of the domain but also due to its impact in everyday life (Martin and Lefcourt, 1983)(McGee and Shevlin, 2009).

In a conversational discourse between interlocutors, their attitude, present state of mind, psychological distance between the topic of humor and the individual (McGraw and Warren, 2010) determine whether what is intended to be humorous is perceived so. Conversational Humor (CH) is a subset of verbal humor. Verbal humor exists in the verbality of what is being spoken. Canned jokes (such as light-bulb jokes, knock-knock jokes) are a part of verbal humor, where they are not context-dependent, i.e., they can be removed from a conversation and still perceived as humorous. On the other hand, CH is heavily dependent on various factors including speakers' personalities, the relevant culture that is referenced, and current events. Numerous studies that focus on the detection of humor in short jokes/tweets rely on the contrastive discourse relation present in humorous instances (Liu et al., 2018). However, in conversations, participants' personalities, their sense of humor, and the relationship between the participants, add unique complexities to the task of detection of CH. The following example (translated) from a Telugu stage play, *Kanyasulkam*.

> Puta: *Where is he?*
> Madhu: *You don't listen to me, I have already told you that the person you're looking for is not here!*

Context:

Giri and Rama are hiding under Madhu's bed. Puta is looking for Giri and the latter is hiding because he is scared of Puta. The audience, Giri and Rama know that Madhu is lying about Giri's whereabouts. This irony is recognized by the audience and causes Dramatic Irony (Dempster, 1932)(Pamulapati et al., 2020). This depicts how the participants contribute to CH.

There have been attempts at formulating a typology for CH. Dynel (2009) defined CH and enlisted the different types of verbal discourses that cause CH. Similarly, Pamulapati et. al (2020) developed a hierarchical framework that considers whether it is a monologue/dialogue, the benignity of the utterance, the type, and techniques used to cause CH. Despite the recent interest in CH, there is a need for more computational work in this field as researchers are striving to make virtual assistants like Siri and Alexa produce more human-like discourses. This paper aims to further the research in this direction by working on the task of detection of CH. For low resource languages such as Telugu, finding conversations for the purposes of this study proved to be a challenge. To the best of our knowledge, this work is the first attempt to compile conversational data in Telugu. To compensate for the paucity of data, pre-trained neural

Proceedings of the 2nd Workshop on Computational Approaches to Discourse, pages 12–19
November 10–11, 2021.

network models such as FastText and BERT and the state-of-the-art Text GCN model are used for the task of detection of CH.

2 Related Work

Chaudhary, et. al (Chaudhary et al., 2021) used features at several levels such as morpho-syntactic, lexico-semantic and pragmatic level and architectures including Logistic Regression, Gaussian Näive Bayes, and SVM to detect and generate CH. They utilized an existing dataset on Kaggle that comprises 200k+ Reddit jokes in English.

Weller and Seppi (Weller and Seppi, 2019) employ a Transformer architecture for humor detection and explain the advantages over traditional classifiers such as CNN and compare their results with human annotated data. Existing datasets such as Short Jokes dataset on Kaggle and Pun of the Day were used. Apart from these they had mined Reddit jokes and used the number of upvotes to demarcate the humorous jokes from the non-humorous jokes.

Annamoradnejad and Zoghi (2020) use BERT to generate embeddings for their combined dataset of several existing humorous datasets, including the dataset provided by Khandelwal et. al (2018). Different classifiers such as SVM and XLNet are compared with their proposed model, the latter giving an F1 score of 98.2 percent.

Yao et. al (2019) proposed a Graphical Convolutional Networks architecture (GCN) for text classification. The framework builds a text graph based on word co-occurrence using the dataset's text. These aforementioned studies provided motivation to incorporate BERT, FastText and TextGCN architectures in the experiments for detecting Conversational Humor.

3 Data and Preparation

To compile humorous data for the task of humor recognition, jokes from various sources were scraped including jokes section of news websites like Times of India (Samayam)[1], and other miscellaneous sources such as Blogspot websites[2]. As the area of focus is Conversational Humorous data, the dataset was filtered to assemble jokes that followed a conversational format.

Though extensive attempts to obtain conversational data from movies or TV shows were made, due to reasons such as unavailability of transcripts, manual transcription needed, and unavailability of multilingual OCRs[3], this direction proved to be unfeasible. Despite jokes being used, in the final dataset several conversational features are intact. All instances of the final humorous data used are of the same format as

Speaker 1: *utterance 1*
Speaker 2: *utterance 2*
...
Speaker n: *utterance n*.

Therefore, features such as turn taking and sequence organization are present. Turn-taking organization is where participants alternate their utterances, minimizing the noise arising from clashing of utterances to have a smooth or effective communication (Sacks et al., 1978). Sequence organization is the organization of these turns. If the conversational goal is to seek information, the first turn is the question, and the second turn is the answer (Schegloff et al., 1977).

3.1 Humorous Data

After filtering, a total of 2,047 conversational jokes were compiled. These jokes did not follow a standard structural format. Therefore, manual intervention was applied to make a homogeneous conversational humorous dataset (translated example given below) as the model could distinguish humorous and non-humorous data based on structural features, rather than semantic.

Original Conversational Joke:

"*Hey! Your dog is exactly like a tiger!*", said Suresh. "*That is (emphasis) a tiger. It has been going around talking and thinking about love and has turned into a dog!*", replied Mahesh.

Manually Converted to:

Suresh: *Hey! Your dog is exactly like a tiger!*
Mahesh: *That is (emphasis) a tiger. It has been going around talking and thinking about love and has turned into a dog!*

3.2 Non-Humorous Data

To assemble non-humorous conversations, tweets and their replies were scraped to form conversations. A reply to a tweet engages with the original tweet's user and their audience. Any subsequent

[1] https://telugu.samayam.com/telugu-jokes/funny-jokes/articlelist/49228696.cms

[2] https://bit.ly/3yL2HuX

[3] As the Telugu script was abundantly mixed with Roman script in the movie dialogues

replies are either directed at the original tweet's user or at the other replies' users.

Every tweet is assigned several attributes such as tweet ID, user ID, timestamp, etc. Using a tweet's ID, conversation ID and timestamp, tweets and their replies were compiled to form a conversation. A combination of Twint[4], an advanced Twitter scraping tool, and Twitter's API v2 endpoint[5] were used to assemble the data.

To avoid the non-canonical forms of words obtained from scraping Telugu tweets written in roman script, 'lang: te' (aiming to fetch tweets written in Telugu script) was used as a filter when scraping tweets. In addition, in order to build a multidomain non-humorous dataset (Mihalcea and Strapparava, 2005), hashtags such as '#cinema', '#politics', '#cooking', etc. were used as filters when collecting tweets.

Usernames were not used as speakers in the conversation for two reasons. Firstly, for the sake of anonymity, and secondly, usernames and names contain numbers and special characters. Consequently, to generate a natural conversation, the usernames were replaced by common Telugu names. Using the tweet's author ID, speaker identity was preserved. For instance,

User123: *Hello, how is today's weather?*
User456: *It seems to be very sunny.*
User123: *Oh, wonderful!*

In both instances, 'User123' is replaced by the same common Telugu name. This resulted in a corpus of 10,156 conversations. Tweets that contained hashtags such as '#funny', '#joke', '#hahaha', or smiling/laughing emojis were removed to improve the dataset. After checking the corpus, conversations that contained profanity were removed to avoid ambiguity whether the conversation was humorous (Fägersten, 2012), finally resulting in 6,202 non-humorous conversations.

3.3 Attempts at Homogenizing Entire Dataset

To make the non-humorous data and humorous data as similar as possible in structure, several changes were made to the collected jokes and non-humorous conversations (statistics in Table 1)

- Preprocessing steps such as removal of URLS, hashtags, and emojis were performed. In the case of a tweet, if emojis comprised the entirety of an utterance, the predecessor utterance was demarcated the end of the conversation, as removing that utterance is essentially changing the conversation.

- The average number of utterances of all the conversational jokes was calculated to be approximately 3. Hence, the non-humorous conversations were trimmed to result in an average of 3 utterances to prevent the model learning patterns based on structural features.

- As previously mentioned, only those conversational jokes and tweets in Telugu script were included in the data. For the tweets, the filter 'lang: te' was used for this purpose. This filter is applied on an average to the characters of a tweet. Therefore, if most characters are in the Telugu script, then it is considered 'lang: te'. Due to this, a tweet may not entirely be in Telugu script but may contain words written in roman script. These words can belong to either English or Telugu languages. For example, the words underlined are written in Roman script, some are English words while others are Telugu. These words written in Roman script were detected using the Python package Langdetect[6]. Subsequently, a Python API, Google Transliteration[7], was used.

- In Telugu culture, it is common that the name of the target/butt of the jokes is named *Subbarao*, or *Apparao* and so on. These are similar in theme to *Sardarji* jokes[8]. These names, found abundantly in the humorous dataset, replaced speaker's names in the non-humorous dataset. This was to prevent the model learning that conversations involving speakers having these names implied humor, which is not the case.

4 Outline of Methodology

There are two approaches that are chosen as part of our proposed methodology (Fig. 1) . The first is the use of pre-trained models, which is then further used for our downstream task of Conversational

[4]https://github.com/twintproject/twint
[5]https://developer.twitter.com/en/docs/twitter-api/early-access
[6]https://github.com/shuyo/language-detection
[7]https://pypi.org/project/google-transliteration-api/
[8]https://en.wikipedia.org/wiki/Sardarji_joke

	Humorous Data	Non-Humorous Data
Collected Data	6,107	10,156
Post Filtration	2,047	6,202

Table 1: Statistics of Conversational Dataset

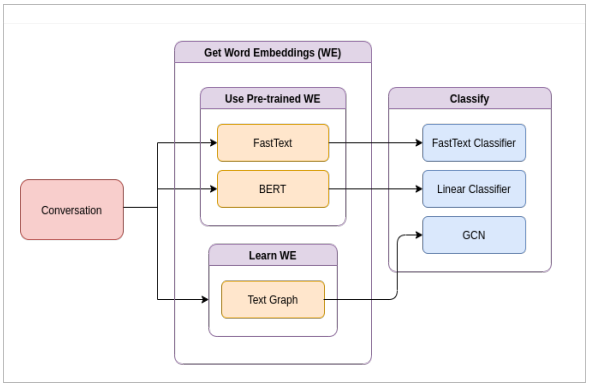

Figure 1: Outline of Methodology

Humor detection. The second is by learning the word embeddings of the conversations using a heterogeneous text graph over which a convolutional neural network is optimized to classify the text. For the first approach we use FastText and BERT's pre-trained models (non-English or dedicated to Indian languages such as Multilingual-BERT or MuRIL) as they have been trained on a large quantity of data. Subsequently, for the second approach, the Text Graphical Convolutional Network (GCN) framework proposed by Yao et. al (2019) is implemented and fine-tuned.

5 Text GCN

5.1 Heterogeneous Graph

There have been several attempts at learning word representations by mapping words and the documents they are a part of to a graph and learning the word-word and word-document relations using different features. In the paper by Yao et. al (2019), using unsupervised learning, they build a heterogenous graph of words and documents. First, this graph is built using word co-occurrence and document-word relations, after which a Text Graphical Convolutional Network is learnt on the corpus.

Using term frequency-inverse document frequency (TF-IDF), edges are constructed between words and documents. Similarly, based on word co-occurrence, edges are constructed between words to make word-word relations. Instead of focusing on word co-occurrence in a single document (in our case, a conversation), a fixed sliding window on all the documents of the corpus is used to calculate global co-occurrence information.

Using PMI (point-wise mutual information), word associations are calculated. These word associations help us capture the relation between word nodes. A positive PMI implies a high association between words. Whereas a negative PMI implies little or no relation between words x and y. Yao et. al (2019) considered only positive values, and only these edges were considered for the heterogenous text graph.

5.2 GCN for Text Classification

A two layered GCN is applied on the built heterogenous graph. A single layer GCN captures information about a node's immediate neighbors, further stacking of such layers integrates a larger neighborhood in its understanding. (2019) experimented with 2 layers of GCN and observed that further stacking of layers did not improve performance. After experimenting, the first layer's embedding size was limited to 200, whereas the window size was set to 20, along with learning rate as 0.02 and dropout rate as 0.5. The training and test data were split into 80:20 ratio and the number of training epochs was stopped early if after 10 successive epochs the validation loss did not decrease.

6 FastText

6.1 Word Embeddings

FastText uses sub-word information to capture a word's representation (Bojanowski et al., 2017). A word is split into character n-grams, and these collectively make up the embedding for a word. FastText is highly advantageous for an agglutinative language such as Telugu (Srinivasu and Manivannan, 2018). In an agglutinative language, a lexeme is attached with suffixes which carry information such as gender, number (singular/plural) or tense. For instance, the word below is made up of smaller morphemes.

intikochindhannamaata

Morphemes that make up the word:

illu + ki + ochuta + i + anamata + emphasis

Translation:

house + to + come + past tense marker + apparently + emphasis

Facebook has released its pre-trained FastText word embeddings for 157 languages[9] including Indian languages like Telugu. For the purposes of this study, Telugu pre-trained word embeddings are used.

6.2 FastText Classifier

The classifier considers a text given as a bag of n-grams. These n-grams are hashed so that their lookup is easier and faster. After fetching the n-grams' embeddings, they are averaged to form the hidden variable of the linear classifier. FastText fetches the pre-trained word vectors of the bag of n-grams and averages it to result in the text's representation. The model was trained for 15 epochs, with the dimension of word embeddings set to 300, the number of word n-grams set to 2 (bigrams), learning rate set to 1.0, and loss set to hierarchical softmax. As it is an imbalanced dataset, class weights are also considered for classification.

7 BERT

7.1 Language Modeling

An important distinction must be made between FastText and BERT. FastText is a Facebook library that offers a word embedding algorithm. However, BERT is a language model. Given a corpus of text, a word embedding algorithm learns a distributed representation of a word based on a window of context. On the contrary, a language model learns the likelihood of a word's appearance in a sequence based on its context (unidirectionally or bidirectionally). BERT can be summed up as a pre-trained model on massive unstructured data using bidirectional encoder representations from Transformers.

BERT considers the complete context surrounding a word and is trained on unlabeled data. BERT contains a stack of encoders such as the Vanilla Transformer that has multi-headed attention layers. It is layered by a feedforward neural network and softmax layer for the purpose of downstream tasks. However, Vanilla Transformers are modeled to look at a word's context unidirectionally. Therefore, to mitigate this disadvantage, BERT uses masking of words/tokens at random so the model can learn the probability of the word appearing in its entire surrounding context over iterations (Devlin et al., 2018).

This pre-trained model can be plugged in for many downstream tasks by taking the pooled output of BERT and passing it to a neural network suitable for the task at hand. An important feature of BERT to be noted is that it does not consider whole words like Word2Vec but rather 'word pieces', and hence is useful for agglutinative languages such as Telugu and can handle unknown or erroneous spelled words.

Numerous pre-trained models are available on HuggingFace. Instead of restricting to using BERT pre-trained models, BERT's cousins AlBERT and DistilBERT are also experimented with. AlBERT and DistilBERT strive to reduce the computational complexity. Specified below are the pre-trained models used for the purposes of this study:

- BERT: Multilingual base model (cased, trained on top 104 languages with the largest Wikipedia), MuRIL (Multilingual Representations for Indian Languages, trained on 17 Indian languages),
- AlBERT: Indic-BERT (ALBERT model pre-trained only on 12 major Indian languages),
- DistilBERT: A distilled version of Multilingual base model (cased)

7.2 Linear Classifier

The pooled output of the BERT (or AlBERT or DistilBERT) encoders is directly sent to a classifier and, after that to a softmax layer. Class weights are considered here, too, due to the imbalanced dataset (2,047 humorous conversations, whereas 6,202 non-humorous conversations). A Pytorch classifier that applies a linear transformation to the given data is used. Subsequently, Cross Entropy loss function is chosen.

8 Results and Discussion

After collecting and pre-processing the tweets and their replies to form non-humorous conversations, experiments were run with Text GCN, FastText and various BERT models (refer to Table 3). The respective model is trained on 80% of the data and is tested on 20% of unseen data. As there are 2,047 instances of humorous conversations whereas 6,202 total instances of non-humorous conversations, this makes it an unbalanced dataset. The weights of the

[9] https://fasttext.cc/docs/en/crawl-vectors.html

Nearest Neighbors	Translation
iMxulO	in this
vAtilO	in those
xInilO	in this
xAnilO	in that
axi	that
vItilO	in these

Table 2: FastText nearest neighbors of word 'aNxulO'

classes are taken into consideration. Accuracy and F1 score are used as evaluation metrics.

Highest accuracy results are obtained by Multilingual BERT by Google. Multilingual BERT is trained on 104 languages using Wikipedia dumps of the respective language. This model is pre-trained with both objectives: Masked Language Modeling (MLM) and Next Sentence Prediction. Additionally, it is observed that FastText and BERT models perform comparatively better than Text GCN. The low F1 score implies a low precision and low recall. This means that the model does not predict that the text is humorous well and sufficiently. In Section 5.1, it is specified that the edges between word-word nodes are given weights based on PMI (point-wise mutual information). The word pair with the highest PMI(12.34) is

<aMxulO, kaxA>

Translation:

<in that, right (in the sentence 'you finished your homework, right?'>

It is evident that both words have no syntactic or semantic relation. Thus, the heterogeneous graph does not capture word relations well. In comparison, FastText's nearest neighbors defined in the pre-trained model for the word 'aNxulO' (translates to 'in that') are shown in Table 2.

By inspecting the nearest neighbors of the queried word, FastText captures syntactic (plural of 'in that' is 'in those') and semantic relations (antonym of 'in that' is 'in this'). This highlights the importance of word embeddings for the model's overall performance for the task to be carried out. Word representations is a key aspect that contributes to the effectiveness and performance of text classification (Shen et al., 2018)(Wang et al., 2018).

As mentioned in Section 3.3, conversational text written in Roman script was transliterated to Telugu script using Google Transliterate API. The API produces an array of most probable transliterations of the input given, after which the first element is considered by default. However, at times, the API does not produce accurate results with a slight margin of error. Thus, potentially producing word(s) that do not exist in the language's vocabulary.

For instance, if the API produces the word *'rANiMcina'*, when it should correctly be *'rAniMcina'* (an error difference of one character) and Text GCN or word embedding algorithms such as Word2Vec encounter this incorrectly transliterated word, it will treat it as an out-of-vocabulary word. Hence, not capturing this unknown word's relation with the correct word *'rAniMcina'*.

FastText and BERT fill this inadequacy by taking sub-word or word-piece information. As the difference between the incorrectly transliterated word and the actual word from the language is one-character, both FastText and BERT will produce word embeddings which will be closer as compared to what TextGCN would produce. For a language such as Telugu, where suffixes are attached to carry semantic and synctactic information, breaking a word into sub-words or word-pieces becomes crucial. Another advantage that transfer learning applications such as FastText and BERT models provide is that they are pre-trained on vast amounts of Wikipedia data and can therefore generalise a word's meaning in a context better (Table 3).

Architecture	Accuracy	F1 Score
Text GCN	0.592	0.374
FastText	0.973	0.946
Multilingual BERT	**0.993**	**0.985**
MuRIL	0.988	0.977
Indic-BERT	0.992	0.982
Multilingual DistilBERT	0.990	0.980

Table 3: Performance of architectures implemented for Conversational Humor recognition

9 Conclusion and Future Work

In this work, the problem of conversational humor detection in Telugu is addressed. Different word embedding algorithms or language models, coupled with different classifiers are used to solve the work at hand. The performances of the various models are evaluated and analyzed to glean insights regarding the mechanisms employed. For low-resource Indian languages such as Telugu, the hurdles that lack of data pose are avoided as pre-trained models on a substantial amount of Telugu

data are used effectively.

The BERT model trained on 104 languages, Multilingual BERT base (cased) by Google, delivered the best performance with an accuracy of 99.3% and an f1 score of 98.5%. Comparatively, FastText comes close with merely a 2% difference in accuracy, 97.3% and an f1 score of 94.6%. State-of-the-art results are thus produced by utilizing transfer learning techniques and methodologies.

Telugu movie scripts could be analyzed to comprehend the trends in the types of humor used in Telugu culture, the influencing factors, and the importance of shared knowledge of culture in the perception of humor (Pamulapati et al., 2020). Instead of using premeditated conversations, real-time conversations transcribed from humorous Telugu interviews would capture the essence of conversational humor better. Detection of humor in conversations could be taken one step further to detect a particular technique(s) or type(s) of Conversational Humor.

References

Issa Annamoradnejad and Gohar Zoghi. 2020. Colbert: Using bert sentence embedding for humor detection. *arXiv preprint arXiv:2004.12765*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Tanishq Chaudhary, Mayank Goel, and Radhika Mamidi. 2021. Towards conversational humor analysis and design. *arXiv preprint arXiv:2103.00536*.

Germaine Dempster. 1932. *Dramatic irony in Chaucer*, volume 3. Stanford University Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Marta Dynel. 2009. Beyond a joke: Types of conversational humour. *Language and Linguistics Compass*, 3(5):1284–1299.

Kristy Beers Fägersten. 2012. *Who's swearing now? The social aspects of conversational swearing*. Cambridge Scholars Publishing.

Ankush Khandelwal, Sahil Swami, Syed S Akhtar, and Manish Shrivastava. 2018. Humor detection in english-hindi code-mixed social media content: Corpus and baseline system. *arXiv preprint arXiv:1806.05513*.

Lizhen Liu, Donghai Zhang, and Wei Song. 2018. Modeling sentiment association in discourse for humor recognition. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 586–591.

Rod A Martin and Herbert M Lefcourt. 1983. Sense of humor as a moderator of the relation between stressors and moods. *Journal of personality and social psychology*, 45(6):1313.

Elizabeth McGee and Mark Shevlin. 2009. Effect of humor on interpersonal attraction and mate selection. *The Journal of psychology*, 143(1):67–77.

A Peter McGraw and Caleb Warren. 2010. Benign violations: Making immoral behavior funny. *Psychological science*, 21(8):1141–1149.

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

John Morreall. 2012. Philosophy of humor.

Vaishnavi Pamulapati, Gayatri Purigilla, and Radhika Mamidi. 2020. A novel annotation schema for conversational humor: Capturing the cultural nuances in kanyasulkam. In *Proceedings of the 14th Linguistic Annotation Workshop*, pages 34–47.

Harvey Sacks, Emanuel A Schegloff, and Gail Jefferson. 1978. A simplest systematics for the organization of turn taking for conversation. In *Studies in the organization of conversational interaction*, pages 7–55. Elsevier.

Emanuel A Schegloff, Gail Jefferson, and Harvey Sacks. 1977. The preference for self-correction in the organization of repair in conversation. *Language*, 53(2):361–382.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. 2018. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. *arXiv preprint arXiv:1805.09843*.

B Srinivasu and R Manivannan. 2018. Computational morphology for telugu. *Journal of Computational and Theoretical Nanoscience*, 15(6-7):2373–2378.

Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018. Joint embedding of words and labels for text classification. *arXiv preprint arXiv:1805.04174*.

Orion Weller and Kevin Seppi. 2019. Humor detection: A transformer gets the last laugh. *arXiv preprint arXiv:1909.00252*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7370–7377.