# CSE 291 Project Report: Jester - An Easy-to-Use Meme Generation System

**Sumanth, Hegde**
Department of Computer Science
UC San Diego La Jolla, CA 92093
s1hegde@ucsd.edu

**Wonsuk, Jang**
Department of Computer Science
UC San Diego La Jolla, CA 92093
w1jang@ucsd.edu

**Yash, Khandelwal**
Department of Computer Science
UC San Diego La Jolla, CA 92093
ykhandelwal@ucsd.edu

## Abstract

Memes are an important part of culture, being used for exchanging ideas in casual, everyday conversations as well as in social media marketing. Our work aims to make the process of generating memes faster and easier by leveraging deep generative models. We propose Jester, a two stage solution to address a text-to-meme translation task[1]. Our model first retrieves relevant images based on the sentiment in the user's text, and further generates candidate prompts which capture the user's intent and scenario. We incorporate the user's input at every stage to allow for as much customization as possible. Unlike previous work, our system is flexible. and can easily incorporate new examples and templates. Our method improves over the baselines in both qualitative and quantitative metrics. On a dataset of 100 templates, we achieve a top-10 accuracy of 77% for template retrieval, and human-evaluated relevance and coherence scores of 3.0 /4 and 2.9/4.

## 1 Introduction

**Problem Definition.** Memes are used and shared among many social media users to convey ideas through humor and reference to popular culture. Although there are tools([1], [2]) that allow users to generate memes, there are almost no open source tools that allow users to generate custom memes from simple user text. As we want to help users create memes easily while allowing them to select specific captions and images based on their needs, our project seeks to create a deep learning framework that uses deep generative models to help the user create a custom meme based on their input text.

**Problem Significance.** Our project aims to help users create a meme by matching the best image template and captions based on the input text. In order to solve the problem, we encode the meaning of the input text and map it to the best image appropriate to serve as a template to the meme. To do so, we train deep learning models for text-to-image retrieval to create a model that can map from input text to an appropriate meme template by training on a dataset of 300 meme templates with 3000 captions per image. Additionally, we utilize a pre-trained text generation model, GPT-3[3] through few-shot learning to help facilitate caption prompt generation in order to generate appropriate captions for the meme template.

[1]The code for the implementation can be found here: https://github.com/SumanthRH/text-to-meme

Preprint. Under Review.

Internet memes can have important implications for public discourse and commentary in a participatory media, environment, even bringing to attention societal issues such as xenophobia and discrimination against minority groups[4], [5]. Additionally, coupled with today's social media, memes can be spread all over the world for others to relate to and agree with, making them an important part of marketing[6]. By allowing users to create their own custom memes on any topic through a simple process, it makes it much simpler for one to communicate their ideas. Therefore, our project can provide an easily accessible tool to others that will make them to create memes easily and share their opinions (or even market their product) with others around the world.

**Technical Challenge.** There are a few meme generation tools available for free on the internet[1], [2]. The alternative to our system is a website([1], [2]) that requires the user to search for a relevant meme template and come up with a good caption. Our aim is to outsource some of the creativity required from the user, all the while building a system that is faster than if the user were to create a custom meme themselves. Our project requires training a text-to-image retrieval system that can quickly serve a bunch of relevant meme templates to the user in a matter of seconds. Further, we need to train general purpose text generation models like GPT-3 [3] through few-shot learning, which requires high-quality data labelling so that the model can generate a coherent, relevant caption for a meme template. Memes can also go in and out of fashion quickly, so our system needs to be flexible to incorporate new templates easily without significant re-training costs. Our goal is a consumer-facing web application[2] which requires us to engineer a full-stack solution to deploy our model on a server, retrieve data from a large image database, etc.

**State-of-the-Art.** Traditionally, previous works have focused on generating suitable captions given a meme template ([7], [8]). MemeBot [9] is a text-to-meme generation system that is trained end-to-end. The task of finding the relevant meme template given user text is formulated as a classification task employing Roberta[10], a transformer[11] based architecture. An embedding of the template, along with the noun and verb phrases in the input text is fed to another transformer to generate the final caption. MemeBot, on a custom Twitter dataset, achieves a BLEU-4 score of 11.12, along with human-evaluated (refer 4) Coherence score of 2.66 and Relevance score of 2.65. Supermeme[12] is a meme generation tool with a text-to-meme generation tool much like ours. However, the meme templates are often not relevant to the user's text. Unfortunately, the authors do not provide a free API that can be used to query their model.

**Contributions** Our main contributions are as follows:

- We solve a novel problem of text-to-meme generation, delegating the creativity required in creating memes to a deep generative model.
- We propose a two-stage solution, which uses a sentence transformer[13] and GPT-3 to outperform existing baselines in quantitative and qualitative metrics.
- Jester is extensible and can easily incorporate new examples and meme templates.

## 2 Related Works

The task of generating memes can be aligned with tasks such as text-to-image retrieval, image caption generation and natural language generation. Our work can be categorized in the domain of conditional text generation and text-to-image retrieval. [14] presents a model to classify a sentence based on sentiment. Works such as [15] and [16] present image captioning models that encode the visual features of an image into a vector which is fed into a recurrent neural network for text generation. In [17], the authors present CLIP, a neural network trained on a large corpus of (image, text) pairs. The network is able to learn visual concepts from raw text and demonstrates strong zero shot capabilities. Given a text snipped, CLIP can predict the most relevant image from a set of images without directly optimizing for the task.

The field of natural language generation has received significant attention over the past few years. Conditional text generation is the task of generating text with a certain sentiment, topic or constraint. [18] discuss a variational autoencoder based model that can generate text conditioned on a desired

[2]The web application can be found here: https://the-jester.streamlit.app

sentiment (positive, negative, etc). Recent work in natural language generation has been clustered around transformer architectures [11]. BERT[19] is a transformer based model that learns a bidirectional representation for natural language and can be used for a variety of downstream tasks such as question answering and natural language inference. The text-to-image retrieval task can also be cast as one of semantic search, wherein new user prompts are matched with meme templates solely based on prompts seen during training, without using the visual information. Sentence-BERT[13] uses a siamese network structure to train BERT on a semantic textual similarity task. Sentence-BERT maps sentences and paragraphs to a 384 dimensional vector and can be used for clustering and semantic search.

Previous works on meme generation have focused on a caption generation or a caption selection task. In [8], the authors introduce a caption generation system that can generate a relevant caption for a given meme template. The most relevant work to our text-to-meme generation model is MemeBot[9]. The MemeBot model consists of two components: a meme template selection module and a meme caption generator module. For a given input sentence, an image meme is generated by combining a meme template image and a text caption where the meme template image is selected from a set of candidate template using the selection module, and the meme caption is generated by an encoder-decoder model. The encoder maps the selected meme template and the input sentence into an embedding and the decoder decodes the caption from the embedding. MemeBot is trained on a dataset of meme captions and templates. The decoder uses the template embedding and the noun phrases and verbs in the input text to generate the full meme caption.

Our work has a similar problem formulation as MemeBot[9]. However, existing datasets only contain meme captions, while we want to operate on user prompts, which are going to have an everyday, unconstrained conversational tone. MemeBot uses the noun and verb phrases in the user text to generate a meme caption. However, this does not capture the full context in the user text, and can completely miss the user sentiment as shown in Appendix A. Since we do not have a dataset of user prompts and meme captions, we resort to labelling the data ourselves. Using a powerful few shot learner like GPT-3[3], we are able to generate relevant, coherent captions from just 5 example (user prompt, meme caption) pairs per template (ref Section 4).

## 3 Methodology

**Problem Setting.** For the text-to-image retrieval module of our system, we are interesting in finding the best template $T_{\text{best}}$ for a given user prompt $S$. This can be treated as a classification task with the following loss function[9]:

$$L(\theta) = \underset{\theta}{\text{argmax}} \sum_{(T,S)} \log((P(T|S,\theta))) \tag{1}$$

where $\theta$ is the parameters of the text-to-image retrieval module, and $T$ is the meme template (a label from 0-99, since we deal with 100 templates) and $S$ is the input sentence, with a maximum tokenized length of 50 tokens. Instead of classifying text into a fixed set of labels, we use a softmax-free approach based on clustering semantically similar sentences 3. Now, for the user prompt $S$, we generate a relevant meme caption through few shot learning[3]. We label 5 examples of a user input for a meme caption. This is fed as input to a conditional text generation model (GPT-3) which generates a full meme caption based on these examples. An example prompt to GPT-3 is shown in the Appendix B.

**Idea Summary.** Figure 1 shows the system overview for our proposed text to meme generation system. The flow starts with a user supplying a sentence in colloquial English, such as "Are you kidding me right now?". The Text-to-Image retrieval system is a deep classification model which is trained on a dataset of memes and captions. The model will retrieve the $k$ most relevant images based on the sentiment expressed in the user input. In Figure 1, we retrieve 3 different meme templates: "Not sure if", "Serious face", and "Side eyes chloe". After image retrieval, the user selects a meme template with which they relate to the most, in this example "Not sure if". The metadata of the template, some template-specific examples and the user input is then fed to GPT-3 [20]. Through few-shot learning, the transformer model generates a set of captions that capture the user's intent within the particular language style of that meme template. Finally, the user gets to pick their choice of caption which is then overlaid onto the meme template to generate the final output. In this example,

the user selects the caption "Not sure if joking <sep> or serious" and overlays it over the "Not sure if" meme template to create the custom meme (<sep> separates the caption into top and bottom parts as shown in Figure 1).

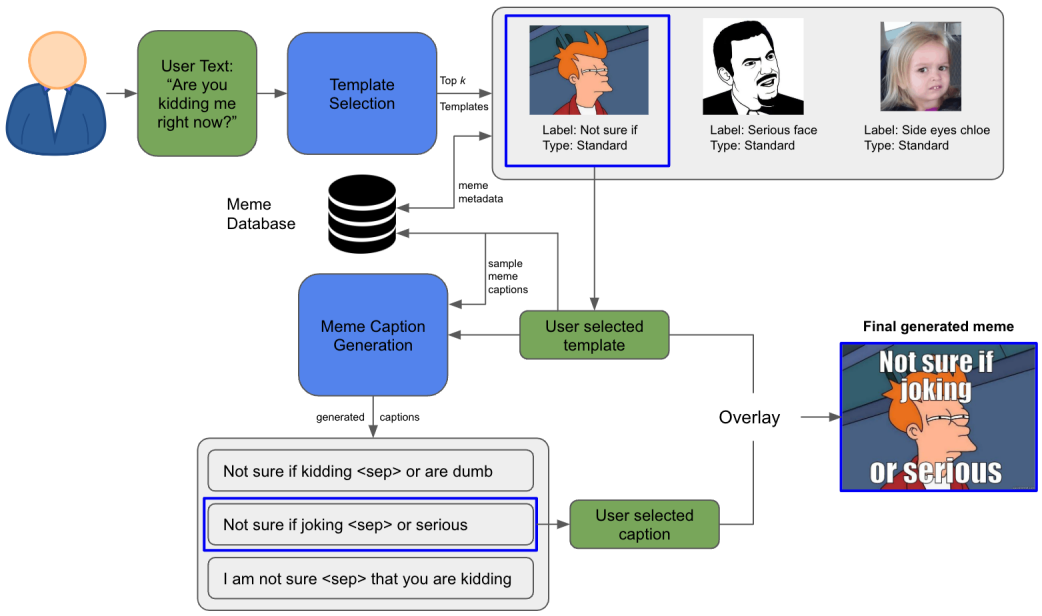



Figure 1: Jester System Overview: *The system consists of two stages (blue boxes): a.) Template selection; b.) Meme caption generation, conditioned on user text and the template information.*

**Description.**

1. **Meme Template Selection System**
   The meme classification system uses a fine-tuned sentence transformer model [13]. Our model is shown in Figure 2. The idea is to create a clustering of various meme templates in a high-dimensional vector space. During training, a RoBERTa model is trained as a siamese network, minimizing the distance between sentence embeddings for the same meme template and maximizing the distance between embeddings for different templates. We used Contrastive Loss [21] for training which is given by:

$$L(y, \mathbf{z}_1, \mathbf{z}_2) = (1-y)\frac{1}{2}D(\mathbf{z}_1, \mathbf{z}_2)^2 + y\frac{1}{2}\{\max(0, m - D(\mathbf{z}_1, \mathbf{z}_2))\}^2 \quad (2)$$

   In above equation, $y$ indicates whether the sentence embedding vectors $\mathbf{z}_1$ and $\mathbf{z}_2$ are from the same category(template), $D(\cdot, \cdot)$ is the similarity function (such as cosine similarity), and $m$ is the margin. For each of the meme templates, we get an average embedding vector of all the meme captions in the training data (*template embedding*). For any user input, we can classify the text based on the cosine similarity score between the text and the template embeddings for all the meme templates. Compared to a RoBERTa model stacked with a classifier, this model does not require architecture re-design whenever a new meme template is introduced. It can just be fine-tuned for a new meme template instead of re-training the complete model from scratch. Thus, it is flexible and extensible.

2. **Caption Generation** The prompt is transformed into a meme caption using a generative transformer, GPT-3. Here we are tasked with conditional text generation under a few shot learning regime. Given $K$ (we use 5) examples of context and a completion, the model is expected to a provide a suitable completion for the final example. The context is template dependent (example: For the "Not sure if" meme, the caption is short, conveys confusion and always begins with "Not sure if"), and thus we label different <user prompt, meme caption> pairs for different templates.

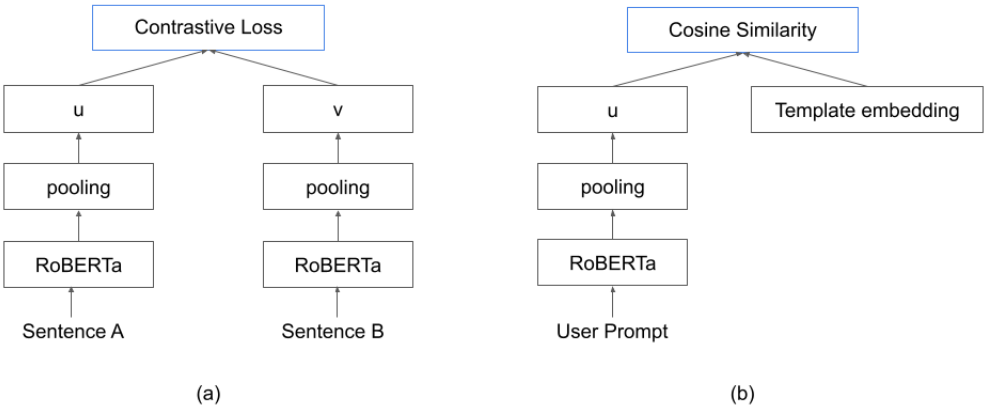



Figure 2: (a) Sentence-RoBERTa architecture during training. The two RoBERTa networks have tied weights (b) Sentence-RoBERTa architecture during inference. The embedding $u$ of a user prompt is compared with the embedding of every template

**Implementation:**

1. **Meme Template Selection System** In the Sentence-RoBERTa model, we follow Sentence-BERT[13] and use mean pooling across all the token embeddings to get the sentence embedding. For the loss function, we dynamically generate two positive and two negative examples for each data-point in each epoch. We set $m = 0.5$ and use cosine similarity for $D(\cdot, \cdot)$. PyTorch[22] is used for the implementation, and we utilize pre-trained RoBERTa weights from the HuggingFace[23] model hub. We used the AdamW optimizer[24] with a linear warmup scheduler with 100 warmup steps. We did not find significant benefits with other schedulers (we tried Cosine and Constant Warmup schedulers).
2. **Caption Generation** For each prompt to GPT-3 we add the label for the meme (In the example in Figure 1, this would be "Not sure if"), provide 5 example <user text, meme caption> pairs that we annotate, and finally add in the user prompt for which we need the meme caption. We set temperature to be 0.7, with other parameters being set to default (no frequency or presence penalty, `top_p` $= 1$). An example prompt is shown in Appendix B.

## 4 Experiments

**Datasets and Tools.** The dataset we are using for our project is based on the Deep Humor Meme dataset, which consists of 300 meme templates with 3000 captions each [25]. We provide five examples of (user prompt, meme caption) pairs to GPT-3 through manual labelling. We do this for 100 meme templates and thus use the same subset for training the Sentence-RoBERTa model, the baselines and the metrics. We use a 8:1:1 train-validation-test split for Sentence-RoBERTa. We thus have 210,000 captions for training and 30,000 for validation and testing. We use Streamlit Cloud to setup a basic web demo for our model.

**Baselines.**

- MemeBot[9] : While we do not have access to the implementation of MemeBot, we compare our model's qualitative results with MemeBot2.
- Random Sample Baseline: Since we do not have access to MemeBot[9], we started out with the simplest possible baseline for our model. In this case, for a given text input, we simply retrieve a random meme template from the dataset. Given a user prompt, our main idea is

that we need a machine learning model that can retrieve a relevant template for the image. For this baseline, we sample a random meme template for a given prompt. This helps us understand how "relevant" a random meme template would be for a user prompt, and how it would differ from a custom, "best match" meme template i.e a meme template that is most "relevant" for this user prompt.

- CLIP (Zero-shot): As another baseline for the text-to-meme retrieval system, we evaluate the zero-shot performance of CLIP[17]. For each caption, we get the text encoding using CLIP and extract similarity scores with the image encodings for all the meme templates. Through this, we can obtain the best match for a candidate caption.
- RoBERTa: We train a RoBERTa[10] to classify each prompt into one of the 100 categories (templates). This is our baseline as well as the gold standard for Top-1 accuracy, because having a softmax-based cross-entropy loss would better enforce different embeddings for different classes, at the cost of lesser flexibility. We show that our model is able to closely match this baseline in Table 1.

**Evaluation Metrics.** We will use the following quantitative metrics:

- Top-$k$ accuracy: For our text-to-meme retrieval system, we measure the top-$k$ accuracy : the percentage of results where the true meme template is present in the top $k$ results for the text. (We report values for $k = 1, 5, 10$)
- Meme Template Relevance (MTR): To quantify "relevance", we measure cosine similarity of sentence embeddings generated from a pre-trained Sentence Transformer model[13]. We use a MiniLM[26] model pre-trained for clustering on datasets like the Reddit comments dataset[27] and the S2ORC [28] dataset. More concretely, on the test set for the model, we do the following:
  - Represent a meme template as the average value of the sentence embeddings of the training dataset.
  - For each method, compute cosine similarity between the embedding (from the Sentence Transformer model) for a candidate caption and the retrieved (best match) meme template.

  In the end, we get an average value of similarities across the 100 meme templates in our dataset. The ideal case Meme Template Relevance comes out to be 0.40. The Sentence Transformer might not capture implicit sarcasm in the caption (since it is trained on traditional datasets collected through web crawling). Further, a variety of captions (with different sentiments) can go with a given template. We suspect these reasons give a low score of $0.40$ for the ideal case.

Subjective scores, obtained through human evaluation of a set of generated memes, can also be used. Following [9], we use the following:

- Coherence: Can you understand the message conveyed through the meme (Image + text)? This score is rated from a score of 1 - 4 with 1 being incoherent and 4 being clear.
- Relevance: Is the meme contextually relevant to the text? This metric will follow the same idea for the evaluation metric as coherence.
- User Like: Did the user like the generated meme?

In order to obtain the human evaluation scores, we created a google form in which the users can provide the Coherence, Relevance, and User Likes for the generated memes. We surveyed students of CSE 291B as well as our friends to evaluate the scores for our generated memes (Section 4).

**Quantitative Results.** The quantitative results for the model are summarized in Table 1. To obtain a reliable measure for the Random Sample Baseline, we average the MTR over $50$ random templates. In terms of the classification metrics (Top-1, Top-5, Top-10), the SRoBERTa is able to closely match the RoBERTa.

| | MTR↑ | Top-1(%)↑ | Top-5(%)↑ | Top-10(%)↑ |
|---|---|---|---|---|
| Random Sample | 0.26 | - | - | - |
| CLIP | 0.32 | 16.35 | 32.95 | 42.20 |
| RoBERTa | **0.40** | **49.31** | 68.75 | 75.31 |
| **SRoBERTa** | **0.40** | 48.75 | **70.88** | **77.65** |

Table 1: Quantitative Results. While RoBERTa has better Top-1 accuracy, SRoBERTa performs better in Top-5 and Top-10 accuracy scores.

**Qualitative Results.** To evaluate the effectiveness of our model, we created a survey consisting of 10 different user prompts. For each prompt, we provide 3 memes using the top 3 meme templates retrieved by SRoBERTa and the top caption generated by GPT-3. After generating our samples, we sent out the survey asking users to evaluate the coherence and relevance of each generated meme as well as whether the users liked them or not (defined in 4). For our survey, we had 30 responses with the following results as outlined in Table 2.

| Model | Relevance | Coherence | User Likes |
|---|---|---|---|
| MemeBot | 2.65 | 2.66 | **0.65** |
| Our Model | **3.0** | **2.9** | 0.58 |

Table 2: Qualitative Results for our model. Our model is able to achieve better relevance and coherence scores, while performing a bit worse in terms of user likes

From our results, we see that our model has clearly outperformed MemeBot in terms of relevance and coherence. We underperform in terms of user likes, with a score of 0.58 than MemeBot's 0.65. We suspect this is mainly due to the very subjective nature of what a "good" meme is.

We have included sample memes across different relevance and coherence scores in Figures 5 and 6 in Appendix C. Jester is able to generate grammatically coherent captions across all relevance and coherence scores, while MemeBot can generate highly incoherent sentences.

**Ablative Studies** Here, we verify the contributions of the two main stages of our model: we study the improvements of using SRoBERTa over CLIP[17] as the template retrieval/selection system and the improvement of using GPT-3 over no caption generation (i.e if we directly use the user prompt as the meme caption). Concretely, we present sample memes in Figures 3 and 4 for the following four models:

(a). CLIP : We simply use CLIP to retrieve a relevant template and use the user prompt as the meme caption.

(b). CLIP+GPT-3 : We use CLIP along with GPT-3 for caption generation.

(c). SRoBERTa : We use SRoBERTa for retrieving a relevant template without caption generation.

(d). SRoBERTa+GPT-3 : This is our proposed method.

SRoBERTa is able to better detect the sentiment and retrieve a relevant template for the user prompt, while GPT-3 is able to provide the caption that suites the template and the prompt.


(a) **Prompt**: Give me the gummy bears, and then you can leave

(b) **Prompt**: Do not touch my fries ever again

(c) **Prompt**: Spent 10 hours making social media marketing material

(d) **Prompt**: We've gotta prepare ourselves, the poster presentation is here!

Figure 3: Results for the ablation study. These are four example memes for CLIP (Top) and CLIP+GPT-3 (Bottom). The addition of GPT-3 improves the coherence of the meme caption with the template.


(a) **Prompt**: Give me the gummy bears, and then you can leave

(b) **Prompt**: Do not touch my fries ever again

(c) **Prompt**: Spent 10 hours making social media marketing material

(d) **Prompt**: We've gotta prepare ourselves, the poster presentation is here!

Figure 4: Results for the ablation study. These are four example memes for SRoBERTa (Top) and SRoBERTa+GPT-3 (Bottom). SRoBERTa is able to retrieve more relevant templates than CLIP, and the addition of GPT-3 improves coherence.

## 5 Conclusion and Discussion

The goal of this project was to generate high-quality memes which capture the user intent by delegating the creativity to generative models. To achieve this goal, we developed Jester, a flexible text-to-meme generative model. The model uses Sentence-RoBERTa for template selection and GPT-3 for caption generation. We trained a Sentence-RoBERTa transformer model on 100 templates (classes) and

primed GPT-3 with 5 examples of caption generation per template. Our method improves over the baselines in both quantitative and qualitative metrics. Future work includes extending Jester to readily incorporate new data in a deployment pipeline. Memes also eventually fall out of public use, so meme popularity scores can be added so that the model forgets unpopular memes over time. The dataset quality, specifically in terms of filtering out hate speech is also a crucial element. We believe that future work should also target improvements both in terms of refined datasets and safeguards at the application level.

## 6 Acknowledgements

We would like to acknowledge Prof.Rose Yu for helpful discussions. We would also like to thank Genesis Cloud for providing additional compute hours.

## References

[1] Dylan Wenzlau. Imgflip: Create and share awesome images., 2011.

[2] Make a meme - funny memes and meme generator.

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020.

[4] Heidi E Huntington. Subversive memes: Internet memes as a form of visual rhetoric. *AoIR Selected Papers of Internet Research*, 2013.

[5] Limor Shifman. Memes in a Digital World: Reconciling with a Conceptual Troublemaker. *Journal of Computer-Mediated Communication*, 18(3):362–377, 04 2013.

[6] Ganna Kostygina, Hy Tran, Steven Binns, Glen Szczypka, Sherry Emery, Donna Vallone, and Elizabeth Hair. Boosting health campaign reach and engagement through use of social media influencers and memes. *Social Media+ Society*, 6(2):2056305120912475, 2020.

[7] Dylan Wenzlau. Meme text generation with a deep convolutional network in keras & tensorflow, Apr 2019.

[8] Abel L. Peirson V and E. Meltem Tolunay. Dank learning: Generating memes using deep neural networks. *CoRR*, abs/1806.04510, 2018.

[9] Aadhavan Sadasivam, Kausic Gunasekar, Hasan Davulcu, and Yezhou Yang. memebot: Towards automatic image meme generation. *CoRR*, abs/2004.14571, 2020.

[10] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[12] Ramsri Golla Sanjeev NC, Nico Botha. Supermeme: Generate original memes powered by ai in 110+ languages, 2022.

[13] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

[14] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[15] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.

[16] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[17] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

[18] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Toward controlled generation of text. In *International conference on machine learning*, pages 1587–1596. PMLR, 2017.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[20] Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. Gpt-neox-20b: An open-source autoregressive language model, 2022.

[21] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[22] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[23] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[25] Ilya Borovik, Bulat Khabibullin, Vladislav Kniazev, Oluwafemi Olaleke, and Zakhar Pichugin. Deephumor: Image-based meme generation using deep learning.

[26] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788, 2020.

[27] Matthew Henderson, Paweł Budzianowski, Inigo Casanueva, Sam Coope, Daniela Gerz, Girish Kumar, Nikola Mrkšić, Georgios Spithourakis, Pei-Hao Su, Ivan Vulić, et al. A repository of conversational datasets. *arXiv preprint arXiv:1904.06472*, 2019.

[28] Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online, July 2020. Association for Computational Linguistics.

[29] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649, 2018.

## Appendix A Shortcomings of Phrase-based caption generation

Given below are some examples where the noun and verb phrases will fail to capture the full meaning of input sentences. We use a state of the art phrase chunking model[29]. We will use NP and VP to denote noun and verb phrases respectively. Note that the this list of phrases, along with the meme template information, is passed as input to the caption generation model.

**Prompt:** Why is this commercial not the same volume as the show ughh
**Input for Caption generation:** "this commercial" (NP), "the same volume" (NP), "the show" (NP)
**Explanation:** The phrase chunking drops "why" and "not", which are important in capturing the sentiment of the user. The sentence "This commercial is the same volume as the show" results in the same set of phrases.

**Prompt:** How can one be this dense?
**Input for Caption generation:** "one" (NP), "be" (VP)
**Explanation:** The caption generation model is only given "one" and "be" and misses out on the phrases "How can" and "this dense", completely losing out on the original sentiment of the prompt.

**Prompt:** Imagine a world without machine learning
**Input for Caption generation:** "Imagine" (VP), "a world" (NP), "machine learning" (NP)
**Explanation:** The caption generation model drops the preposition "without", therefore changing the meaning of the sentence.

**Prompt:** Never say never
**Input for Caption generation:** "say" (VP)
**Explanation:** The sentence transformer classifies "never" as a preposition and drops it.

**Prompt:** You should definitely try that again
**Input for Caption generation:** "You" (NP), "should definitely try" (VP), "that" (NP)
**Explanation:** The caption generation model classifies "again" as an adverb and drops it.

One can see that phrase-based caption generation requires the model to hallucinate a lot more details and does not provide accurate context to the model.

## Appendix B Prompt for GPT-3

Given below is the prompt crafted for the 'koala can't believe it' meme template. The black text represents the hard-coded prompt template. The magenta text changes as per the template, and the blue text is the user input.

```
Give a humourous, witty meme caption based on the input provided. The label
of this meme is 'koala cant believe it'

input: you telling me the leaves are actually poisonous?
output: what?<sep>are the leaves poisonous?

input: are you kidding me? 4/20 was yesterday? damn
output: wait.<sep>what do you mean 4/20 was yesterday?!

input: wait a sec, karren did what!
output: hold the phone<sep>karren did what!

input: oh my god I forgot my potpie was in oven after I left for work rip
output: left for work<sep>forgot potpie in oven

input: damn it's thursday already man, time flies
output: what?!<sep>it's thursday already?!

input:<user text>
output:
```

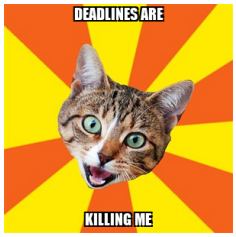
## Appendix C Sample Memes Across Different Relevance and Coherence Scores

Here we present sample memes from Jester used in the human evaluation survey. The survey asks raters to input how much they agree or disagree with the following statements:

- The meme is contextually relevant to the text (Relevance)
- The message conveyed through the meme (Image + text) is consistent. (Coherence)

We present sample memes across different relevance and coherence for Jester and Memebot[9] below. As studied in Section 4, the baseline for caption generation is an identity mapping i.e the meme caption is just the user prompt. In both Figures 5 and 6, Jester generates gramatically coherent captions throughout, while Memebot sometimes generates gramatically incoherent captions. For the cases with the lowest coherence and relevance scores, Jester sometimes serves a template that is not that relevant. Further, since we use a relatively high temperature setting for GPT-3, the captions sometimes do not capture user intent. Better quality examples for GPT-3 can alleviate this issue.

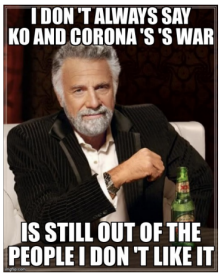

(a) **Strongly Disagree**, Caption = "Only took me 10 tries to clear the tutorial"



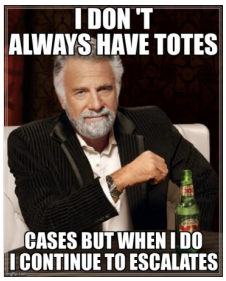

(b) **Disagree**, Caption = "Deadlines are killing me"


(c) **Agree**, Caption = "Give me the gummy bears, and then you can leave"


(d) **Strongly Agree**, Caption = "Forgot to turn the lights off before my trip"

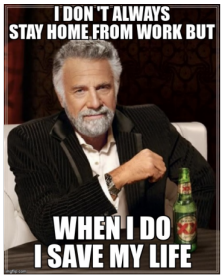



(e) **Strongly Disagree**, Caption = "People say the corona virus is like an invisible war"



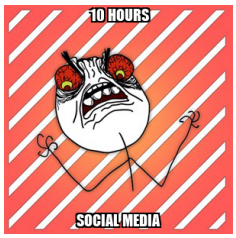

(f) **Disagree**, Caption = "Corona virus cases continue to escalates."


(g) **Agree**, Caption = "It's illegal to post fake information about corona virus."



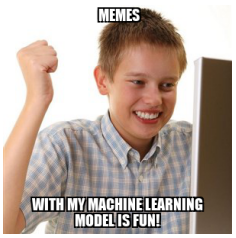

(h) **Strongly Agree**, Caption = "Stay at home and save your life. "

Figure 5: Memes Across Different Relevance Scores for Jester (Top) and Memebot[9] (Bottom). Jester generates high quality captions throughout, while captions from Memebot are gramatically incoherent.

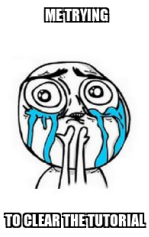



(a) **Strongly Disagree**, Caption = "Spent 10 hours making social media marketing material"


(b) **Disagree**, Caption = "Forgot to turn the lights off before my trip"

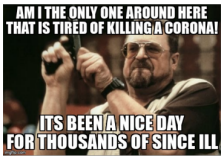



(c) **Agree**, Caption = "Generating memes with your own machine learning model is fun!"

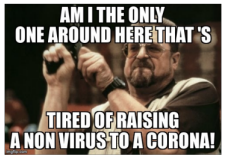



(d) **Strongly Agree**, Caption = "Only took me 10 tries to clear the tutorial"

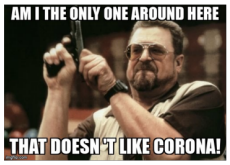

(e) **Strongly Disagree**, Caption = "People say the corona virus is like an invisible war"




(f) **Disagree**, Caption = "Corona virus cases continue to escalates."




(g) **Agree**, Caption = "It's illegal to post fake information about corona virus."




(h) **Strongly Agree**, Caption = "Stay at home and save your life. "

Figure 6: Memes Across Different Coherence Scores for Jester (Top) and Memebot[9] (Bottom)