Visual Semantic Composition for Meme Generation

Aman Shrivastava
Department of Computer Science, University of Virginia
as3ek@virginia.edu

Abstract

Over recent years, extraordinary work has been done in conditional natural language generation. However, an area that is relatively unexplored is computational humor. In the internet age, memes have become the primary form of media to communicate humor. In this project we aim to create joint visual semantic embedding space that can represent the visual contents of a meme template composed with the emotion and tone it conveys. These embeddings can then be further used to generate humorous captions given a meme template.

1. Introduction

Language is used to evoke and convey thoughts, feelings, and emotions. One of the emotions commonly expressed through language is humor. Through evolution humans developed a sense of humor due to its biological function of communicating information effectively in processed form in addition to invoking a positive emotional reaction. Human brains, therefore, have an innate ability to syntactically detect and generate humorous sentences without significant training. In this project, we intend to explore if a neural network can mimic this syntactic ability by training it on human generated humorous meme captions.

Individual memes templates generally represent a distinct tone and context that is not always detectable form the visual content of the image. This problem complicates the established task of image captioning even further, as there is significant variation in the captions generated on the same template as long as the captions generated follow the semantic setup-punchline formulation associated with the given meme template. Therefore, current image captioning methods [9, 6] fail to capture the complexity of the problem setup as it is not enough to condition the caption generation module of the architecture just on the features extracted from the image. In this project, we propose that conditioning the text-decoder module of the architecture with features obtained by composing the image features with semantic features extracted from additional word tokens (henceforth referred as modifiers) that represent the theme of each caption.

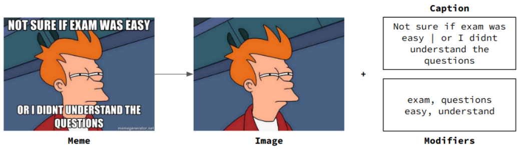

NOT SURE IF EXAM WAS EASY
OR I DIDNT UNDERSTAND THE QUESTIONS
memegenerator.net
Meme
Image
+
Caption
Not sure if exam was easy | or I didnt understand the questions
exam, questions easy, understand
Modifiers

Figure 1. Representative example from the data scrapped from memegenerator.net

To this effect, we aim to design a composed latent space that combines the visual contents of an image with contextual semantic representation of the meme. We aim to achieve this by designing a training mechanism that continually uses the visual representation of the meme template and the semantic representation of the captions to create a composed latent space. This representation can then be used to condition a text generation module to create novel captions.

2. Related Work

With the advent of sequence-to-sequence models for language understanding and CNNs for generating image embedding, the field of image captioning [6, 9] has seen tremendous progress. Although generating and understanding humor has been a long standing challenge in the field of Natural Language Processing, there has been limited research that combines text and vision to generate humor. Researchers at Stanford demonstrated the use of an encoder-decoder based image captioning system to generate humorous captions [7]. Exploring a different modus-operandi, researchers at CMU developed a Nonparanormal approach for generating meme descriptions [8], which uses a nonparanormal ranking model to identify the funniest caption.

In recent years, much work has been done on multimodal alignment of latent spaces for image captioning [1, 2] and for image retrieval [3, 4, 5]. We intend to build on this work to generate visual-semantic embeddings for meme templates given the image, contextual descriptors and the humorous captions.

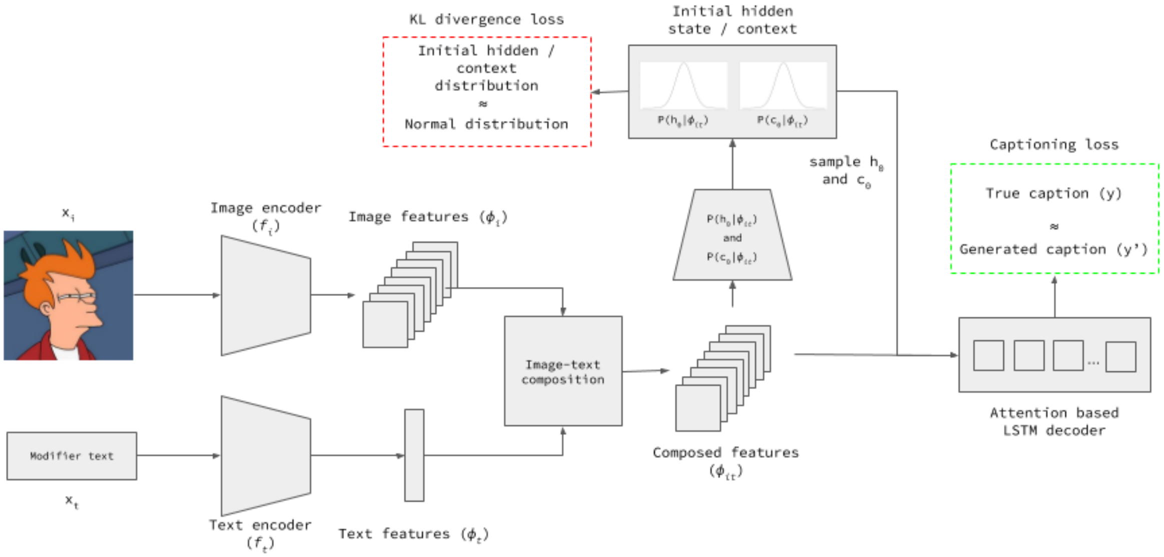

KL divergence loss
Initial hidden / context distribution ≈ Normal distribution
Initial hidden state / context
P(h_0|ϕ_{it})
P(c_0|ϕ_{it})
sample h_0 and c_0
Captioning loss
True caption (y) ≈ Generated caption (y')
x_i
Image encoder (f_i)
Image features (ϕ_i)
P(h_0|ϕ_{it}) and P(c_0|ϕ_{it})
Image-text composition
Composed features (ϕ_{it})
Attention based LSTM decoder
Modifier text
x_t
Text encoder (f_t)
Text features (ϕ_t)

Figure 2. Schematic representation of the model architecture

3. Data

The dataset for training the model has been scrapped from memegenerator.net. The dataset consists of 1200 meme templates as images and between 1000 captions for each template. The dataset overall consists of around 1.2 million image-caption pairs. The text-modifiers used to generate the thematic embedding has been extracted from the description of the meme template as well as operative words in each caption (Figure 1).

4. Model

In this work we propose an encoder-decoder based architecture that first composes the representations extracted from the image and the modifier-text to generate a composed latent representation. This representation is then fed into a text-decoder that generates a predicted caption. First, we use a CNN to get a $2d$ spatial feature vector $f_i(x_i) = \phi_i \in \mathbb{R}^{W \times H \times C}$, where W is the width, H is the height and C is the number of feature channels and is dependent on the choice of the encoder CNN architecture. Further, we aggregate the the embeddings for the modifiers and use a fully connected network as $f_t(x_t) = \phi_t \in \mathbb{R}^d$ where d is the number of dimensions in the representation. We then use our composition approach to compute $\phi_{it} = f_{com}(\phi_x, \phi_t)$.

4.1. Image-Text Composition

In order to combine image and text features using the Text Image Residual Gating [3] approach which is described as:

$$\phi_{it} = w_g f_{gate}(\phi_i, \phi_t) + w_r f_{res}(\phi_i, \phi_t)$$

where w_g, w_r are learnable weights, the gating function is described as:

$$f_{gate}(\phi_i, \phi_t) = \sigma(W_{g2} \times RELU(W_{g1} \times [\phi_i, \phi_t])) \odot \phi_i$$

where σ is the sigmoid function, $\odot$ is element-wise product, and W_{g1}, W_{g2} represent convolutional layers with 3×3 filters. Also, the text feature vector ϕ_t is braodcasted along height and width to match the dimensions of the image feature map ϕ_i. Further, the residual connection is computed by:

$$f_{res}(\phi_i, \phi_t) = W_{r2} \times RELU(W_{r1} \times [\phi_i, \phi_t])$$

where W_{r1}, W_{r2} again represent convolutional operations with 3×3 filters.

4.2. Caption generation

In the decoder part of the architecture, we use a long short-term memory (LSTM) network [11] that produces the caption by generating one word at each time step conditioned on a context vector, the previous hidden state and the

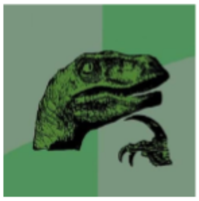
Input image	Modifiers	Generated caption
	pride	Oh you are a vegan | you must be so proud
	friends	Oh you have friends | I ate them
	late	Oh you are here | you must have wheels
	doctor	Goes to a dentist | gets malaria
	eat	Eats popcorn | movie gets cancelled
	lazy	Wants to go to the bathroom | too lazy
	game	What if the world is a game | we are the same
	lizard	What if | I am a lizard
	think	What if | you think I am a stupid

Figure 3. Captions generated by the model given the meme template image and the modifier tokens. **Note:** The pipe symbol (|) has been added manually to enhance the setup-punchline formulation of the generated jokes.

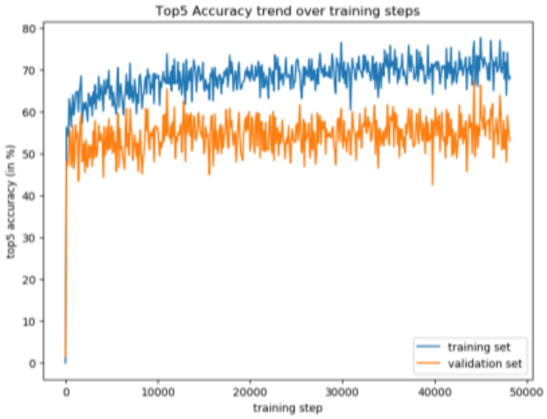


Figure 4. Top5 accuracy of the model on the training and the validation set

previously generated words. The context vector here is a dynamic representation of the relevant part of the composed feature vector. We use the attention mechanism described in [6] to update the context vector at each time step. The attention mechanism generates a series of weights α_i corresponding to features extracted from different spatial location of the $3d$ feature representation as follows:

$$\alpha_{lj} = \frac{exp(f_{att}(a_j, h_{l-1}))}{\sum_{j=1}^{J} exp(f_{att}(a_j, h_{l-1}))}$$

where $a_j : j = 1, ..., J$ is the annotation vectors corresponding to the composed features extracted at different image locations and $l = 1, ..., L$ represent the time step. Therefore, for each location j, at time step l, α_{lj} can be interpreted as the relative importance given to location j in blending the features together.

Following the variational inference setup [10], the initial memory state and the hidden state of the LSTM are sampled from a distribution generated using the composed annotation vectors as follows:

$$p(c_0|a^{(i)}) = \mathbb{N}(\mu_c^{(i)}, \sigma_c^{2(i)})$$

and

$$p(h_0|a^{(i)}) = \mathbb{N}(\mu_h^{(i)}, \sigma_h^{2(i)})$$

for the (i)-th sample, where the corresponding μs and σ^2s are computed using a neural network that takes the annotation vector as an input. Additionally, to best harness the generative power of the architecture, we choose a prior distribution over h_0 and c_0 as a normal distributions centered at 0. Therefore the overall KL divergence loss can be described as:

$$L^h_{kld} = -\frac{1}{2}\sum_m (1 + \log(\sigma_h^2) - \mu_h^2 - \sigma_h^2)$$

and

$$L^c_{kld} = -\frac{1}{2}\sum_m (1 + \log(\sigma_c^2) - \mu_c^2 - \sigma_c^2)$$

where m is the dimensionality of the hidden state of the LSTM, which in our case is equal to the number of channels in the annotation vector.

Finally, we use a output layer to compute the output word probability given the LSTM state, the context vector and the previous word at each time step and cross entropy loss with the target caption is used to optimize the network. Concretely, the model is trained end-to-end by minimizing the following penalized negative log-likelihood:

$$L_{cap} = -log(P(y|x_i, x_t)) + \lambda \sum_j^J (1 - \sum_l^L \alpha_{lj})$$

where J represents the locations in the annotation map and L is the length of the caption. The second term in the above loss can be encouraging the model top pay equal attention to parts of image over the course of generation. Therefore, the overall loss can be formulated as:

$$L_{ovr} = L_{cap} + \lambda_{kld}(L^h_{kld} + L^c_{kld})$$

with λ_{kld} as the weighting parameter.

5. Results

The model is designed to produce humorous captions given an input image template and a set of modifier words that represent the desired theme of the meme to be generated. Figure 3 shows some novel captions generated by the model on three common meme templates. We demonstrate the ability of the model to generate multiple varied captions on the same template based on the modifiers provided. It can be observed that the model has successfully learnt the specific semantic setup-punchline format associated with the given meme templates. Additionally, it can be seen that the model is able to generate varied humorous captions on the same template conditioned on the modifier tokens provided.

Quantitative assessment of of humour is an open problem, therefore in this problem setup, traditional metrics for text generation do not provide a representative measure of the quality of the humorous captions produced. Consequently, we believe that the best method to assess the quality of the memes generated is to perform an exhaustive human evaluation.

6. Training Setup

The model was trained on 4 Nvidia GTX1080Ti GPUs with a batch size of 64. The dimension of the image representation obtained using a Resnet18 were $512 \times 7 \times 7$. For each image-caption pair 5 modifier tokens were randomly sampled from the operative part of the caption and the title of the image. These text-embedding learnt in the model has the dimension of 512, therefore the modifier text representation has the dimensions 5×512, which is aggreagted to get a representaion with 512 dimensions. The modifier text representation and the image representation extracted were then composed to give an annotation map with the dimensions $512 \times 7 \times 7$. This composed feature map is then used with the attention mechanism and fed into the LSTM decoder which has the hidden state dimension of 512. The Pytorch implementation has been made available at - https://github.com/4m4n5/reversible-meme

References

[1] Laina, Iro, Christian Rupprecht, and Nassir Navab. ”Towards Unsupervised Image Captioning with Shared Multimodal Embeddings.” Proceedings of the IEEE International Conference on Computer Vision. 2019.

[2] Guo, Longteng, et al. ”Aligning linguistic words and visual semantic units for image captioning.” Proceedings of the 27th ACM International Conference on Multimedia. 2019.

[3] Vo, Nam, et al. ”Composing text and image for image retrieval-an empirical odyssey.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.

[4] Gao, Dehong, et al. ”Fashionbert: Text and image matching with adaptive loss for cross-modal retrieval.” Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 2020.

[5] Song, Yale, and Mohammad Soleymani. ”Polysemous visual-semantic embedding for cross-modal retrieval.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.

[6] Xu, Kelvin, et al. ”Show, attend and tell: Neural image caption generation with visual attention.” International conference on machine learning. 2015.

[7] Abel L. Peirson V, and E. Meltem Tolunay. ”Dank Learning: Generating Memes Using Deep Neural Networks.” CoRR (2018).

[8] Wang, William Yang, and Miaomiao Wen. ”I can has cheezburger? a nonparanormal approach to combining textual and visual information for predicting and generating popular meme descriptions.” Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2015.

[9] Vinyals, Oriol, et al. ”Show and tell: A neural image caption generator.” Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[10] Kingma, Diederik P., and Max Welling. ”Auto-encoding variational bayes.” arXiv preprint arXiv:1312.6114 (2013).

[11] Hochreiter, Sepp, and Jürgen Schmidhuber. ”Long short-term memory.” Neural computation 9.8 (1997): 1735-1780.