Comment-Reply Generator for YouTube Videos

Eric Bianchi
Department of Civil Engineering
Virginia Tech
Blacksburg, VA 24060
`beric7@vt.edu`

Reza Sepasdar
Department of Civil Engineering
Virginia Tech
Blacksburg, VA 24060
`sepasdar@vt.edu`

Abstract

Inspired by the fake bot-scripts, our motivation was to create a model which could learn how to reply and comment to specific YouTube videos by examining the existing comment and reply data for the specific video. Our method details the process of obtaining YouTube data, preparing the data for consumption, and training the data using a deep learning natural language processing technique. We examined our model trying several parameter configurations. Our results were a mixture of success and failure. We believe that with more target data, and a stronger fine-tuned model, we could achieve a stronger performance.

1 Introduction

The inspiration behind this project came from the comedian, Keaton Patti. He claims that he 'forces' a bot to watch 1000 hours of something, like a commercial from a specific brand, and then asks the bot to write its own commercial for that brand. He called his thread *bot-scripts*. Ultimately the content that his *bot* produces is not from a bot, but from his own head. We wanted to make our own, real, bot-script for YouTube comments for a given video. Instead of forcing our bot to watch hours of video, we would force our bot to read thousands of comments and replies for a specific video with the goal of producing realistic replies and comments. The reach goal was to have the ability to alter the sentiment of generated the replies and comments.

2 Related Work

Recent advances in natural language processes (NLP) have shown success in tackling many NLP tasks (6; 2; 8; 7). The tasks that we are interested in for this research are sequence to sequence mappings for comment-reply generation, and comment generation with keyword priors. At the time of writing, the state of the art method from prior input sentences is OpenAI's GPT2 model (6). This transformer network can produce paragraphs of impressively convincing text from a prior sequence of text. Another state of the art transformer network is BERT (2), Bidirectional Encoder Representations from Transformers. A pre-trained version of the network can be fine-tuned with an additional output layer to work for NLP tasks such as question and answering. For the sentence generation given a certain sentiment, we found SentiGAN (8) to have impressive results. Using this model the authors for SentiGAN were able to alter the sentiment of a response to be negative or positive. We chose a sequence-to-sequence translation scheme that was implemented in PyTorch by (5) to translate from on language to another. The method contains two recurrent networks i.e., an encoder that inputs a sentence, and a decoder that generates the output translated sentence. We chose this framework since it was similar to the methods that we learned in the Deep Learning course. In addition to the comment-reply sequences, we believed that we would be able to also fit a sentiment score as an input parameter to our sequence-to-sequence model. This would effectively serve as the input to adjust the sentiment of a reply or comment.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

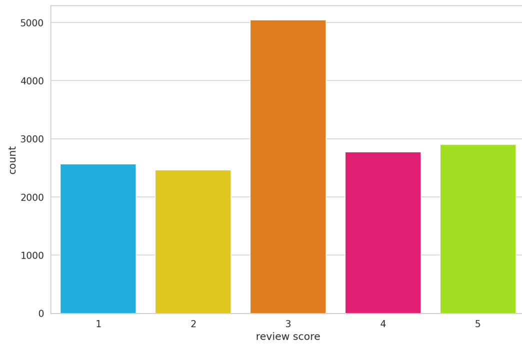


Figure 1: Unbalanced IMDB dataset

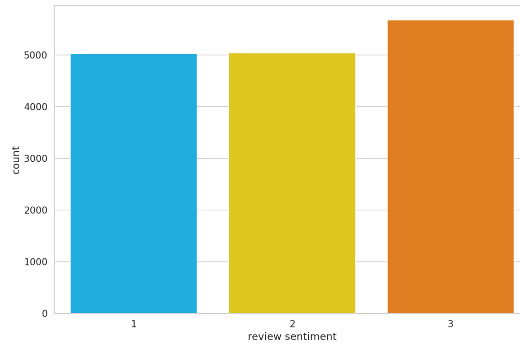


Figure 2: Balanced IMDB dataset

3 Approach

In this project a recurrent neural network is used to generate fake comments and fake replies based on training data from a YouTube video. The fake comments are generated based on two key words which are provided as the input. Hence, the network is trained using a data set containing several comments and the corresponding key words. The fake replies are generated based on a sentence (i.e., a comment) as the input. Therefore, the network is trained using a data set, comprised of several paired sentences. Each pair includes a comment and the corresponding reply. The video we chose to analyze was Chilish Gambino's *'This is America'* (3).

3.1 Sentiment Analysis

In order to generate comments or replies with a given sentiment, we first needed a way to predict a sentiment for any given comment or reply that we gathered. We went with a current state of the art Natural Language Processing method, BERT (Bidirectional Encoder Representations from Transformers) (2). BERT can be fine-tuned from a pre-trained state to perform sentiment classification by vectoring comment inputs with a sentiment label. We used the IMDB(4) dataset of movie reviews, comprised of 25,000 training and 25,000 testing samples.

The reviews were split into 5 class ratings [1-5], with 1 being the worst sentiment and 5 being the best. When viewing the IMDB dataset we found that it was imbalanced. The neutral, bin [3], had over double the number of reviews than the negative comments in bins [1,2], and positive reviews in bins [3,4]. Therefore, we combined the positive bins and combined the negative bins so that there were only three class ratings [1-3]. Once the dataset was balanced we trained the model. The model's f1-score for the test dataset was 90%, Figure [3] displays the results from the test dataset.

Even though this model appeared to have great success on the IMDB validation dataset it did not perform well on the comment and reply data gathered from YouTube. After our failure to train our own model, we were also not successful at finding an accurate pre-trained model as a work-around. Furthermore, designing a specific classifier to classify the YouTube comments based on their sentiment score was not possible since producing a sufficiently large dataset ourselves would be excessively time consuming.

3.2 Data preparation

The data was gathered from YouTube using the YouTube API (Version 3) calls. After setting up a Google Console account, we created a secure JSON-token for YouTube authentication. Once the authentication had been made, YouTube allowed our client-side to access their data. However, for a given period of time, there was a quota (10,000 units) on the number of data requests that we could make (each comment request is one unit). Once the quota was reached, then we had to wait until the next day to request more data. A flow diagram of this process may be viewed in Figure [4].

There are two types of requests that can be made to access the YouTube data, *Relevance*, and *Time*. *Relevance* is based on YouTube's algorithms of what they find relevant (number of likes, comments, or whether or not it is trending, etc.). *Time* is based on when the comment or reply being requested

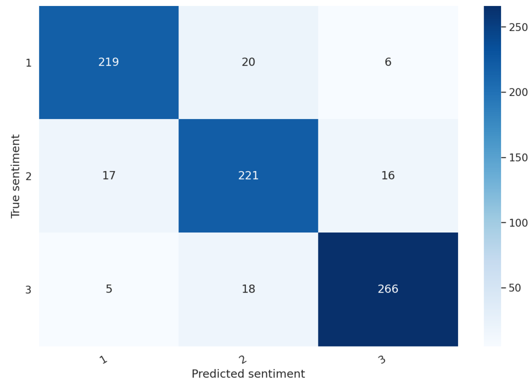


Figure 3: Model confusion matrix results. Count of predictions.

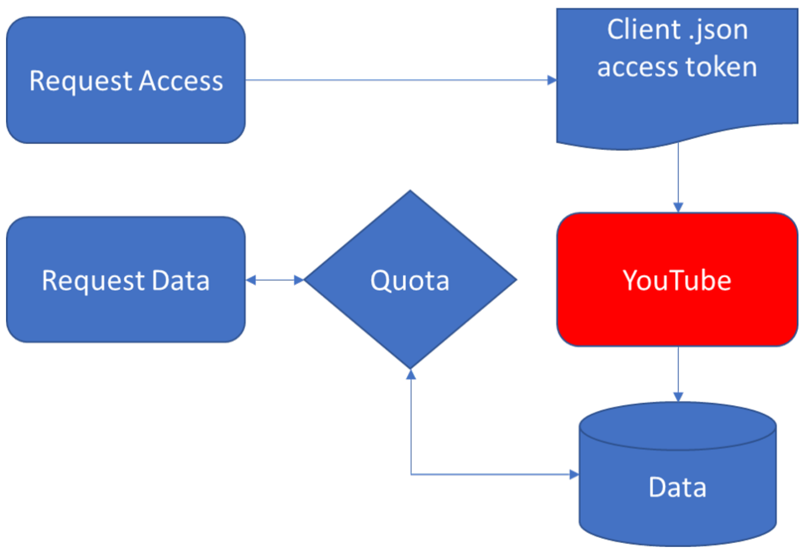


Figure 4: Flow Diagram of Requesting Access to YouTube's Data

was written. We first tried to request all our data by *Relevance*. We configured our request algorithm to request 10,000 comments. However, it would only give us back 2,000 comments, or less - if we requested under 2,000. As we found out, YouTube only stores 2,000 of the most *relevant* comments. So in order to obtain the other 8,000 comments we wanted, we had to request data based on *time*.

During the data requesting process, summarized in Figure [5], we checked if a comment contained a reply. If it did contain a reply, we then loaded in all the replies for a given comment. If the comment did not contain a reply we add that comment to our validation dataset for reply generation. Next, we selected the initial reply to the comment, since the most recent reply may not make sense. Once we had gathered the maximum number of data for the day, we ran several pre-processing steps to normalize our input comment and reply data. First we de-mojized the text or removed the emojis and replaced them with their text equivalent. Then we removed all sentences which were not in English. Next we dropped duplicate comments to ensure that we did not have severe over-fitting, as some comment threads were spam repeats. Then we converted all sentences to lowercase and added spaces between a designated list of special characters. After this, we removed all multiple spaces that were not either returns or tab-spaces. In this state the data had been regularized.

The final pre-processing step was run on this regularized dataset to obtain the first two keywords for the fake comment generation. Keywords are defined, in this paper, as any word which is not a *stop word*. A *stop word* has been referred to as a commonly used word, which search engines are programmed to ignore (1). The stop words which we used, were from the python library NLTK (1).

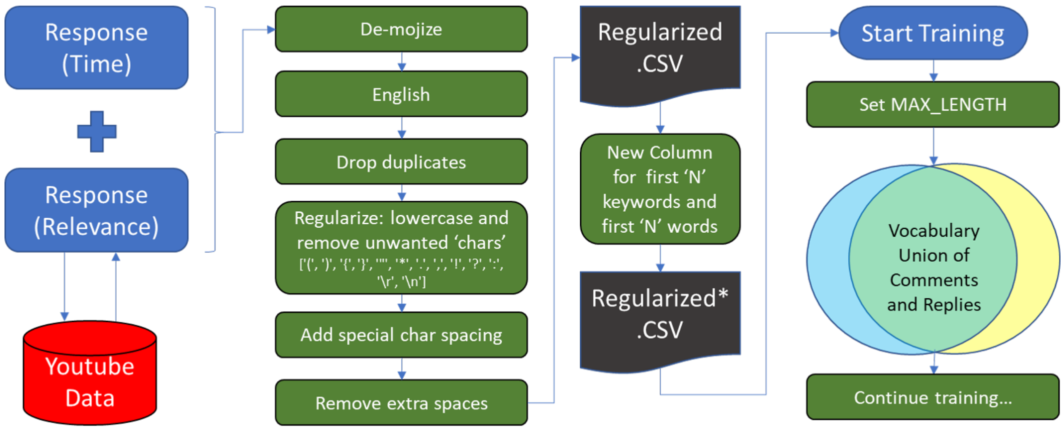


Figure 5: Flow Diagram of Annotating YouTube's Data

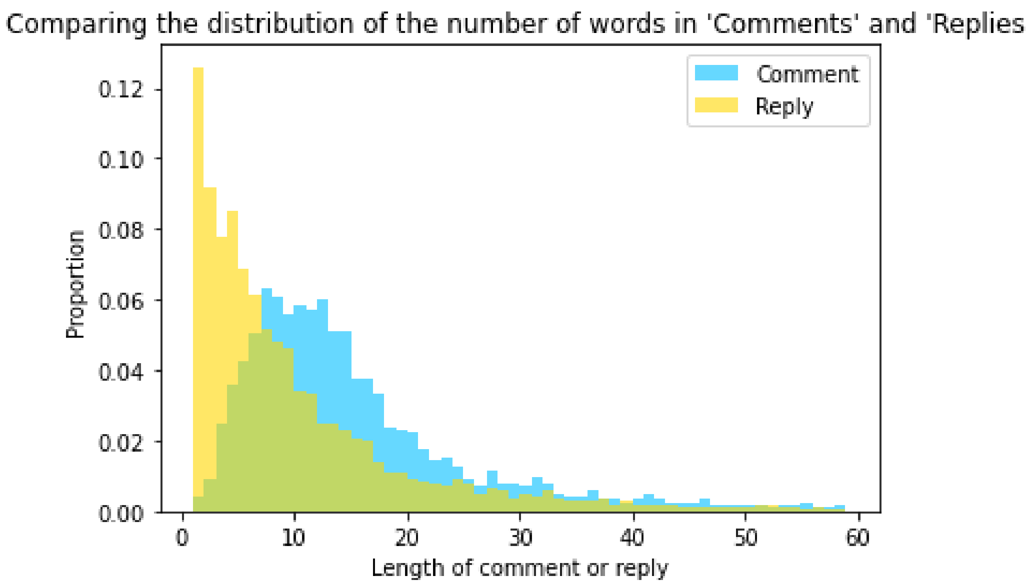


Figure 6: Histogram of comment lengths and reply lengths

We were able to visualize a histogram distribution in Figures [9, 10, 11] of the top 25 most common words found in comments and replies for our training and validation dataset. This gave us a window into what comments and replies our generative network will most like produce. The histograms include both distributions of the vocabularies with and without the stop words. It is clear, that in relation to the content, removing the stop-words show more meaningful results.

In this state, the regularized* dataset was ready for training. During training, the maximum length for the comment was set. This maximum length determine the size of the vectors to be used for the network. One clever way to pick an optimal max length is to observe the distribution of the comment/reply lengths with a histogram Figure [6]. We used this histogram as a guide for reasonable max-length parameter selections. However, adjusting the maximum length of the comment/reply, adjusts the words which are present in a vocabulary. This is an important consideration since we could be shortening the vocabularies of the replies and the comments with a shorter maximum length. After choosing a maximum length we took the union of the words in the replies vocabulary and the comments vocabulary and removed any comment-reply pair which had words outside this union (but within the maximum length).

3.3 Network architecture

We used a sequence-to-sequence translation scheme to perform the objectives of this project. The method was implemented in PyTorch by (5) to translate a sentence from one language to another. The method includes two recurrent networks i.e., an encoder and a decoder. The encoder is a recurrent neural network that inputs a sentence in a given language. It contains an embedding layer that changes the representation of the input words and converts them to vector form, followed by a Gated Recurrent Unit (GRU), as depicted in Figure 7a.

Depending on the way the information is passed from encoder to decoder, two different architectures were considered for the decoder. The two considered decoders and the manner at which they communicate with the encoder are as follows:

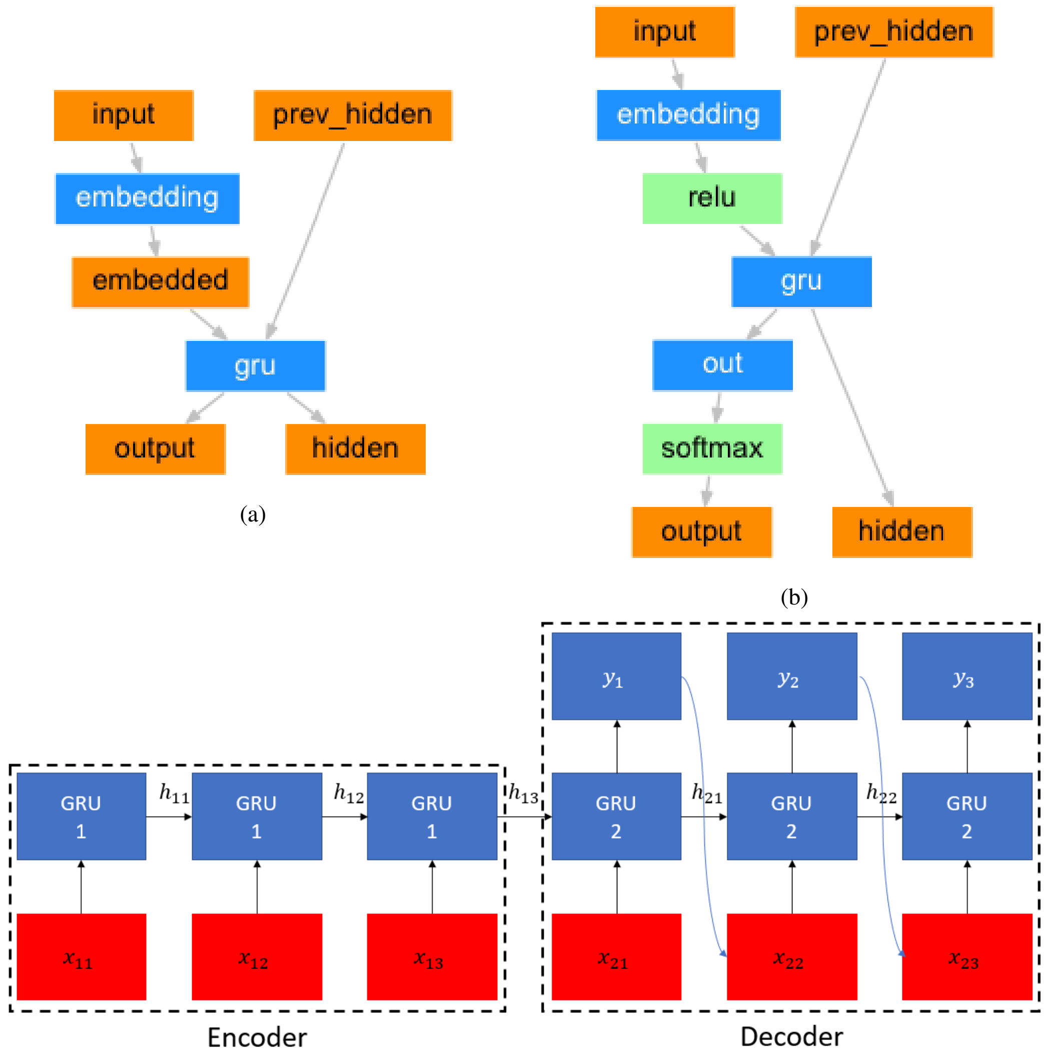
- The first decoder has an ordinary architecture, similar to that of the encoder. The only differences are the embedding layer has a Rectified Linear Unit (ReLU) activation and the GRU layer has a Softmax activation, as shown in Figure 7b. This decoder is connected to the encoder through the final output hidden state of the encoder. The final hidden state of the encoder network is called context vector, and will be the initial input hidden state of the decoder network. The overall architecture of a sequence-to-sequence translation network using the explained decoder is illus traded in Figure 7c
- The second decoder is called attention decoder and provides a better performance compared to the ordinary decoder. In the attention decoder, it is realized that encoding an entire sentence into one single context vector causes inaccuracy and loss of information. Hence, the attention decoder is designed to use different parts of the encoder outputs every time it is to generate an output. In this technique, the output of the embedding layer and hidden states of the decoder are first concatenated and passed through a hidden layer with a Softmax activation. The output is called attention weights. The attention weights are then multiplied by the encoder output vector to create a weighted combination. The output is then concatenated with the embedding output and is passed through a hidden layer with the ReLU activation. The output of this layer contains information about specific parts of the input which the decoder needs to generate an appropriate and accurate word. Therefore, the output can pass through a recurrent network followed by a Softmax activation to generate the decoders final output. Figure 8 depicts the architecture of the attention decoder. In the attention decoder, drop-outs are also considered in the embedding layer to prevent overfitting.

We used the above explained framework for the purpose of fake comment and reply generation. Instead of translating from one language to another, we translate from two key words to a comment, and from a comment to a reply.

4 Experiments

In this section, the results are presented, the framework performance is evaluated, and the studied parameters are discussed. The studied parameters include the effect of various recurrent networks (i.e., GRU, RNN, LSTM), different considered decoder architectures (i.e., ordinary and attention decoders), maximum length limit variations for the sentences, and drop-out magnitude variations. We also used the GloVe pre-trained word embedding to see if the performance can be improved. In our study, the network performance was evaluated by visual inspection of the generated sentences on test data sets. It must be noted that quantitative measures such as BLEU score is not applicable to fake text generation problems since a fake text can be anything, and there is no single sentence that can be considered as “correct" to which the generated sentences can be compared.

For both cases of fake comment generation and fake reply generation, The framework with the default values and networks proposed by (5) was used as the control model. Only the GRU model was replaced by LSTM since LSTM had a better performance as will be discussed in the next subsection. Hence, by default, the recurrent network model was LSTM, the attention decoder was used, a maximum length limit of 20 was assumed for the sentences, and a drop-out equal to 0.1 was considered. The performance was evaluated based on the epoch with the smallest training loss. The optimizer was stochastic gradient descent (SGD) with a learning rate equal to 0.01.



(c)

Figure 7: Schematic illustrating of (a) encoder network (5), (b) ordinary decoder network (5), and (c) sequence-to-sequence translation framework for the case of ordinary decoder.

It must be noted that, in general, the majority of the generated sentences were not completely flawless. The generated sentences either had grammar issues, were incomplete, or were irrelevant to the input. Generating fake sentences, in general, is a complex problem, and we believe that our data set was not large and robust enough to handle this problem. However, the quality of the training data was improved by the pre-processing stage so that the best possible performance was attained.

4.1 Effect of recurrent network model

We compared the performances of the networks with GRU, RNN, and LSTM used as the recurrent networks. In general The RNN method had the worst performance and it could barely learn the training data. The RNN for the case of fake comment generation could not be trained at all and only generated “this" for any input words. The LSTM network had an slightly better performance compared to the GRU since the generated sentences by LSTM tended to be more complete with less grammatical errors than those generated by GRU. Both LSTM and GRU networks had difficulties generating long sentences. The generated long sentences were either incomplete or full of errors. The typical generated sentences using different recurrent networks are shown in Table 1 in the Supplementary Material section.

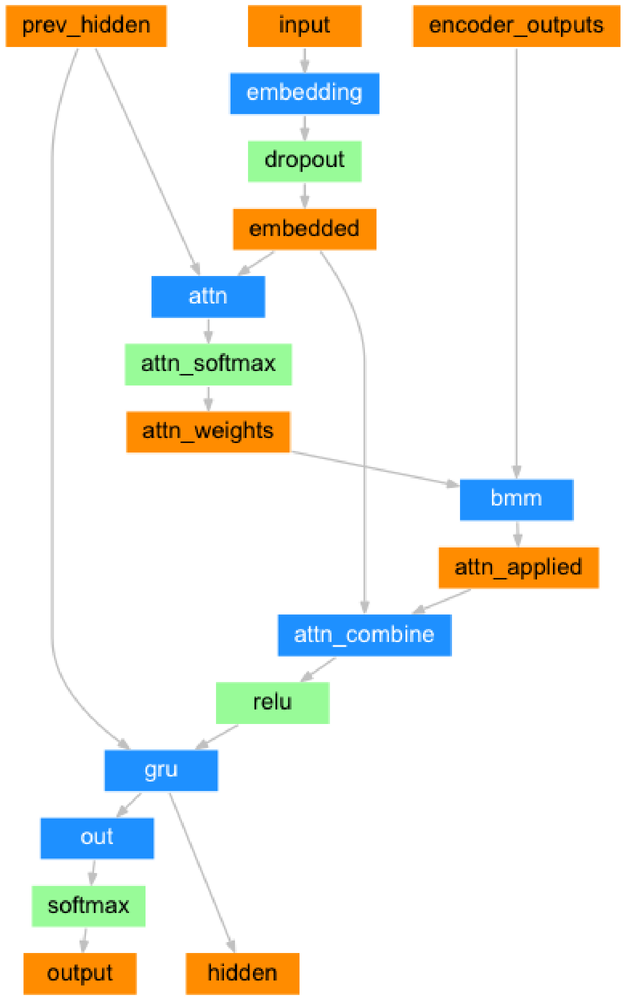


Figure 8: Schematic illustrating of the attention decoder network and the way it communicates with the encoder (5).

4.2 Effect of decoder type

The performance of the network with the attention decoder was compared to that of the ordinary decoder. For both problem of fake reply generation and fake comment generation, the attention decoder outperformed the ordinary decoder. The generated sentences with the attention decoder were longer and more diverse. The typical generated sentences using both ordinary and attention decoders are presented in Table 2 in the Supplementary Material section.

4.3 Effect of maximum length of sentences

We compared the performance of networks trained using sentences with different maximum lengths of 10, 20, and 40. It must be noted that as the maximum length limit decreased, the number of paired sentences in the training set also decreased and as a result, because the vocabulary list got shorter. For both problems of fake comment generation and fake reply generation, the accuracy decreased with the increased of maximum sentence length. The maximum length limit of 10 provided the best performance although the generated sentences were shorter compared to the larger sentence maximum length limits. Referring to Figure 6, a histogram of comment and reply lengths, it makes sense that a maximum length of 10 would do well since the two comment length distributions intersect at about 10 words. For the case of a maximum length limit of 20, the generated sentences were larger, but the number of incomplete sentences and sentences with grammar errors was also greater than the maximum length limit of 10. The maximum length limit of 40 had the worst performance and only a few generated sentences were meaningful. The typical generated sentences by the networks trained with different maximum sentence lengths are presented in Table 3 in the Supplementary Material section.

4.4 Effect of drop-out magnitude

We compared the performances of the networks trained using drop-out magnitudes of 0.1, 0.4, and 0.7. A drop-out magnitude of 0.1 provided the best performance. A large drop-out magnitude of 0.7 prevented the proper training of the network and caused the network to generate only one word regardless of the input. The typical generated sentences by the networks trained with different considered drop-out magnitudes are presented in Table 4 in the Supplementary Material section.

4.5 Effect of pre-trained word embedding

We also tried to improve the performance by using the GloVe pre-trained word embedding instead of an untrained word embedding. However, with the pre-trained word embedding, the network was not able to train well and the the performance worsened.

5 Conclusion

In this project we used a sequence-to-sequence translation scheme to generate fake comments and fake replies to comments in a popular YouTube video. The fake comments were generated based on two key words as the input whereas the fake replies were generated based on a sentence (i.e., a comment) as the input. The dataset was gathered from YouTube using their API, and prepared by cleaning and regularizing the sentences so that they could be better digested by the sequence-to-sequence training network. The sequence-to-sequence translation framework contained an encoder which is responsible to read the sentences, and a decoder which is responsible to translate the input sentence into output sentence. Based on the manner the encoder and decoder communicate, two different decoders, namely ordinary and attention decoders, were used and their performances were compared. The effect of different recurrent networks, drop-out magnitudes, maximum length of the sentences was studied. The network with the attention decoder, LSTM recurrent network, maximum sentence length of 10, drop-out of 0.1, and untrained word embedding provided the best performance for both studied problem of generating fake comments and fake replies.

Future areas of work could be to:

- Generalize a base model: training the network on many YouTube videos' data. Then use this base model to fine-tune smaller target-video subsets and vocabularies.
- Modify the objective function in order to penalize the loss if the input words are not in the generated comment.
- Explore fine-tuning the target data on transformer models like GPT2.

References

[1] E. L. Bird, Steven and E. Klein. *Natural Language Processing with Python*. O'Reilly Media Inc, 2009.

[2] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[3] C. Gambino. This is america, 2018.

[4] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[5] S. Robertson. Nlp from scratch: Translation with a sequence to sequence network and attention.

[6] I. Solaiman, M. Brundage, J. Clark, A. Askell, A. Herbert-Voss, J. Wu, A. Radford, and J. Wang. Release strategies and the social impacts of language models. *CoRR*, abs/1908.09203, 2019.

[7] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks, 2014.

[8] K. Wang and X. Wan. Sentigan: Generating sentimental texts via mixture adversarial networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4446–4452. International Joint Conferences on Artificial Intelligence Organization, 7 2018.

6 Supplementary Material

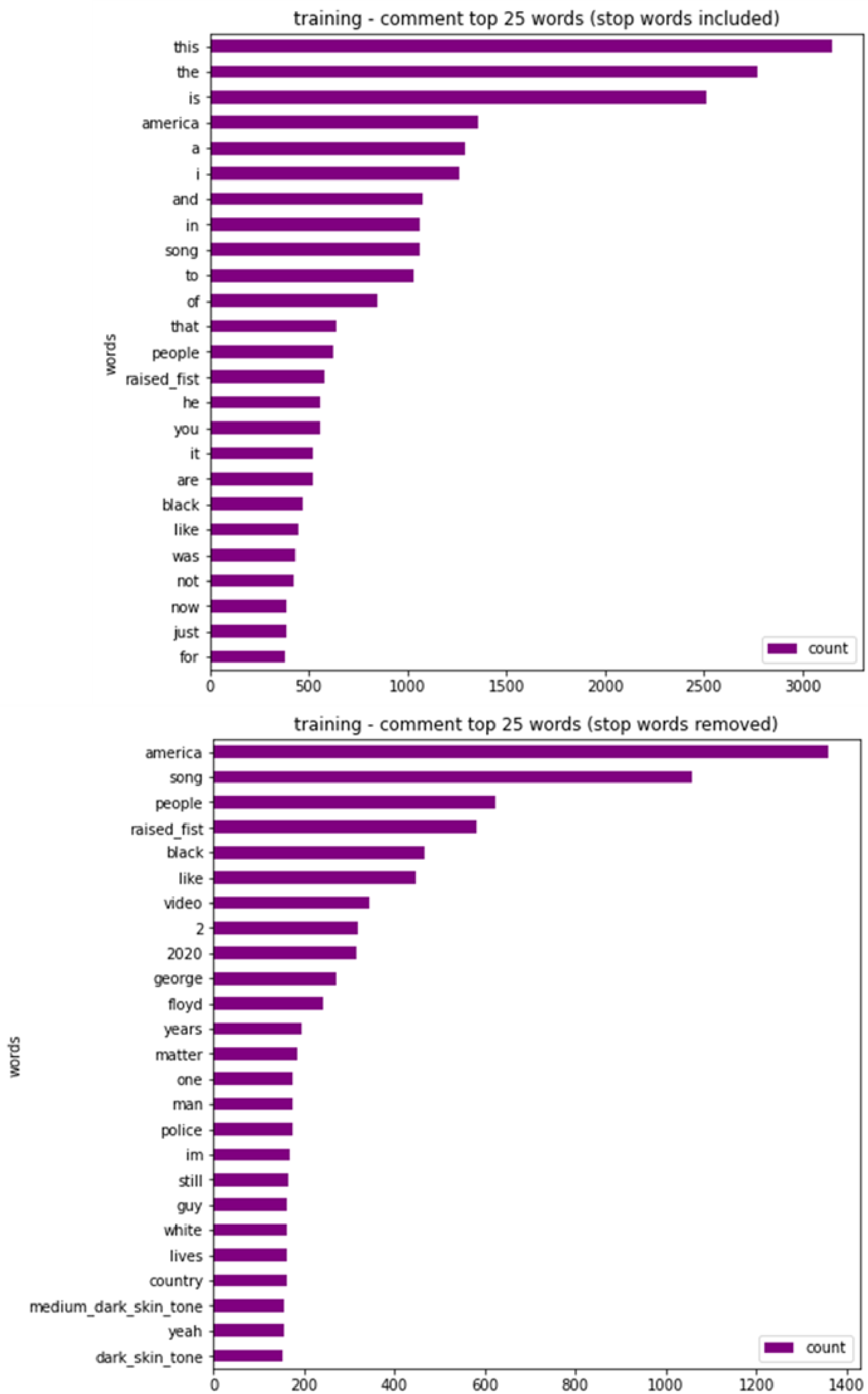
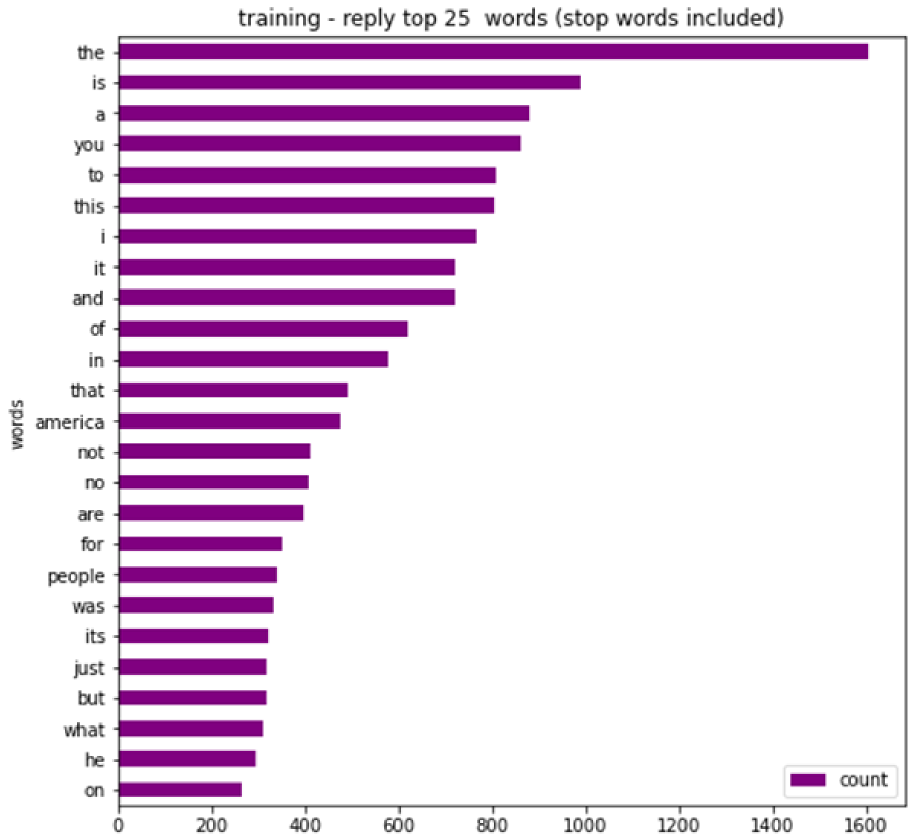


Figure 9: Training - Top 25 words found in comments. Top row are the comments with the stop words included, and the bottom row are the stop words removed.



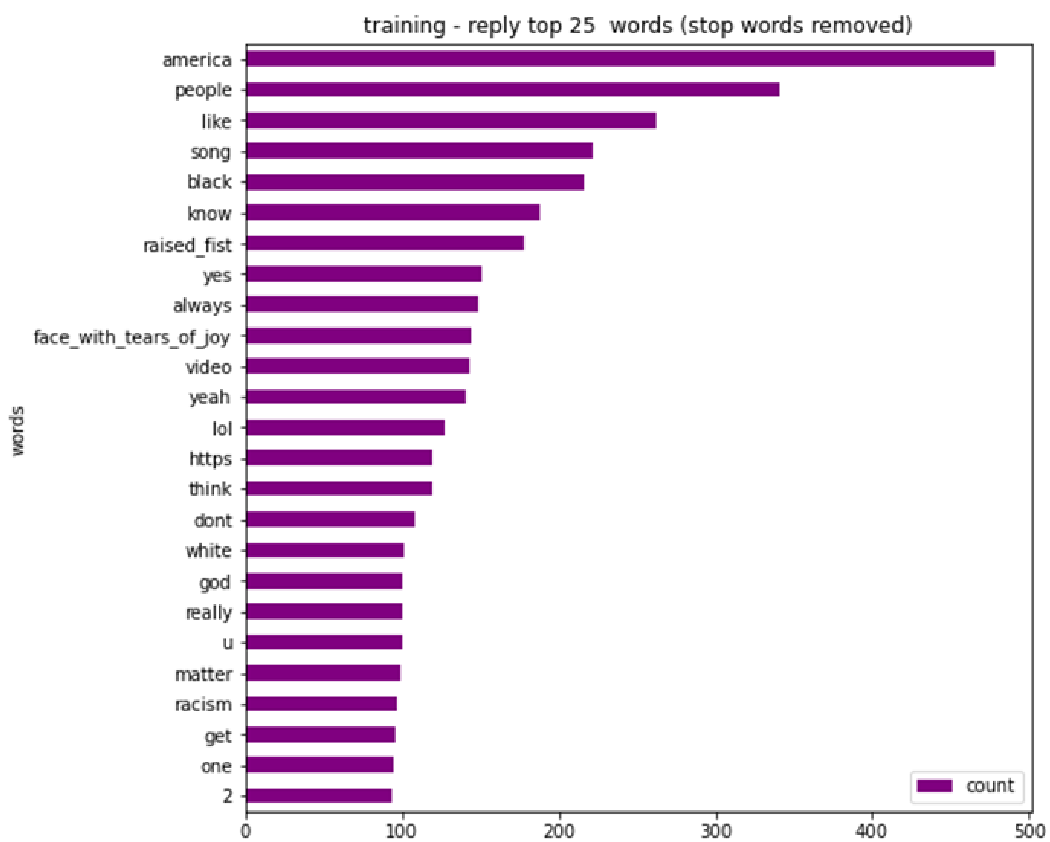
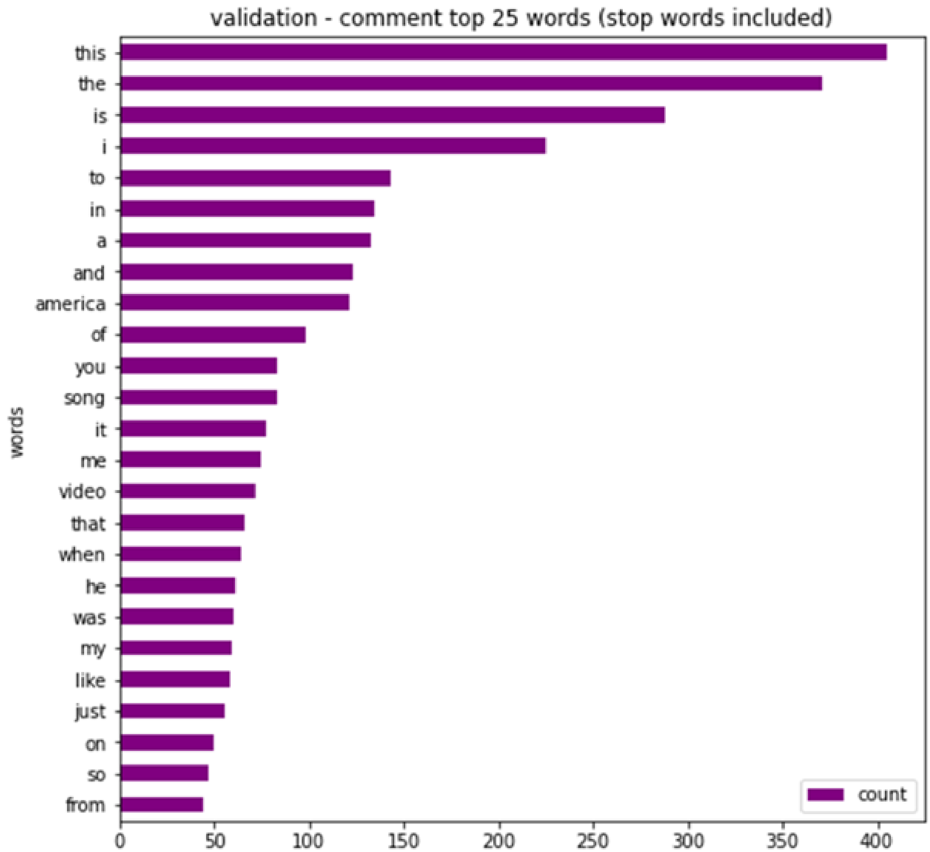


Figure 10: Training - Top 25 words found in replies. Top row are the comments with the stop words included, and the bottom row are the stop words removed.



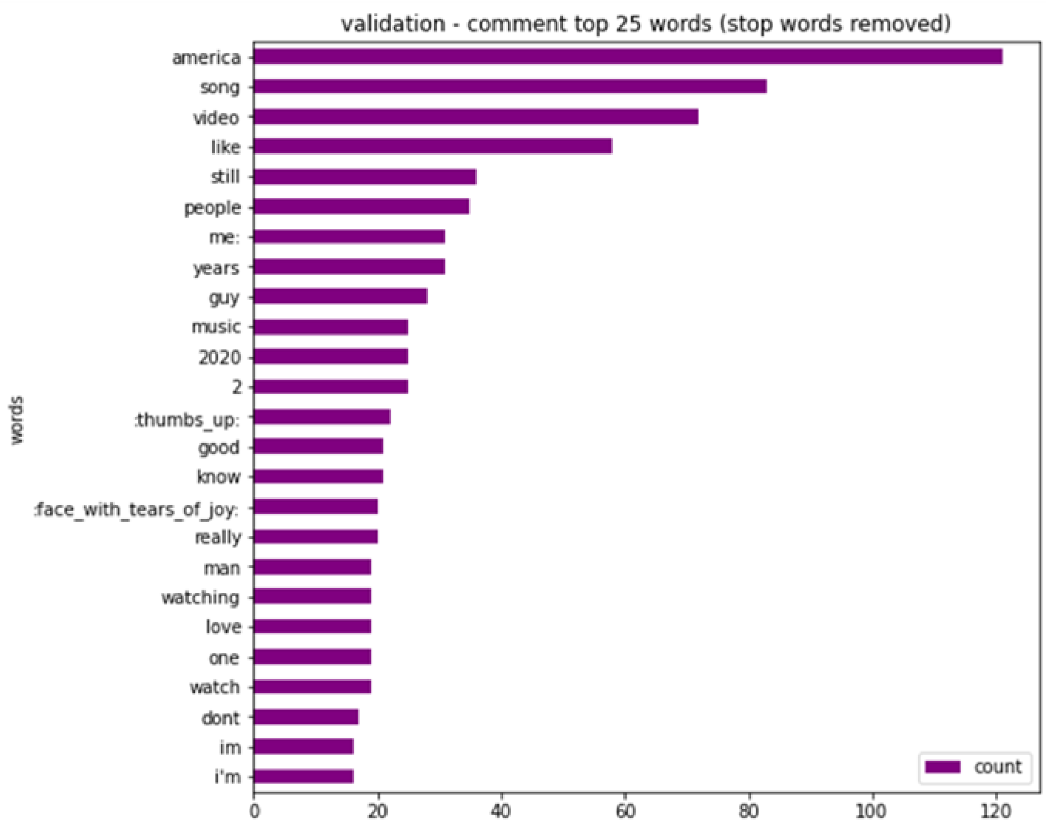


Figure 11: Validation - Top 25 words found in comments. Top row are the comments with the stop words included, and the bottom row are the stop words removed.

Table 1. Performance of the network using different recurrent networks. The sentences are typical examples.

	#	Fake reply generation			
		Good performance		Bad performance	
		Input comment	Output reply	Input comment	Output reply
GRU	1	i'm listening to this song now and what the hell	its so sad . .	imagine never watching this video not me	idk came is true but right now idk is good music video .
	2	i long for the day this gets a billion views	don't worry , it will be .	no , this is patrick ! ! ! ! *	not a , took took it really isn't .
	3	this song can be heard across minneapolis .	no , not like this	we not gon talk about how he predicted the future	no hes not the future if this song
	#	Fake comment generation			
		Good performance		Bad performance	
		Input keywords	Output comment	Input keywords	Output comment
	1	sorry 2	im sorry this is america	check video	this is video video and the video
	2	imagine song	imagine if this song was made in 2020	must taken	first i you be a lot of racism
	3	i'm surprised	i'm here because of a music video ?	song still	this song is still so
RNN	#	Fake reply generation			
		Good performance		Bad performance	
		Input comment	Output reply	Input comment	Output reply
	1	so did i miss the " ft . young thug " comment	you think this is is funny	imagine this song being released this year .	he wasn't
	2	its been almost 2 years and i still don't know what the hell i just watched	u just watched america 2020	forever will be one of my favorite songs .	he wasn't up from a with you are just more from 2020
	3	no this is patrick	you are always the same	just dance away the problems	i you think go
	#	Fake comment generation			
		Good performance		Bad performance	
		Input keywords	Output comment	Input keywords	Output comment
	1	*any two words*	this	*any two words*	this
	2	*any two words*	this	*any two words*	this
	3	*any two words*	this	*any two words*	this
LSTM	#	Fake reply generation			
		Good performance		Bad performance	
		Input comment	Output reply	Input comment	Output reply
	1	blows my mind that this was 2 whole years ago	this was always relevant .	this song is so sad cuz is true	i just to .
	2	i dont understand what i just watched	u just watched america	this song gives me the chills	and
	3	amrica is my dream	omg so edgy so deep	2020 and i still hear it	hes dont eat
	#	Fake comment generation			
		Good performance		Bad performance	
		Input keywords	Output comment	Input keywords	Output comment
	1	opinion music	this music video is so accurate in 2020	dont understand	i dont understand why this song is
	2	people background	people who actually die in the future	there's reason	we a a reason why this song is
	3	interesting song	most overrated song song ever in 2020 .	imagine canada	imagine if this was made in the ,

Table 2. Performance of the network using different decoders. The sentences are typical examples.

	#	Fake reply generation			
		Good performance		Bad performance	
		Input comment	Output reply	Input comment	Output reply
Ordinary Decoder	1	this song is genius .	this song has a meaning .	i cant believe they named a country after this song	i cant believe they named a country after this song
	2	lol i didnt know he could dance	i just came for it though .	how is this going to be 2 years old	why , what is the it
	3	the racism is a s***	it is	who is listening in 2020 ?	he didnt predict it . you . . .
	#	Fake comment generation			
		Good performance		Bad performance	
		Input keywords	Output comment	Input keywords	Output comment
	1	one comes	i am the only one who came here after george floyd ?	comment know	i know if know the song , not of .
	2	know didnt	i know he know that was 2 years ago and nothing has changed .	yup still	still still this is still
	3	perfect representation	this is a masterpiece .	idk man	i swear , man , the the that black guy
Attention Decoder	#	Fake reply generation			
		Good performance		Bad performance	
		Input comment	Output reply	Input comment	Output reply
	1	blows my mind that this was 2 whole years ago	this was always relevant .	this song is so sad cuz is true	i just to .
	2	i dont understand what i just watched	u just watched america	this song gives me the chills	and
	3	amrica is my dream	omg so edgy so deep	2020 and i still hear it	hes dont eat
	#	Fake comment generation			
		Good performance		Bad performance	
		Input keywords	Output comment	Input keywords	Output comment
	1	opinion music	this music video is so accurate in 2020	dont understand	i dont understand why this song is
	2	people background	people who actually die in the future	there's reason	we a a reason why this song is
	3	interesting song	most overrated song song ever in 2020 .	imagine canada	imagine if this was made in the ,

Table 3. Performance of the network using different maximum length limits for the sentences. The sentences are typical examples.

	#	Good performance		Bad performance	
Max sentence length limit = 10		**Fake reply generation**			
	#	Input comment	Output reply	Input comment	Output reply
	1	i dont understand what i just watched	u just watched america 2020	sadly still accurate . . .	for
	2	masterpiece of the decade	think 10 million might disagree	very much dislike this song .	your comment .
	3	this is a comment	this is a reply	very true music .	the door is open
		Fake comment generation			
	#	Input keywords	Output comment	Input keywords	Output comment
	1	let's go	let's go change the whole world !	racist virus	the prophecy doesn't our anthem .
	2	really childish	this is is a , a time-traveller	well went	well , goddam . . .
	3	dont understand	i dont understand the song	yup still	still can't is trash
Max sentence length limit = 20		**Fake reply generation**			
	#	Input comment	Output reply	Input comment	Output reply
	1	blows my mind that this was 2 whole years ago	this was always relevant .	this song is so sad cuz is true	i just to .
	2	i dont understand what i just watched	u just watched america	this song gives me the chills	and
	3	amrica is my dream	omg so edgy so deep	2020 and i still hear it	hes dont eat
		Fake comment generation			
	#	Input keywords	Output comment	Input keywords	Output comment
	1	opinion music	this music video is so accurate in 2020	dont understand	i dont understand why this song is
	2	people background	people who actually die in the future	there's reason	we a a reason why this song is
	3	interesting song	most overrated song song ever in 2020 .	imagine canada	imagine if this was made in the ,
Max sentence length limit = 40		**Fake reply generation**			
	#	Input comment	Output reply	Input comment	Output reply
	1	i thought this was europe	why ?	what in the hell did i just watch ? !	i think the people who really think he didn't .
	2	this is america right now .	this is	how is this going to be 2 years old	you you mean you have to do with this
	3	now this is real music .	its been happening for a long time	this is a movie , not a music video .	i
		Fake comment generation			
	#	Input keywords	Output comment	Input keywords	Output comment
	1	always relevant	this is so relevant relevant now	imagine song	imagine this this song song song
	2	can't believe	i can't believe this guy in 2020	sadly still	this is still the
	3	love floyd	i just came here after george floyd	got plug	i got the the

Table 4. Performance of the network using different drop-out magnitudes. The sentences are typical examples.

	#	Good performance		Bad performance	
Drop-out = 0.1		**Fake reply generation**			
	#	Input comment	Output reply	Input comment	Output reply
	1	blows my mind that this was 2 whole years ago	this was always relevant .	this song is so sad cuz is true	i just to .
	2	i dont understand what i just watched	u just watched america	this song gives me the chills	and
	3	amrica is my dream	omg so edgy so deep	2020 and i still hear it	hes dont eat
		Fake comment generation			
	#	Input keywords	Output comment	Input keywords	Output comment
	1	opinion music	this music video is so accurate in 2020	dont understand	i dont understand why this song is
	2	people background	people who actually die in the future	there's reason	we a a reason why this song is
	3	interesting song	most overrated song song ever in 2020 .	imagine canada	imagine if this was made in the ,
Drop-out = 0.4		**Fake reply generation**			
	#	Input comment	Output reply	Input comment	Output reply
	1	-	-	imagine never watching this video not me	same
	2	-	-	this is america right now .	no
	3	-	-	sadly still accurate . . .	i
		Fake comment generation			
	#	Input keywords	Output comment	Input keywords	Output comment
	1	song perfect	this song is so well	got plug	i got the the the the
	2	can't believe	i can't believe this is the same guy	video like	this video is like this video .
	3	person watching	who else is watching this in 2020	remix better	this is a
Drop-out = 0.7		**Fake reply generation**			
	#	Input comment	Output reply	Input comment	Output reply
	1	-	-	two movements to a revolution	two
	2	-	-	idk man . . i think i like the " so call me maybe " version	this
	3	-	-	this is the future .	no , not america
		Fake comment generation			
	#	Input keywords	Output comment	Input keywords	Output comment
	1	wait minute	wait wait did he said the future	dislikes stupid	all the dislikes are from
	2	love hate	i hate this song song song	let's go	let's to go the people in the
	3	anybody notice	anybody else notice about how he was released	sorry	plot a shirt