Naturalization of Speech

Richa Sharma: PES1201700662
Parth Shah: PES1201700134

UE17CS333: Natural Language Processing

ABOUT THE PROJECT

Project idea:

- This project looks at generating speech which sounds like a human by using speech disfluency or filler words like “uh” and “ums” as well as pauses inserted at appropriate places. [1][2]
- The goal of this project is to generate speech which sounds convincingly natural or is practically indistinguishable from actual human speech.
- This project also consists of comparing the results of the different models and their ability to fulfill our goal by conducting a mass survey.

UNIQUENESS AND ANALYSIS

- Some of the most widely found NLP projects focus on areas like sentiment analysis, detection or recommendation systems.
- The focus of this project is on generating speech which sounds human-like which isn't found as widely as the other topics.
- The hybrid model we have used is unlike any other project in this area of generating natural text.
- For the first time a survey form has been sent to a large group of people to get their responses as a way to test our models and capture the effect of each model.

DATASET SOURCE AND PREPROCESSING DONE

1. Dataset source:

   We used movie scripts from two sources as the data. We had about 200,000 lines of data.

   - https://nlds.soe.ucsc.edu/fc2
   - https://www.kaggle.com/Cornell-University/movie-dialog-corpus#movie_lines.tsv

2. Preprocessing:

- All basic punctuations and anything that was in brackets was removed. Punctuations such as "…" and "---" were replaced with (pause) and all 'uh', 'um' were replaced with (uh) and (um) respectively.
- All the text was converted to lower-case .

DATASET SOURCE AND PREPROCESSING DONE

This is an example:

Um … let's see...Susan's 15. Aundrea's 9. Every stupid cliche you hear about kids - they change your life they make you a better person they make you whole... It's all true!Now -- I get it.

(um) (pause) let's see (pause) susan's 15 aundrea's 9 every stupid cliche you hear about kids (pause) they change your life they make you a better person they make you whole (pause) it's all true! now (pause) i get it

LITERATURE REVIEW

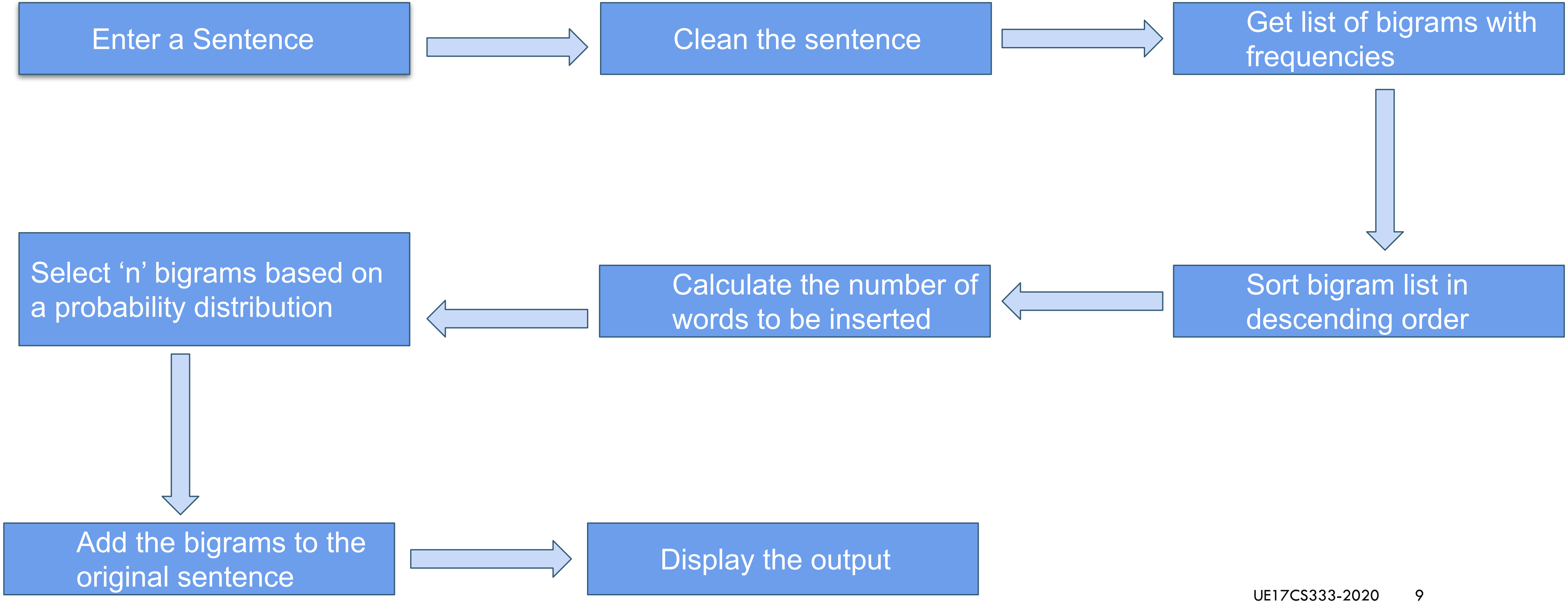
Paper Title, Year	Authors	Outcome
An Empirical Text Transformation Method For Spontaneous Speech Synthesizers, Eurospeech 2003, Geneva	Shiva Sundaram, Shrikanth Narayanan[3]	Used empirical techniques to mimic spoken language including FSAs. Results sensitive to initial data set.
Using 'uh' and 'um' in spontaneous speaking, Elsevier, 2002	Herbert H. Clarka, Jean E. Fox Tree[4]	Pauses and filler words are more likely to occur alone than together.
Neural Models of Text Normalization for Speech Applications, Computational Linguistics, 2018	Hao Zhang *et al*.[5]	Text normalization is a hard problem in English. RNNs provide high accuracy.

LITERATURE REVIEW

Paper Title, Year	Authors	Outcome
A Deep Contextual Long-Short Term Memory Model for Text Normalization, 2015	Min, Wookhee and Bradford Mott[6]	Self-generated dictionary based normalization significantly outperforms the without-dictionary model.
Text normalization using memory augmented neural networks, 2019	Subhojeet Pramanik and Aman Hussain[7]	A neural architecture that serves as a language-agnostic text normalization system while avoiding the kind of unacceptable errors made by the LSTM-based recurrent neural networks.

QUANTITY OF WORK - HIGH LEVEL BLOCK DIAGRAM OF OUR IMPLEMENTATIONS

Bigram approach



Enter a Sentence

Clean the sentence

Get list of bigrams with frequencies

Sort bigram list in descending order

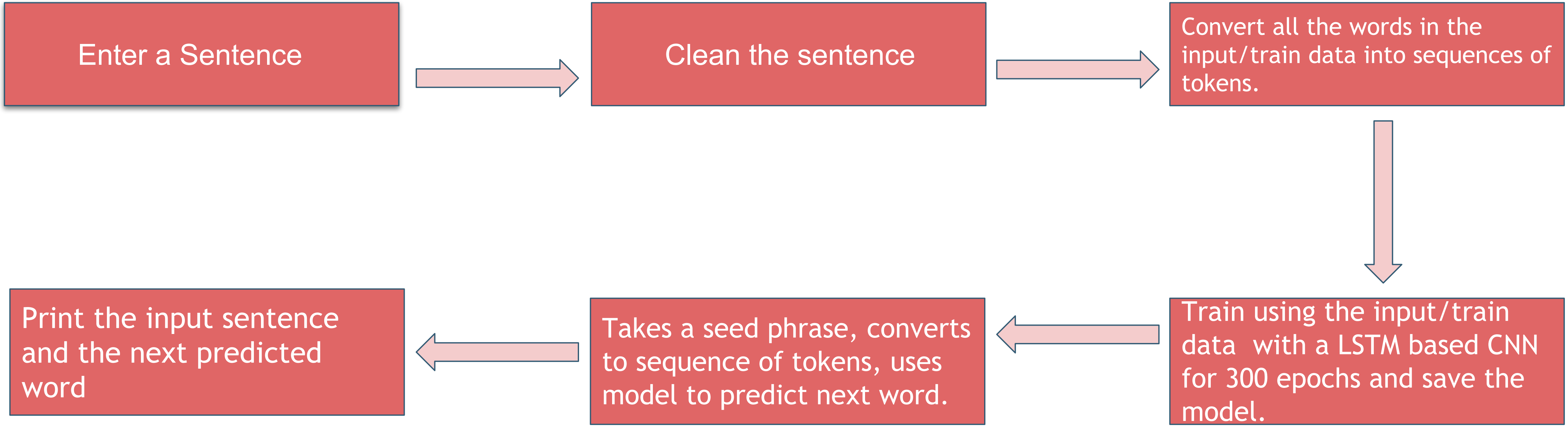
Calculate the number of words to be inserted

Select ‘n’ bigrams based on a probability distribution

Add the bigrams to the original sentence

Display the output

RNN approach



Enter a Sentence

Clean the sentence

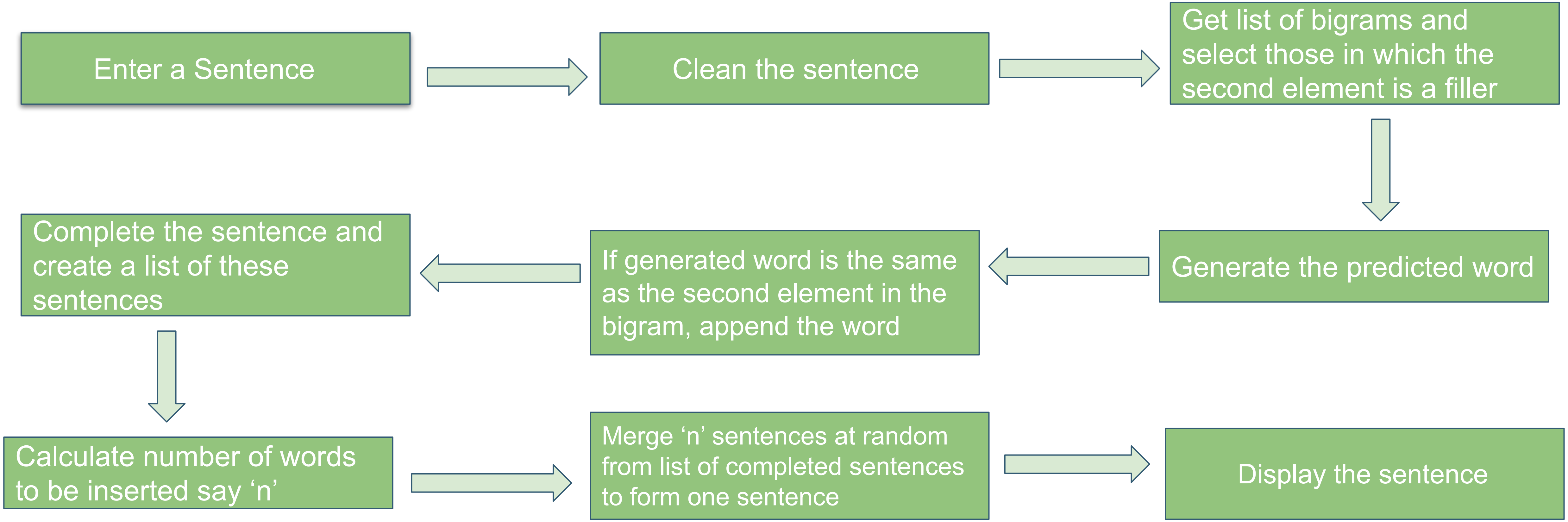
Convert all the words in the input/train data into sequences of tokens.

Train using the input/train data with a LSTM based CNN for 300 epochs and save the model.

Takes a seed phrase, converts to sequence of tokens, uses model to predict next word.

Print the input sentence and the next predicted word

Hybrid approach



Enter a Sentence

Clean the sentence

Get list of bigrams and select those in which the second element is a filler

Generate the predicted word

If generated word is the same as the second element in the bigram, append the word

Complete the sentence and create a list of these sentences

Calculate number of words to be inserted say ‘n’

Merge ‘n’ sentences at random from list of completed sentences to form one sentence

Display the sentence

QUANTITY OF WORK – THE MAIN CODE MODULES

BIGRAM APPROACH

Serial no	Code module description	Status (% complete)	What it does ?
1.	cleanInput	100	Cleans the given input sentence
2.	bigramDriver	100	Program driver
3.	possibleAlt	100	Creates the list of bigrams
4.	bigramSort	100	Sorts the bigram by descending order of frequencies
5.	createDist	100	Used for the probability distribution

RNN APPROACH

Serial no	Code module description	Status (% complete)	What it does ?
1.	loadTokenizer	100	Creates X and y where X is all two word sequences and y is the next word for all sequences.
2.	modelTol	100	Creates and saves the model in the first run and loads the model in subsequent runs.
3.	generateSeq	100	
4.	generateWord	100	

HYBRID APPROACH

Serial no	Code module description	Status (% complete)	What it does ?
1.	cleanInput	100	Cleans the given input sentence
2.	bigramDriver	100	Program driver
3.	possibleAlt	100	Creates the list of bigrams
4.	bigramSort	100	Sorts the bigram by descending order of frequencies
5.	loadTokenizer	100	Creates X and y where X is all two word sequences and y is the next word for all sequences.
6.	modelTol	100	Creates and saves the model in the first run and loads the model in subsequent runs.

HYBRID APPROACH

Serial no	Code module description	Status (% complete)	What it does ?
7.	gen_sentence	100	Calls gen_word function based on bigram words. Returns list of partially formed sentences.
8.	gen_word	100	Predicts the next word
9.	gen_entire_sentence	100	Generates the entire sentence with one filler word.
10.	gen_p	100	Checks the number of words to be inserted.
11.	indexing	100	Picks out the filler words from the list of completed sentences
12.	flatten	100	Based on number of words to be inserted, flattens out the sentences
13.	final_sent	100	Returns final sentence

QUALITY OF WORK – MILESTONES THAT ARE DONE AND WORKING

Serial no	Milestone description	Status (% complete)	Comments
1.	**bigram.py:** Generates sentences based on bigrams.	100	Gave a very good output
2.	**pos_bigrams.py:** Generates sentences based on POS tags of the words.	100	This is our fallback model incase bigram.py fails to work
3.	**flow.py:** Predicts the next word to be generated. Uses an RNN	100	Good output
4.	**hybrid.py:** Hybrid of bigram.py and flow.py. Generates the predicted word based on bigrams.	100	Good performance
5.	**pos_hybrid:** Hybrid of pos_bigram and flow.py. Generates the predicted word based on POS tags of the words.	100	Satisfactory output

RESULTS OBTAINED

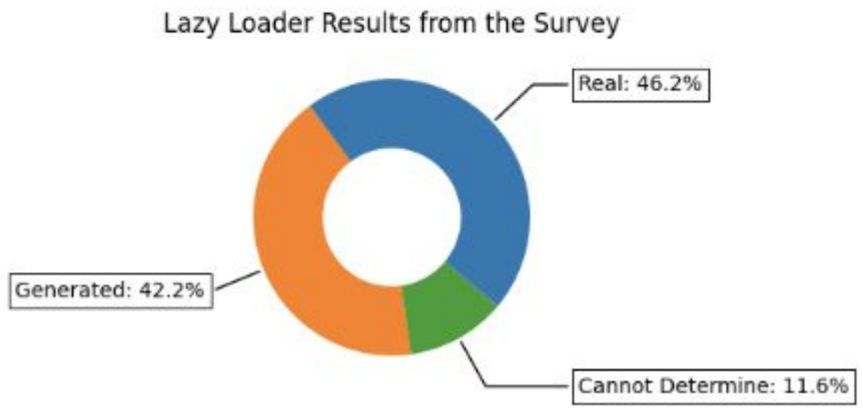


Fig. 2. Lazy Loading Results from the Survey

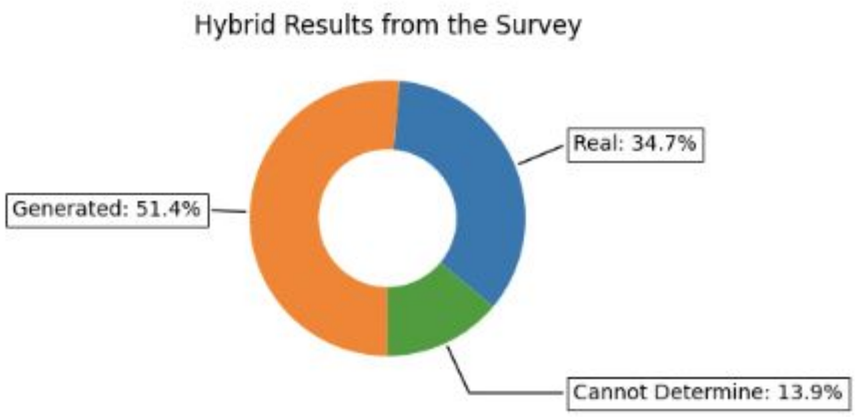


Fig. 3. Hybrid Results from the Survey

OUR TOP THREE LEARNING IN THIS PROJECT

1. Learning # 1: We learnt how to conduct an efficient literature survey and use important bits of information from them as required.
2. Learning # 2: We learnt how to work with raw datasets and how to clean them to our requirements. Without taking time to do this we wouldn't have been able to get good results. This is know an input biasing.
3. Learning # 3: We learnt how to use LSTMs and RNNs for text processing. We learnt how to tweak hyperparameters based on existing results and what they translate into.

TOP CHALLENGES UNRESOLVED SO FAR

No Issues currently but some challenges we solved:

1. Finding a dataset which fulfills our requirement.
2. Choosing an appropriate model.
3. Engineering a hybrid model.
4. Building a scientific survey with no biases.

OUR GOING FORWARD PLAN

1. Use a better dataset with diverse sources. Currently we are locked in by movie dialogues.
2. Increase accuracy of the LSTM based RNN by providing better represented input.
3. Publish this in a journal as no similar studies exist

References

[1] - https://www.youtube.com/watch?time_continue=59&v=D5VN56jQMWM&feature=emb_logo

[2] - https://ai.googleblog.com/2018/05/duplex-ai-system-for-natural-conversation.html

[3] - https://www.isca-speech.org/archive/eurospeech_2003/e03_1221.html

[4] - http://www.columbia.edu/~rmk7/HC/HC_Readings/Clark_Fox.pdf

[5] - https://research.fb.com/wp-content/uploads/2019/03/Neural-Models-of-Text-Normalization-for-Speech-Applications.pdf

References

[6] -
https://noisy-text.github.io/2015/pdf/WNUT17.pdf
[7] -
https://www.sciencedirect.com/science/article/abs/pii/S0167639318302395?via%3Dihub