Developing an Emotion Classifier and its Applications in Retrieval-Based response

Nasif Hossain

Introduction

Emotion expression and recognition plays a pivotal role in human communication. This phenomenon has garnered significant interest in the field of Natural Language Processing, inspiring a multitude of research endeavors in emotion detection, sentiment analysis, and generating emotionally articulate responses. Building on these areas, this project focuses on the development of an emotion classifier and its integration into a retrieval-based chatbot. Emotionally aware chatbots have been recognized as a key area in enhancing human-like interaction (Bostan & Dumitrescu, 2020).

Data

The project consists of two primary datasets. For the emotion classification I used the Emotions dataset for NLP [1], a public dataset consisting of 19,000 sentences labeled with six different emotions (joy ~36%, sadness ~31%, anger ~15%, fear ~13%, love ~9%, and surprise ~4%). The datasets are separated into 'train.txt', 'test.txt', and 'val.txt' with similar distribution (80-10-10 split) and each row in these files represents an utterance associated with an emotion label. The sentences in these texts mostly represent a person talking about themselves and labeled with the emotion that contains within the text (e.g “i try my best to love on them shed some light but i feel deeply compassionate with their problems and hurt even if its someone in the media;love”) . The second dataset is called EmpatheticDialogues [2] which is a large-scale multi turn empathetic dialogue dataset, containing 24,850 one-to-one open-domain conversations with 32 evenly distributed emotion labels. The second dataset is used for retrieval of responses in the model. The 32 different emotions were mapped on the 6 emotions of the first dataset. The

‘context’ column contains the emotion and the ‘utterance’ column contains the sentence in dialogues. These two columns were used for the retrieval-based response system

Methods

The emotion classification task in this project was handled by implementing a Recurrent Neural Network (RNN) with Gated Recurrent Units (GRU) using TensorFlow. The data for this task was the Emotions dataset for NLP. This dataset was loaded onto the python program as a pandas dataframe. With two columns ‘Text’ and ‘Emotions’, the contents of the ‘Text’ column was put through different methods of preprocessing.

Preprocessing

First, punctuations in the texts were removed and all the characters were made lowercase. Then a tokenizer was initialized using the Tensorfow.keras.preprocessing.text library. Tokenization was especially important for this task as it gives us a way to split the text into smaller meaningful ‘tokens’ which allows the model to understand the text at a low level. It also allows us to conduct vectorization in the text and make it a suitable input for the RNN model. Each text in the dataset is then transformed into a sequence of integers, where each unique word in the dataset is mapped to an unique integer by the tokenizer. It is essential for the model because neural networks do not accept raw text data and take in numerical data. After each text is transformed into a sequence of integers, padding is used because the sequences vary in length and are not all the same length. Since neural networks require input data to be the same shape, I had to pad the sequences to make all the sequences the same length. The max length of a sequence is 63 and all the other sequences are also converted to length 63 for the RNN input. The labels (Emotions) were converted to integers using the sklearn.preprocessing library. Since the labels (y_value for the model) were in textual form, I converted it into numerical integers. These labels were then converted into a binary matrix representation for multi-class

classification using one-hot encoding, 1 represents the presence of the label and 0 represents absence.

Architecture

The model used for this project is a Gated Recurrent Unit (GRU) model which is a RNN. It includes an embedding layer which uses pre-trained GloVE word embeddings [3] (Pennington et al., 2014). It transforms the integer-encoded vocabulary into a dense vector representation. The ‘glove-wiki-gigaword-300’ model was used which is trained on a corpus of 6 billion tokens from wikipedia 2014 and Gigaword5. The embeddings have 300 dimensions which provide a rich representation of words by capturing different semantic properties. After the input and embedding layer, a GRU layer with 128 units was implemented which serves as the core component of the model’s architecture. A dropout rate of 30% was set for the GRU layer so that a percentage of neurons in the layer are randomly deactivated at each time step. This was done to prevent overfitting by ensuring that the model does not rely too much on a single neuron and forces it to learn more robust features from the data. The ‘return_squences’ parameter was set to false so that the GRU layer only returns the last output for each input sequence rather than a full sequence of outputs. For the classification task, the model should get a single label for the entire output sequence and not a label for each word. The GRU layer also includes an L2 activity regularizer to prevent overfitting by adding a penalty equal to the square of the magnitude of weights, keeping the weights as small as possible and reducing the model’s complexity. This GRU layer allows the model to understand the sequential nature of the text data and capture temporal dependencies. This GRU layer is then connected to a Dense output layer with 6 units for the 6 emotion categories to predict from. Using a softmax activation function, the dense layer will convert these scores into probabilities and output the emotion category with the highest probability. The use of the softmax function makes this model suitable for multiclass classification. Similar to the GRU layer, an L2 activity regularizer is also applied to

the Dense layer. The Dense layer serves as the final decision maker that takes the features learned by the GRU layer to make the final emotion classification. The learning rate was set to 0.01 and the Adam optimizer was used to update the network weights iteratively based on the training data. The loss function used is called the ‘categorical_crossentropy’ which is appropriate for this multi-class classification problem and works well since our targets are one-hot encoded.

Retrieval-based responses

The EmpatheticDialogues dataset was loaded and the 32 emotions in the dataset was mapped into the 6 emotions in the first dataset. The mapping is as follows,

Fear: afraid, terrified, anxious, apprehensive

Sadness: sad, embarrassed, lonely, ashamed, guilty, disappointed, devastated

Anger: angry, jealous, annoyed, furious, disgusted

Joy: proud, joyful, grateful, excited, confident, nostalgic, caring, trusting, hopeful, content

Love: sentimental, faithful, nostalgic, caring, trusting

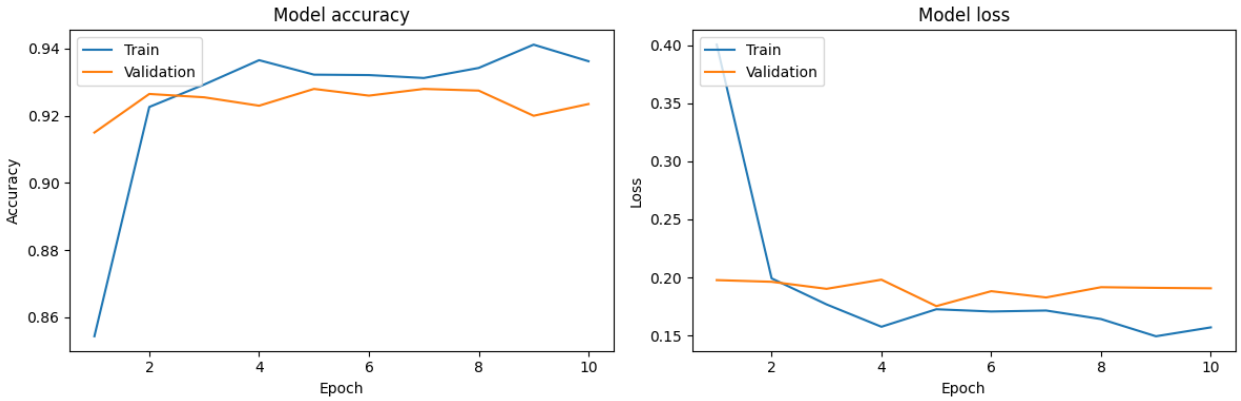
Surprise: prepared, surprised, anticipating, impressed

The input statement is classified into one of six emotions using a previously trained model, and a corresponding response is retrieved from a database based on its contextual similarity to the input. The system utilizes a bag-of-words model, implemented through the CountVectorizer from sklearn library, to convert the text data into a numerical representation suitable for similarity measurement. Upon receiving a user input, it first undergoes the same preprocessing steps as the database text. The emotion of the user input is classified using the RNN model described above. The system then grabs all utterances associated with the same emotion, and selects it from the database and transforms them into vectors. The cosine similarity is then calculated between these vectors and the user input vector. Cosine similarity measures the cosine of the angle between two vectors, which is a measure of their directional similarity, disregarding their

magnitude. The response with the highest cosine similarity to the input is retrieved and the next utterance is returned as the system's output for a demonstration of a human-like conversation dialogue.

Results

The classification model was trained for 10 epochs and we can see below that the loss is minimized overtime and the accuracy increases overtime. It's worth noting that the validation loss slightly increases after the 5th epoch while the accuracy doesn't show significant improvement after the 2nd epoch. This could imply that the model started to overfit the training data after a certain point. Techniques like early stopping could be implemented to prevent such overfitting.



Accuracy: 0.916, Recall: 0.916, Precision: 0.916

Classification Report:

	precision	recall	f1-score	support
anger	0.92	0.92	0.92	275
fear	0.87	0.92	0.89	224
joy	0.93	0.93	0.93	695
love	0.79	0.80	0.80	159
sadness	0.95	0.96	0.96	581

surprise	0.86	0.65	0.74	66
macro avg	0.89	0.86	0.87	2000
weighted avg	0.92	0.92	0.92	2000

The model has an accuracy, recall and precision score of 91.6% which indicates a high rate of correct predictions and the recall and precision represents that its correctly detecting correct classes and not raising false-negatives, In the classification report, we can see an f1-scre of over 0.74 for all classes which tells us that the model is reliable over different emotions. The highest f1-score is for sadness (0.96) and lowest for surprise (0.74) which means that the model is very good and detecting ‘sadness’ and needs to be improved for detecting ‘surprise’. This could be the case because of the lack of data in the ‘surprise’ category compared to the size of other categories. The model was then given some sample sentences to predict the emotion it depicts and it correctly predicts the corresponding emotion, which supports its practical use in identifying emotions from text. The testing of the model tells us that the model exhibits robust performance in this emotion classification task.

The retrieval-based system could not be tested through any metrics other than human understanding. However, it was given some sentences to respond to and it managed to output coherent sentences and responses with human-like conversation capabilities.

Input: "I lost my pen today" | Output: (input emotion: sad) “thats a bummer how long ago did you lose it”

Despite retrieving responses said in a different dialogue, the system still manages to find responses to general and common inputs.

Conclusion

This project presented the development and integration of an emotion classifier into a retrieval-based response system. The emotion classification task, using a GRU model, gave a good evaluation score with 91.6% accuracy. As for the response system, it effectively produced coherent, contextually appropriate responses to general and specific input prompts which demonstrates its capability for human-like conversation if refined further. The system may benefit from an even larger and more balanced dataset. It could have a more complex model architecture and a more refined retrieval system. The retrieval-based response system could be better evaluated through user satisfaction surveys or other forms of user scoring. A reinforcement learning method could be implemented through this for further improvement of the system. Despite these areas for improvement, the system effectively leverages the power of emotion detection and retrieval-based responses to enhance the capabilities of chatbots, providing a more human-like, emotionally aware conversational experience.

Works Cited

1. Emotions dataset for NLP, Praveen
2. Empathetic Dialogues, Rashkin
3. Pennington, J., Socher, R., & Manning, C. (2014). GloVe: Global Vectors for Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)