Dialog Act Classification for Speech Assistants

Abhishek Kanike (abkanike@iu.edu)
Manoj Joshi (manjoshi@iu.edu)
Preetham Kowshik (pkowshik@iu.edu)

Abstract

Over the years, experiments on Dialog Act classification was based on models such as Hidden Markov Models, maximum entropy, conditional random fields, graphical models, and support vector machines. These models captured contextual information of a given utterance. Recent experiments used deep learning techniques such as Convolutional Neural Networks and Recurrent Neural Networks to classify dialog acts. However these models don't consider prosodic features such as intonation, pitch and stress in order to classify dialog act. In this paper we proposed a method which extracts both contextual as well as prosodic features. These features are then fed into a Convolution Neural Network which classifies each utterances. We initially classify the utterances into statements and question where each question is further classified as YES/NO, Wh and declarative questions. Statements are further classified as view/opinion or descriptive.

Introduction

Identifying the motive of users in dialogue system is important. Till date, considerable number of methods were developed to recognize user's intention in dialog system. User's intention can be deduced in an utterance, by categorizing utterance in finite classes as dialog act labels. Dialog act (DA) represents a function of a speaker's utterance in either human-to-human or human-to-computer conversations. DA theory asserts that a sentence or spoken utterance can be separated into two components, the Propositional Content (i.e. what is it about) and the DA (i.e. what is it saying about the propositional content). For example, the propositional content "door is shut" can be in a question DA "Is the door shut?", an instruction DA "Shut the door!" or an assertion DA "The door is shut.", which mean quite different things. Developing techniques to identify and classify dialog acts automatically is challenging task.

In past, many studies explored different approaches for dialog act modelling. These approaches uses contextual information to classify the current utterance. For instance, if the previous utterance was a question, then there is high chance that response would be be an answer to the question utterance. Model such as Hidden Markov models (HMMs), conditional random fields (CRF) and dynamic Bayesian Networks (DBN).

With recent success in deep learning techniques such as Convolutional Neural Networks, Recurrent Neural Networks in natural language processing tasks viz. document classification, machine translation, we also try to use neural networks for dialog act classification. Since these methods doesn't use speech signal features such as intonation, pitch, stress and volume we try to employ these prosodic features along with contextual information. For simplicity we divided our classification task into two 1) stress word detection 2) statement and question type classification. In statement sentences we considered two classes viz, view/opinion and descriptive type statements and in question sentences we considered five classes namely, YES/NO, Wh (When, what, where) and declarative questions.

In the following section we state the related work in dialog act classification. Next section is about the data we used for our model. Following section discusses about the model by 1) introducing the engineering work done on the prosodic features and contextual features 2) two classification i.e, stress word detection and sentence type. The next section lists the results we achieved using our model. Last section discuss the future work on the approach.

Background

Traditional approaches in dialog act classification models were built on machine learning algorithms such as Maximum entropy, DBN, HMM, and SVM. These models explored features such as lexical, syntactic features, prosodic cues, and speaker interactions. Specifically these included contextual information. Early studies used HMMs where the hidden states are the DA tags, which generate the sequence of words as observations. The observation probabilities are obtained by DA specific word based language models, and a DA tag based n-gram language model provides the transition probabilities between the DA tags. Conditional Random Features(CRF) was used widely for extracting contextual features. Among these, those which used more sophisticated techniques for extracting lexical information, achieved the best results before deep learning was applied to the problem. During early 2000's, researchers tried to extract the prosodic features. Shriberg et al. (2000) was one of the first works that explored the prosody as a potential knowledge source for dialog-act classification. Few researchers tried to add these prosodic features to lexical probabilities at surface level.Novielli and Strapparava (2013) investigated the effect of sentiments on DA classification.

Recent experiments have explored Deep learning techniques for Dialog Act classification. Both Contextual and Sequential features were extracted using CNN and RNN. Kalchbrenner and Blunsom (2013) used a mixture of Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). CNNs were used to extract local features from each utterance and RNNs were used to create a big picture view of the utterance. Rojas-Barahona et.al used a mixture of LSTM and CNN for DA classification and slot filling dialogue. Ji et al. (2016) built a hybrid model that

combines an RNN language model with a discourse structure that considers relations between two continues utterances as a latent variable. This approach increased the accuracy by 3% when applied on the Switchboard corpus.

Data

A corpus consisting of 88 audio samples were manually recorded and labelled. We classify each audio samples into following classes;

1. Descriptive Statement - These are simple utterances which does not contain any intonation contour or any stress word. Examples such as “A man bought a car ”, “Look she is pretty” are labelled as descriptive statements.

2. View/ opinion Statements - These are utterance s where the speaker is trying to given his stance on a given topic. Examples includes “I feel whatever she is doing is correct”, “I am of the opinion that he will become a billionaire one day”. These utterances have a partial falling intonation on the focus word.

3. Declarative Questions - these utterances are used to confirm or verify an information which is already known to the speaker. Examples such as “You bought what?”, “Are you sure?”. They have a rising intonation on the focus word/stress word.

4. Wh Questions - these utterance are basically “What, Where, When, How” questions. Examples include “Where are you from?”. “What are you doing?” and so on. These utterances have a falling or rising intonation on the focus word.

5. Yes/No Questions - these utternace are questions which get yes or no responses. Example include “Did you eat that donut?”, “Do you need a snack? ” and so on. These utterance have rising intonation on the focus word.

In order to have our model to be invariant of voices of different speakers. Same utterances are recorded with there different voices. The MFCC features should take care of speaker dependent features.

Model

Feature Extraction

Mel Frequency Cepstral Coefficients (MFCCs) - The most widely used feature extraction technique in Automatic Speech Recognition(ASR) is the Mel Frequency Cepstral Coefficient. It is based on human misconception of word. Inorder to extract features from audio sample MFCC tries to incorporate both human speech production and speech perception. MFCC subsumes the logarithmic perception of loudness and pitch of human auditory system while it eliminates speaker dependent characteristic by removing the fundamental frequency and their harmonics.

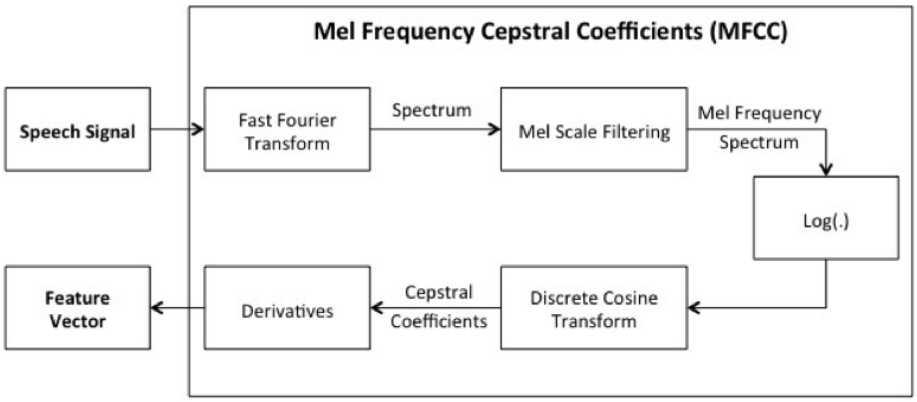


Figure 1 MFCC Feature Architecture

Given a speech signal following transformations are applied to extract the power spectrum features of the signal.

1. *Fast Fourier Transform* - The first step in this process is to calculate the frequency domain representation of the input signal.

$$c_{\tau,k}^{(1)} = \left| \frac{1}{N} \sum_{j=0}^{N-1} f_j \; exp \left[-i \; 2\pi \; \frac{jk}{N} \right] \right| \qquad k = 0, 1, ..., (N/2) - 1$$

where N = number of sampling points within a time frame t.

2. *Mel Scale Filtering* - The second step calculates the mel frequency spectrum. A spectrum is passed through a number of band pass filters and the power of each frequency band is calculated. This represents the human auditory system which uses the power over a frequency band as signal for further processing.

$$c_{\tau,j}^{(2)} = \sum_{k=0}^{N/2-1} d_{j,k}\, c_{\tau,k}^{(1)} \qquad j = 0, 1, ..., N_d$$

where d is the amplitude of the band-pass filter with the index j at the frequency k and N_d equidistant band-pass filters on the mel scaled.

3. *Logarithm* - This step takes the log of the signal which represents human perception of loudness . Research has shown that human auditory system picks up loudness on a log scale.

$$c_{\tau,j}^{(3)} = log(\, c_{\tau,j}^{(2)}\,) \qquad j = 0, 1, ..., N_d$$

4. *Cepstral coefficient* This step is used to eliminate the speaker dependent characteristics by computing the cepstral coefficients. Higher order cepstral coefficient show the frequency spectrum of the source signal while the lower order cepstral coefficient shows the he frequency response of the vocal tract. By taking only the lower order cepstral coefficient for further processing we can remove the speaker dependent characteristics.

5. *Derivatives* - In Order to capture the dynamicity of the speech signal, the first and second order derivatives of the cepstral coefficients are also added to the feature vector.

Inorder to extract MFCC features we use the librosa mfcc api which takes in the speech signal as the input along with the sampling rate and a numpy array representing log-power Mel spectrogram. It returns a numpy array consisting of MFCC features for the input speech signal.

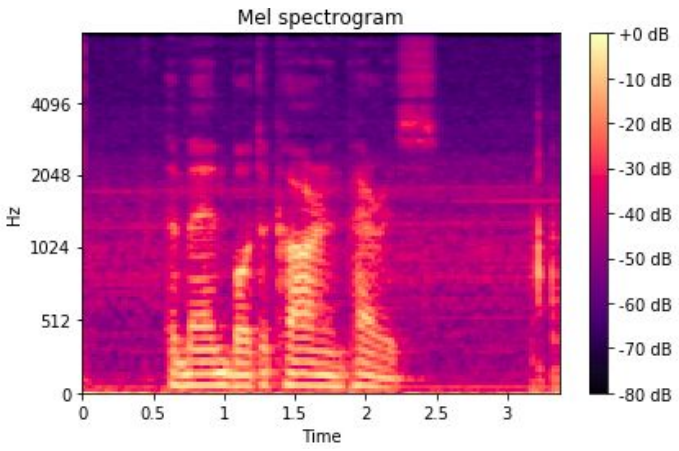


Figure 2 Mel Spectrogram of a sample audio file

Root mean square Energy (RMSE)- Root mean square energy approximates as a prosodic feature since it directly relates to intensity of the sound wave. Intensity of sound wave is defined as average amount of energy emitted over single unit of time for a specific unit area in a particular direction.

Speech signal is usually represented in the form of amplitude wave signal such as sinusoidal wave. Amplitude measures the displacement of a medium from its resting state. Energy of the signal is computed from amplitude using the following equation over a frame length equal to *n*.

$$E = \sum_{n} |x(n)|^2$$

If above formula is used to calculate for example energy in a standard sinusoidal wave, it would equate out to zero since the amplitude fluctuates between positive and negative value of equal magnitude. As a result this doesn't capture real information of intensity (using amplitudes), since low-amplitude and high-amplitude sine waves appear a lot in speech signals. Moreover intensity depends on the average energy rather than just sum of amplitude square. An appropriate and meaningful reference to measure the average amplitude of a wave over time: root mean square energy (RMSE). The root mean square energy represents a squaring of instantaneous amplitude for each point in time (or a sufficient number of equally spaced points) of a waveform, calculating the average (arithmetic mean) of those values, and finally taking the square root of this average. Following is an equation to calculate RMSE in a signal over *N* frames.

$$E = \sqrt{\frac{1}{N} \sum_{n} |x(n)|^2}$$

We extracted RMSE features from libros's rmse api which accepts speech signal as input with an optional sampling rate (default value = 22050) and frame length (default value = 2048). API returns a single dimensional numpy array with each value representing root mean square energy value over the default or specified frame length.

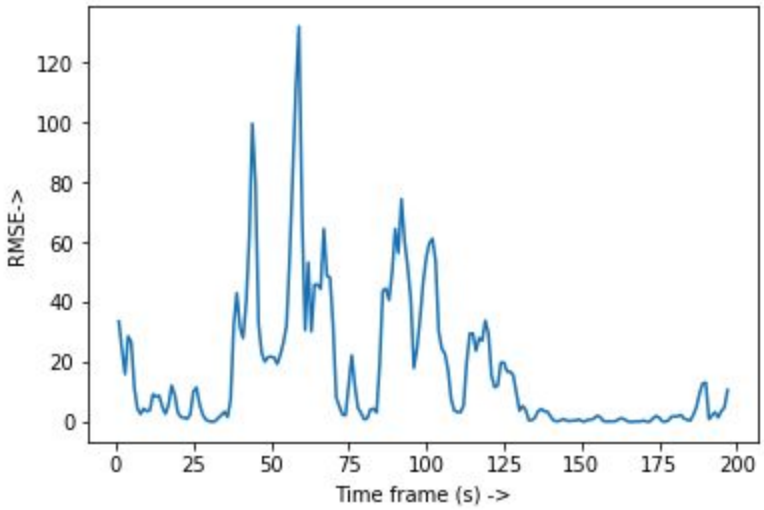


Figure 3 RMS value plotted against time for an audio sample

Stress word Detection - Random Forest classifier is used as a model for stress word detection. In Random forests, multiple random decision trees are created using only a small random subset of the features. Samples of the training dataset are taken with replacement, but the trees are constructed in a way that reduces the correlation between individual classifiers. Specifically, rather than greedily choosing the best split point in the construction of the tree, only a random subset of features are considered for each split. Using Random Forests also removes the necessity of pruning trees which would otherwise be required to prevent overfitting of the model to the training dataset.

The parameters for the model used are:

1. n estimators = 5. This refers to the number of decision trees in the forest. Increasing this parameter increases the accuracy of prediction as it reduces the impact of over-fitting.
2. max features = 200. This determines how many features each tree is randomly assigned. The smaller, the less likely to overfit, but too small will start to introduce under fitting.
3. max_depth = 7. This will reduce the complexity of the learned models, lowering over fitting risk.

Prosodic rmse feature was used to detect the presence of stress word in a sentence. A 2-dimensional vector of rmse feature of all audio samples is extracted and given to the above Random Forest model as input vector. RMSE feature vector of each audio sample is restricted to a fixed length of 200 since length of rmse depends on the duration of speech.

Stress word Identification - As part of motivation to future work, stress word identification was done using Google Cloud Speech API. Google Cloud Speech API is a powerful speech recognition tool that enables developers to convert audio to text by applying powerful neural network models in an easy to use API. Google Cloud Speech API performs highly accurate speech-to-text transcription in near-real-time. Each audio sample in wav form is fed to API which returns following data in JSON format.

1. Transcript: An english text converted sentence of the input speech sample.
2. Confidence: With what confidence did the google Engine return the transcript in range of 0 (Low probable transcript) to 1 (High probable transcript).
3. Words: A list of words contained in the transcript with each word containing following information:
    a. Start_time: The exact start_time of the word in the audio sample.
    b. End_time: The end_time of the word in the audio sample.
    c. Word: The actual word in above time interval.

After obtaining these timing information, rmse vector for each word is extracted from the audio file. A maximum value over the rmse vector of each word is considered and the maximum over all these maximum rmse value is identified as the stress word.

Architecture

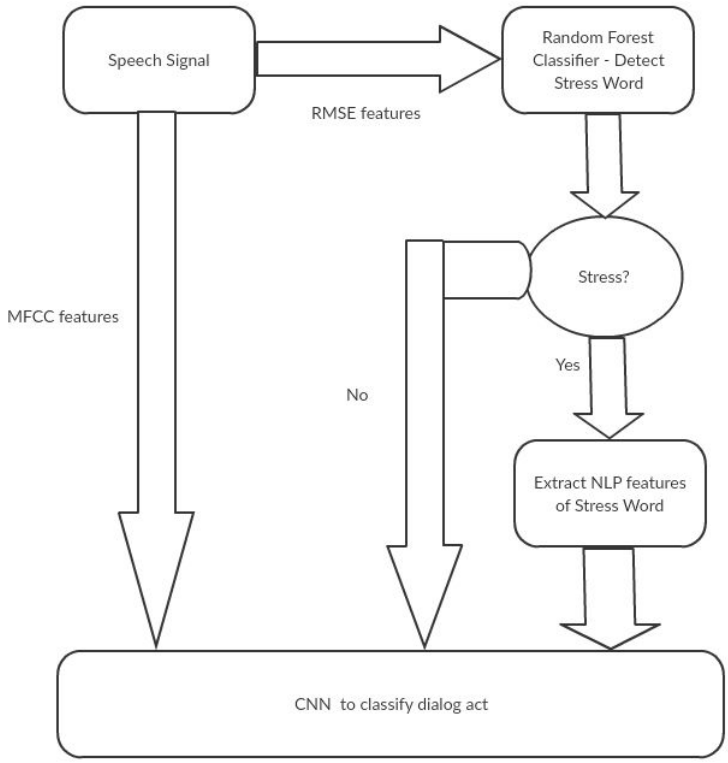


Figure 4 Model Architecture

The figure above shows the model architecture. Audio samples are initially passed to the librosa mfcc API to extract MFCC features. They are also passed to librosa RMSE API to extract the RMSE features. These features are then passed to the Random Forest Classifier which classifies whether a given utterance has a stress word or not. As a part of future work we have passed the audio sample to Google API to get the actual stressed word. POS tag of the stress word is then passed along with the MFCC features to the Convolutional Neural Network.

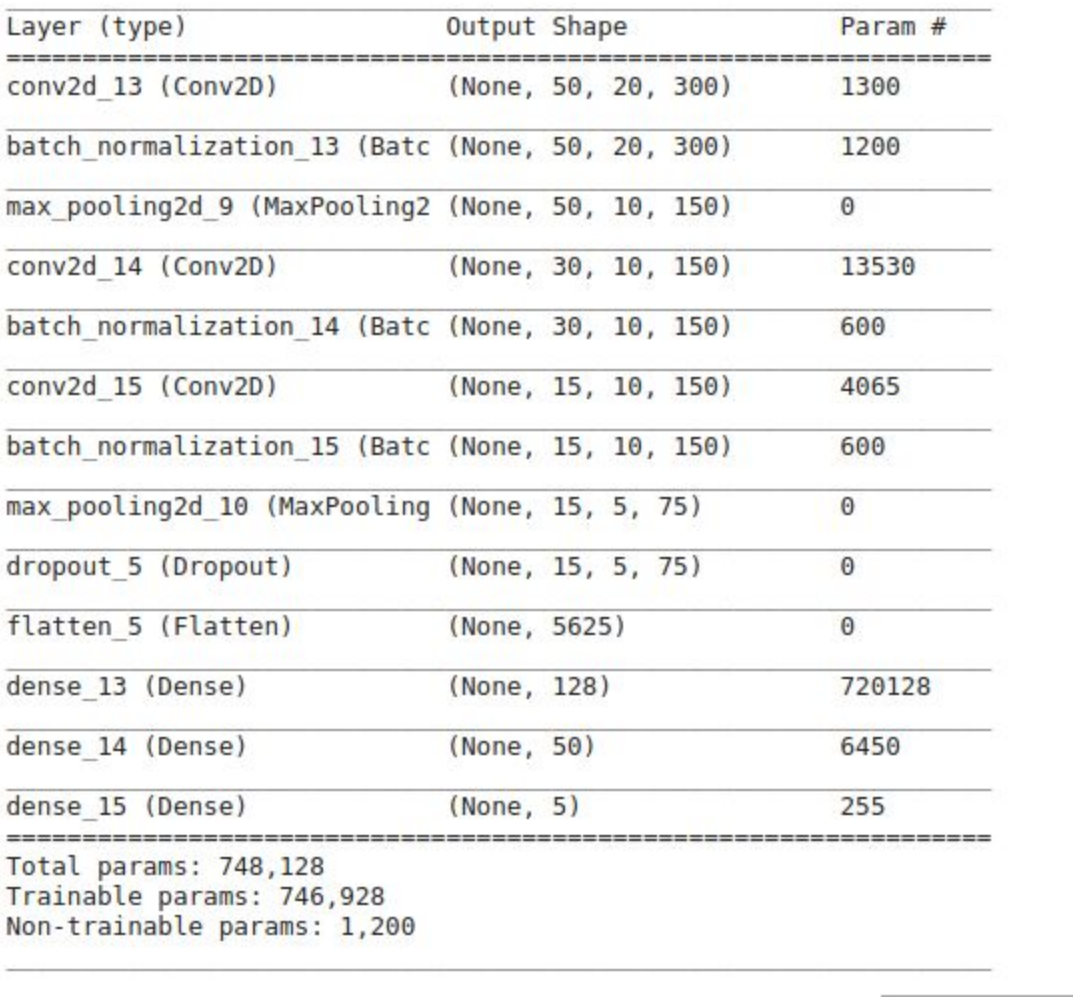
Dialog Classification

The model involved classifying the speech signals into one of the 5 classes - YES/NO question, WH question, Declarative question, view/opinion statement and descriptive statement. We had 88 speech files. For each of the speech file, MFCC features were obtained using Librosa. These MFCC features were fed to a Convolutional Neural Network (CNN) for classification. The input to a 2D Convolutional Neural Network is a 3D matrix of dimension (1,N,D) where N - Number of time frames of the speech signal, D - Number of features per time frame and 1 - Number of channels. Convolutional Neural Networks are majorly applied to the field of vision. In the domain of vision, number of channels represent the number of colors like RGB. Since there is no such concept in a speech signal, the number of channels is set to 1. Also, since the CNN expects the input size to be the same for every input sample, we pad the input sequences to a dimension of (1,20,300) - every sample is represented as a matrix of (1,20,300). These sequences along with their labels are fed to the CNN. The labels are converted into a one-hot-vector of dimension 5 - number of classes. This one-hot encoding enables us to set the loss function as a categorical cross-entropy loss.

Convolutional Neural Network

The model is a multi layer CNN with batch normalization after every convolution operation. The first CNN layer has 50 filters with same padding. This layer is followed by a Batch Normalization layer to normalize the output from the CNN layer. The Batch Normalization layer helps in faster convergence of the loss function. Finally, a Max-Pooling operation with a size of 2 is applied to the Batch Normalized output. This layer reduces the output dimension by half and also captures the most important features for every patch of size (2,2) with a stride of 1. The same set of layers are repeated two more time with CNN filter size of 30, 15 with same padding to avoid loss of information at the edge of the matrix. Before flattening the matrix to a vector, a Dropout layer is added to avoid overfitting. After flattening, the matrix into a vector, it is fed to two fully connected layer to capture interactions between

each dimension. Finally, a Dense layer is added with a softmax activation function to get the output probabilities for each of the 5 classes. Rectified Linear Unit (ReLu) is used an activation function for each of the CNN layers to facilitate better gradient flow. The gradient descent optimizer used was Adam and the model was run for 30 epochs with a batch size of 32. Below is a screenshot of the model summary using Keras. The screenshot shows the number of trainable and non-trainable parameters along with the output dimension for each of the layer



Layer (type)	Output Shape	Param #
conv2d_13 (Conv2D)	(None, 50, 20, 300)	1300
batch_normalization_13 (Batc	(None, 50, 20, 300)	1200
max_pooling2d_9 (MaxPooling2	(None, 50, 10, 150)	0
conv2d_14 (Conv2D)	(None, 30, 10, 150)	13530
batch_normalization_14 (Batc	(None, 30, 10, 150)	600
conv2d_15 (Conv2D)	(None, 15, 10, 150)	4065
batch_normalization_15 (Batc	(None, 15, 10, 150)	600
max_pooling2d_10 (MaxPooling	(None, 15, 5, 75)	0
dropout_5 (Dropout)	(None, 15, 5, 75)	0
flatten_5 (Flatten)	(None, 5625)	0
dense_13 (Dense)	(None, 128)	720128
dense_14 (Dense)	(None, 50)	6450
dense_15 (Dense)	(None, 5)	255

Total params: 748,128
Trainable params: 746,928
Non-trainable params: 1,200

Figure 5 Model Summary

Results

For the stress word detection task, we were able to obtain an accuracy of 63% which is a good number considering the fact that only the RMSE features of the speech signal were fed as an input to the Random Forest Model.
For the dialog act classification task, we were able to obtain 97.7 % training accuracy on just 30 epochs with cross-entropy loss of 0.0845. However, due to lack of speech data for this task, we were unable to validate the results on unseen test data. This high training accuracy might be an indication of overfitting the model using just 88 speech samples. Although, this model may be overfitting, this is a clear indication that the model is able to learn something only on the speech signal. A good amount of speech signals for this specific task would have helped us to get more clearer results.

Future Work

Although, the results look promising, the amount of data we had for this specific task was very low. We believe that having more speech data would enable us to build better models. Lastly, we feel that the stress word detection module can be incorporated with the dialog act classification module - for instance, we can append the part of speech tag of the stressed word to the input fed to the CNN model. There can be a good indication of dialog act class based on what was stressed. The part of speech feature could send an additional signal for dialog classification.

Reference

[1] Daniel Ortega, Dialog-act classification using Convolutional Neural Networks
[2] Vivek Kumar Rangarajan Sridhar, Srinivas Bangalore, Shrikanth Narayanan , Combining lexical, syntactic and prosodic cues for improved online dialog act tagging.
[3] Vivek Kumar Rangarajan Sridhar, Srinivas Bangalore, Shrikanth Narayanan, Exploiting prosodic features for dialog act tagging in a discriminative modeling framework.
[4] Wei Li, Yunfang Wu Multi-level Gated Recurrent Neural Network for Dialog Act Classification
[5] Yang Liu, Kun Han, Zhao Tan, Yun Lei Using Context Information for Dialog Act Classification in DNN Framework
[6] http://recognize-speech.com/feature-extraction/mfcc