Proceedings of the Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)
IEEE Xplore Part Number: CFP21OSV-ART; ISBN: 978-1-6654-2642-8

2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC) | 978-1-6654-2642-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/I-SMAC52330.2021.9640683

Deep Learning based Transcribing and Summarizing Clinical Conversations *

Niharika G Menon
Department of Computer Science Engineering
SRM Institute of Science and Technology Ramapuram
Chennai, India
nm6015@srmist.edu.in

Avani Shrivastava
Department of Computer Science Engineering
SRM Institute of Science and Technology Ramapuram
Chennai,India
as3780@srmist.edu.in

N D Bhavana
Department of Computer Science Engineering
SRM Institute of Science and Technology Ramapuram
Chennai, India
bd6266@srmist.edu.in

Judy Simon
Department of Electronics and Communication Engineering
SRM Institute of Science and Technology Ramapuram
Chennai, India
judys@srmist.edu.in

***Abstract*—Doctor-Patient interaction plays a very vital role in patient care. The essence of the interaction is lost when a lot of time is consumed in writing or typing the required patient details. These administrative tasks happen at the expense of patient care. With Artificial Intelligence and Natural Language Processing, this research work creates different ways to reduce the time spent on such onerous, trivial administrative tasks at healthcare facilities and concentrate on serving the patients in a better way. This paper proposes an automation mechanism of noise suppression to eliminate environmental disturbance, transcription and sum- summarization of the recorded conversation taking place betweenthe doctor and the patient(s) to focus only on the essential information, since abridging the entire conversation as a whole may be counterproductive. The tabular summary obtained at the end of the process can be used by the doctors and patients alike, to understand the patient history, prognoses and/or diagnoses. A supervised deep learning technique is used for noise suppression by using a convolutional network, the Google Speech- to-Text API for transcription of the conversation and a basic SVM module which categorizes text based on the given tags and relative frequency of occurrence of a word to create the tabular summary of the said doctor-patient verbal exchange.**

***Index Terms*—Transcribing, Speech to text, Noise Reduction, NLP, SVM, Deep Learning**

I. INTRODUCTION

With the rise in the number of patients because of the COVID-19 pandemic, the need for quick medical attention has become especially critical. Dealing with large volumes of patients can be a huge burden for the healthcare system and its staff as a whole. Thus decreasing a physician's workload by the use of a system to record their interactions with patients and extract important insights from this recorded report can reduce the strain on the healthcare system and increase the overall quality and speed of healthcare delivered to the patient. The storage and effective organization of medical data transcribed during physician-patient interactions is also crucial for research. Another important use of recording medical data efficiently is to provide incentives and penalties to healthcare workers and for billing and reporting purposes. This is where the idea of a system that can both transcribe medical conversations and provide prognosis comes in.

Medical scribes have been prevalent in the 20th century, as a means for the physician to focus on the interaction with the patient and getting the information they need, while letting a scribe transcribe the needed information in shorthand. There are several ways this is done, from having scribes follow the physician to their appointments and take notes using the EHR system, to physicians sending recordings of their appointments to get them transcribed by a remote scribe. The focus of the proposed system is to automate the task of scribing, which will save time and manual labor. Such a system should be able to record an appointment accurately, reduce noise and convert the speech to text. As seen in Fig 1, essential medical data can be captured from the transcription.

Noise reduction first gained popularity through the technique developed by Ray Dolby in 1966 for use on the Dolby B system, where the noise reduction was done using an encode/decode system that resulted in reducing noise up to 10 dB. Current systems have a greater focus on noise suppression using the software. More recently, there have been papers on deep learning networks such as RNNs being used for this purpose. Noise suppression can create clearer, more intelligible audio that is easy to transcribe. Thus, focusing on noise suppression can mean that using a commercial speech-to-text API will be sufficient to transcribe the audio accurately. However, there are currently no consistent benchmarks with which to compare the efficiency of noise suppression algorithms, which

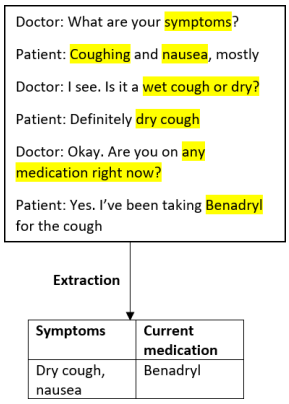


Fig. 1. Transcription with information extracted from it

makes evaluating their performance difficult.

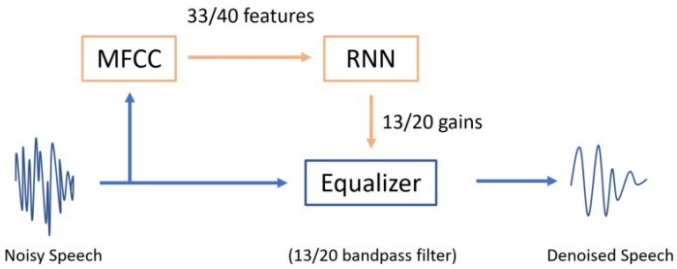


Fig. 2. Example of an existing RNN denoising system

Another aspect of efficient scribing is the extraction of a meaningful summary of all the transcribed text. This may include removing extraneous non-medical speech, summarizing relevant data such that it can directly be entered into a report, and that it is concise and accurate in capturing the information needed. This summary should also be useful for research purposes. All of this can be done by classification of text into relevant medical data and non-medical data, having the turns in conversation tagged, tagging symptoms, tagging affected body systems, and so on. Several summarization techniques have emerged in recent years, such as Attention-based systems, ASR models, and Hierarchical RNNs.

Many existing systems do not combine recent noise suppression techniques and transcription. The transcription process can suffer from a lack of medical data. Thus we propose a novel system, that can handle the effect of noise in the recorded speech using a noise suppression system, convert the speech to text and categorize the text based on tags, such as symptoms, existing medical conditions, previous medications, as required, to extract the data from the clinical interaction required to both effectively summarise the conversation and for the purpose of using this data in other applications, such as determining a prognosis. The data extracted from the transcription can also be used in other transcription systems to compensate for the lack of data available for training and testing systems, which will be elaborated on in the succeeding sections. The system proposed in this paper, uniquely combines noise suppression using deep learning, with speech to text and an SVM module to label the text, identifying the necessary medical terms from the transcription.

II. Previous Work

[1] The related work proposed the usage of Speech to text system and a text classifier to take input in audio form and convert it to text to further get results of text classification. The results of text classification were found to be: Logistic regression obtained 98.9% F-score outperforming Naive Bayes (at 97.13% F score) and SVM (at 98.85%) as it was lower for physical exam section when it was compared to the history section due to increased typing errors for physical exam section, since physical exam findings are usually depicted as phrases instead of sentences. Some limitations of this work were its inability to use speaker diarization where the patient's voice was very low and the limited amount of data it was trained with, making it inefficient in dealing with a wide range of cases. One improvement suggested was training speech recognition and punctuation insertion models specifically trained for physical exam findings to improve upon the low speed observed. The unavailability of enough and realistic clinical data can be solved by the system proposed in this paper as it can be used to create medical datasets from actual interactions with patients.

[2] The major area of research here was 'Improving the quality of the recordings' using speech enhancement The module succeeded in noise suppression and de-reverberation using RNN-N and RNN-R which trained with Fourier transform of clean distorted speech used as sample output and input of the network, respectively. As a result, noisy data could be enhanced with the RNN-R i.e. reverberation of noisy speech became clearer. This technique has been long improved upon by more recent models such as GANs, Wavenet style systems and CNNs. This model shows poor results when the recording has a combination of additive noise and reverberation. So there is a need for using newer models in our noise suppression approach.

[3] This paper was a discussion of the various challenges involved in developing an automated clinical scribe system that would record audio, convert them into transcripts and extract clinical data for meaningful summarization of a person's report, data for AI and ML algorithms and medical concepts. The need to solve some of these challenges, such as obtaining high quality audio and extracting useful data motivated our proposed system.

[4] This was a review that analyzed the current state of digital scribing using performance data from various sources. From the twenty articles in the paper, three of them describe ASR models for clinical conversations and seventeen articles presented models for extraction and classification. Although the focus of the studies was on technical validity as opposed

to extensive reporting and repeatedly studying technical and clinical usability and validity, the conclusion that the paper arrived at was that the most efficient models made use of context-sensitive word embeddings merged with attention-based neural networks. However, it is still in the dark when it comes to the advancements made by companies offering digital scribes commercially, because their research is not made public.

[5] Extractive summarization of meaningful phrases was the focus of this paper. For this purpose, it tested a multi-head attention-based model. This was implemented by choosing the most necessary statements through correlation of tokens and passing it to the 'segments and positional embeddings' module. Output: attention scores that are statistically transformed to extract key phrases and can be used for a projection on the heat-mapping tool for visual better correlation. The concerns with this system related to loss of important information and the data it requires to make better summaries can only be obtained by trials with physicians. Such an application would better assist with a medical dataset that can be made using the system proposed in our work.

[6] This particular study focused on combining voice and text data to provide a tailored situational awareness information. A bidirectional deep RNN with long short-term memory for speech recognition, and convolutional neural network applied on word vectors (that underwent custom-training for sentence-level classification tasks) worked together to categorize sentences into four sub-divisions: patient status, medical history, treatment plan, and medication reminder. It still requires improvement on the UI and real-time usage fronts.

[7] The use of annotation schemes to extract relevant clinical concepts, develop labeled corpus. Used to train the Span-Attribute Tagging (SAT) model to detect turn of speech and splitting report into symptoms of each system in the body. Turn detection had better recall when trained by treating each turn as independent input while SAT resulted in better precision. This tradeoff indicates that their utility is determined on a case-by-case basis and thus was not suitable for the versatile model we wanted to implement.

[8] Pretraining and knowledge from previously admitted patients with their outcomes from clinical notes and scientific cases about diseases, diagnosis, symptoms built up the Produced Clinical Outcome Representations (CORe) to teach the model relations between symptoms, risk factors.A Pre-trained CORe model with BioBERT weights with baselines(CNN, BOW, Discharge BERT, Bio BERT, ClinicalBERT). The proposed model had outperformed baselines except for Discharge BERT. Method to incorporate ICD code hierarchy into the model was also added; the ICD+ method assigned eight additional labels to the example diagnosis and therefore supplied the model with further information about the diagnosis during training. ICD improved the predictions. One improvement suggested was the exploration of additional clinical datasets to compensate for noisy labels, and the creation of such datasets can be facilitated by our proposed system.

[9] The basic method was to use a primitive linear processing pipeline of the speech-processing module-this included a speaker diarization module that identifies the speaker and uses this information to segregate the audio recordings. The next phase used a hierarchical recurrent neural network (RNNs) that did the job of tagging turns and sentences to their predicted class, each sentence had a single vector encoded by a word-level RNN with an attention mechanism. Extracting information from tagged sentences using a mix of complete and partial-string matches to identify terms from ontology was another technique tested. This model requires more usage in clinical settings to get useful clinical data. It has no work regarding noise suppression, which is explored in our proposedsystem.

[10] Input data was filtered out into a set of phrases comprising 26.6% of medical terms that were used to summarize consultation discussions, less than 20% of the words in the transcripts were included from the huge conversation. The results showed the final summarization of the consultation data makes up only a small portion of GP consultations, and automated summarization solutions—such as digital scribes focus majorly on identifying the 20% relevant information for auto-generating consultation summaries.

[11] This noise suppression technique named SEGAN uses a generative adversarial network for the purpose of suppressing noise. Given two inputs, a speech signal with noise and its latent representation, the network produces the enhanced sound as output. The signal is first compressed through convolutional layers, followed by PreLUs and this encoding is decoded in the decoder network. There are also skip connections in the GAN, training the network better and passing fine details of the waveform directly to the decoder without compression. This network performed better when compared with Wiener filtering using CSIG,CBAK,COVL and SSNR scores, but worse when it came to PESQ scores. It was also evaluated using a subjective test with 16 listeners, where even though both the methods had a similar SNR, SEGAN filtering was preferred for a larger percentage of the samples shown. It still does not suppress the entirety of the noise, and some solutions could be devised to reduce the high frequency disturbances that could arise due to the method. There is also a scope to evaluate it better by comparing its performance with other types of filtering.

[12] One system uses a capsule algorithm to optimize the classification of multi-label texts. After processing the input, the system uses a layer of primary capsules to capture latent data such as local order or semantic word representations of the input. Then it sends this input to a layer of classification capsules that produce the prediction vector, by adding up the inputs and multiplying them with a suitable weight matrix. As the number of iterations increases the coupling of the capsule layers increases, thus the similarity of their predictions also increases. This method optimizes classification when compared to several baselines such as CNNs and LSTMs, and has a simplistic model which is viable for real time applications. However, it still faces limitations in terms of

applications as it does poorly with higher frequency datasets.

III. METHODOLOGY

We began by preparing the dataset to make it more balanced. Once the dataset was prepared, we employed the usage of an auto-encoder model for noise suppression.

There are two existing systems discussed for noise suppression [2] and [11] to tackle some of the challenges mentioned in [3] relating to poor quality of recorded sound. Because [2] is a very primitive network, we opt for a more efficient Convolutional Autoencoder architecture inspired by [11]. It has five one-dimensional convolutional layers, along with seven one- dimensional deconvolutional layers, and six skip connections. This is depicted in Fig 3. The model, once trained, produces a clean sound sample that is used for the next module. This handles some of the deficiencies in recorded sound quality mentioned in [3].

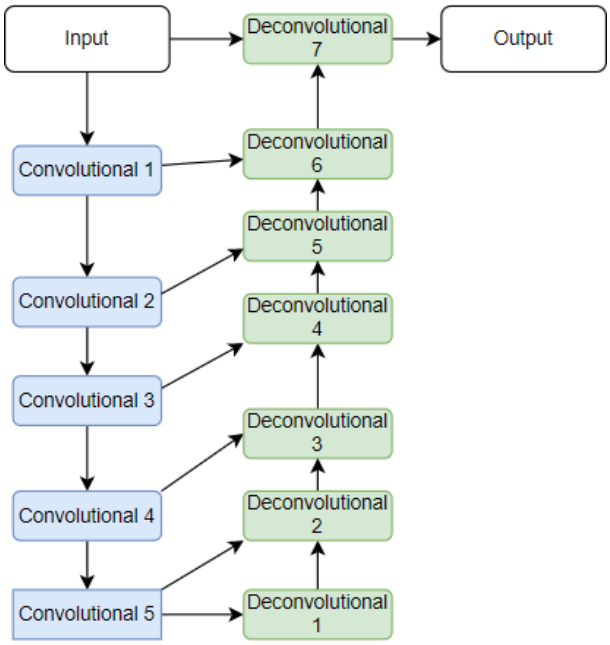


Fig. 3. Convolutional Autoencoder Network for Noise Suppression

Since this paper's main focus points are the noise suppression algorithm and the tabular summary, we have used the commercial Speech-To-Text API available on the Google Cloud Platform. This service provided by Google Cloud Platform works on an audio file or real-time audio input through the microphone and converts it into text. There are a lot of benefits of using the Speech-to-Text API. The availability of Speech adaptation - you can specify keywords, rare words, some hints, to improve the accuracy of the transcribing process; the availability of domain-specific models and real-time audiostreaming.

Once the noise suppression process is completed, the output audio file is passed as input to the API, which then transcribes the audio and gives a text as the output.

IV. EXPERIMENT

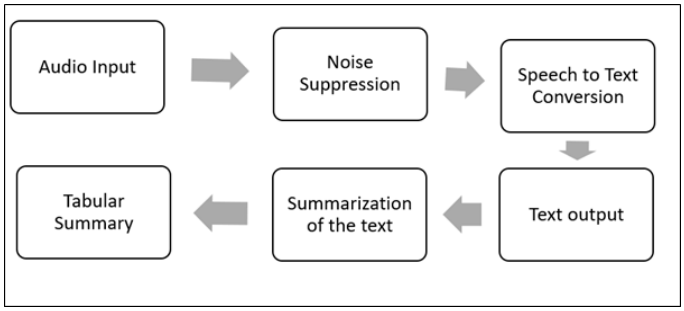


Fig. 4. Process Workflow

While the entire process is divided into three major modules, Fig 4 illustrates how the entire workflow takes place in a total of six steps. At first, the audio recording of the verbal exchange is obtained and stored in appropriate format. Next, noise suppression is performed on the recording to reduce or eliminate unwanted sounds that may be present. In the subsequent step, the speech-to-text conversion is carried out using the commercial Google API. The use of the API ensures quick and accurate text output. The obtained text output is then used as input to the summarization model. The trained model works on this text to produce a tabular summary. This final output could be put to use by the doctor(s) or the patient(s) in ways they find useful.

V. EXPERIMENT

A. Noise suppression module

The dataset for this experiment is prepared by downloading clean sound samples from an online audio library and adding noise to produce noisy samples. The noises mixed in are noises that are commonly found both indoors and outdoors. Downsampling the noises and the clean sound to 16 kHz. The noises are extended to the same length as the speech files and added to the samples to get noisy samples. The clean and noisy samples; then used to make a dataset. Then we split it into training and testing data.

After the preparation, we fit the model onto the data. The use of Adam as an optimizer produces more consistent outputs than other optimizers. We train the model using the noisy input (noisy sound) and the corresponding clean output (noise-suppressed audio) given by the training dataset. Once the model has been trained, the model can suppress the noise in the audio, producing a clean file. This audio file is used in the next part of the project.

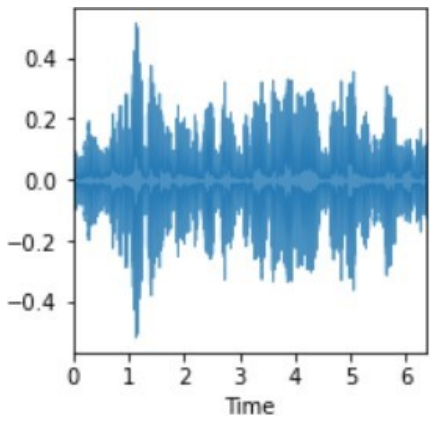


Fig. 5. Noisy Wave

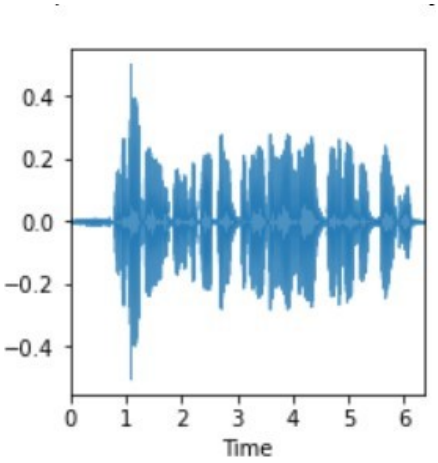


Fig. 6. Cleaned Wave

B. *Transcription module*

The Speech to text API works on the output provided by the noise-suppression module. To directly predict the phoneme target. The API uses a quantized LSTM acoustic model trained with connectionist temporal classification (CTC). The memory footprint is minimized using an SVD based compression approach and single language model for both dictation and voice command domains, which is built using Bayesian interpolation. Finally, to ensure effective handling of device-specific information effectively, proper names, and other context-dependent information, vocabulary items are inserted into the decoder graph and bias the language model on the fly. The API processes requests on audio files of duration up to 1 minute sent synchronously. This way the API ensures that the speech recognition is completed on one audio file before proceeding to the next one. The API gives a response after processing and recognizing all of the audio.

```
#import library
import speech_recognition as sr
# Initialize recognizer class (for recognizing the speech)
r = sr.Recognizer()
# Reading Audio file as source # listening the audio file and store in audio_text variable
with sr.AudioFile(r'D:\Speech-to-text-Noise-Supression\New-Recording-2.wav') as source:
    audio_text = r.listen(source)
  # recoginize_() method will throw a request error
  # if the API is unreachable, hence using exception handling
    try:
        # using google speech recognition
        text = r.recognize_google(audio_text)
        print('Converting audio transcripts into text ...')
        print(text)
    except:
         print('Sorry.. run again...')
```

Fig. 7. Transcription with Google API - sample code

```
Converting audio transcripts into text ...

Hi. How are you? I am fine doctor. I have been very sick for the last two
days. I am having nausea, headache and body pain. Could you help me? Of
course. Let me run some tests before that.
```

Fig. 8. Google API transcription - sample output

C. *Labelling module*

The need for a labeled and structured dataset in text form is evident and much needed to make our model learn the patterns and correlations to filter out most used words, unigrams, and bigrams then only the actual essence i.e. the accurate features needed would be got as a summary. Obtaining it and labeling it from the previous speech conversion API's output was a critical problem faced since what we got at the end was a highly unstructured long paragraph (due to lack of appropriate datasets and also since it's usually done manually). So the basic idea was to reduce the paragraphs of texts to a labeled set and label it according to the four categories (patient detail, symptoms, previous prescription, current situation).

After text pre-processing; the process of removal of punctuations, stop words, least used words, lemmatization (an algorithm for deducing the root word from a given word by referring the dictionary and the context of the conversation as a labelled training set) we applied some Feature engineering, a process that picks out features(in our case important words). The feature engineering process: Now, as we all have seen before, in our case, our data had to be represented in rows to be treated as a single document of the corpus. We went ahead with providing the module with a labelled dataset of the categories i.e the features to pick out,the features selected from the set varied according to the feature creation method we selected: we went ahead with the TF-IDF vector(TF -Term Frequency, IDF- Inverse Document Frequency), a score-value to judge the relative importance of a term in the corpus based on the n-gram model which decides the rank of a word based on the n-grams(in our module:unigrams and bigrams). This method together with Word Count Vectors is known as 'Bag of Words' and here the order in which words appear is not considered, the TF-IDF method was used since its value increases proportionally with the frequency of a word appearing.

text	punctuation_removed
i feeling good now, just some slight head ache...	i feeling good now just some slight head ache ...
any more giddiness?	any more giddiness
occasionally , yes i do feel giddy sometimes	occasionally yes i do feel giddy sometimes
how is ur diet , feel hungry now?	how is ur diet feel hungry now
no, i don't feel that hungry still	no i don't feel that hungry still
that may be signs of bloating due to meds , it...	that may be signs of bloating due to meds itl...
how is your arm pain	how is your arm pain
still arm pain yes , its painful	still arm pain yes its painful

stopword_removed	frequentword_removed
feeling good slight head ache anf fever reduce...	feeling good slight head ache anf fever reduce
giddiness	giddiness
occasionally yes feel giddy sometimes	occasionally giddy sometimes
ur diet feel hungry	ur diet hungry
don't feel hungry still	don't hungry
may signs bloating due meds itll clear	may signs bloating due meds itll clear
arm pain	arm pain
still arm pain yes painful	arm pain painful

Fig. 9. Text Preprocessing

Now our dataset was cleaned and important parts were present, from here our final task was to select sentences and categorize them into categories to get a tabular form, columns like patient name, symptoms, prescribed previous medication, important comments needed to be picked from our pre-processed paragraph of text so that the resultant table could give the doctor the current situation of the patient at a glance without asking the patient and effectively provide consultation for more patients in a short timespan(labelling module).

We made use of machine learning to train our model to pick and correctly place the sentences in the appropriate category/columns. A machine learning model requires numerical values and labels for accurate prediction; this is done by label encoding which creates a dictionary to map each label to a numerical value, for example if the model encounters a proper noun and from the labelled dataset it's known that its the patient's name then it maps the name to category 1, where we had set the numerical value '1' for the Patient Name columns, so the name is mapped to the label 1 i.e to Patient name column.

This step involved setting up a test set to check the prediction percentage of new unseen data after labeling, the dataset was split in the ratio 85% training test and 15% test set (was done with new data so that the evaluation metric remained less biased). This phase also enabled us to do hyperparameter tuning using cross-validation i.e control the learning rate so the model does not overfit and simply copy the noise too; and also applying the Chi-squared test to see what unigrams and bigrams(from the TDF-IF model for selecting words which are more relevant to the given category of the tabular format(encoded categories) are most correlated with each category was suggested in one of the works we came across and it had worked well in our case too.

Parameter	Value
N-gram range	(1, 2)
Maximum DF	1
Minimum DF	10
Maximum features	300

Fig. 10. Parameters based on which the TDF-IF model decides the relatively important sentences.(DF-document frequency)

First, we can see what hyperparameters the model has:

```
svc_0 =svm.SVC(random_state=8)

print('Parameters currently in use:\n')
pprint(svc_0.get_params())
```

```
Parameters currently in use:

{'C': 1.0,
 'cache_size': 200,
 'class_weight': None,
 'coef0': 0.0,
 'decision_function_shape': 'ovr',
 'degree': 3,
 'gamma': 'auto',
 'kernel': 'rbf',
 'max_iter': -1,
 'probability': False,
 'random_state': 8,
 'shrinking': True,
 'tol': 0.001,
 'verbose': False}
```

We'll tune the following ones:

- C: Penalty parameter C of the error term.
- kernel: Specifies the kernel type to be used in the algorithm.
- gamma: Kernel coefficient.
- degree: Degree of the polynomial kernel function.

Fig. 11. Picking out all the existent Hyperparameters in the SVM model

So now finally the long text from the speech to text API was converted to a tabular format which has classified the long sentences into columns such as patient name, symptoms etc. so it was much easier to pick out the data needed instead of going through the whole text document , now all we needed to confirm was that if the classification was done accurately, there comes the use of a predictive model to predict the accuracy percentages and find out if actually the word/sentence classified as a symptom was actually a symptom which was basically testing if our hyperparameter tuning was done right, was the set of hyperparameters set a good choice.

Among the several machine learning models tested in the literature review to figure out which one may fit better to the data, it is concluded that for this domain a Support vector model which doesn't overfit and has the best result in predicting if our dataset was categorized accurately into the four categories by cross checking the unseen dataset with a labelled set(consisting of the categorized document).

A Support vector machine an algorithm used in machine learning classification and regression (mostly used for classification) .It is preferred over other algorithms due to its ability of providing a notable accuracy despite running lesser number of computations and it also gives reliable results even if there is less data which is our major problem here as well .A SVM takes data points(data represented as vectors) and produces the hyperplane that best classifies the tags, Hyperplane is an imaginary line or a decision boundary,

anything that falls to one side of it we will classify as falling under that category.

We also did a random search and grid search to cover a wider range of values for each hyperparameter in a relatively lesser execution time, we needed to confirm which search would select the hyperparameters more accurately so we implemented both and grid search was finally used since it showed a mean accuracy of 94.98% while random search gave a mean accuracy of 92.1% .

We first need to define the grid:

```
# C
C = [.0001, .001, .01]

# gamma
gamma = [.0001, .001, .01, .1, 1, 10, 100]

# degree
degree = [1, 2, 3, 4, 5]

# kernel
kernel = ['linear', 'rbf', 'poly']

# probability
probability = [True]

# Create the random grid
random_grid = {'C': C,
               'kernel': kernel,
               'gamma': gamma,
               'degree': degree,
               'probability': probability
              }

pprint(random_grid)
```

```
{'C': [0.0001, 0.001, 0.01],
 'degree': [1, 2, 3, 4, 5],
 'gamma': [0.0001, 0.001, 0.01, 0.1, 1, 10, 100],
 'kernel': ['linear', 'rbf', 'poly'],
 'probability': [True]}
```

```
# First create the base model to tune
svc = svm.SVC(random_state=8)

# Definition of the random search
random_search = RandomizedSearchCV(estimator=svc,
                                   param_distributions=random_grid,
                                   n_iter=50,
                                   scoring='accuracy',
                                   cv=3,
                                   verbose=1,
                                   random_state=8)

# Fit the random search model
random_search.fit(features_train, labels_train)
```

Fig. 12. Preparing SVM for random search (used for cross-validation of the hyperparameters chosen)

Grid Search Cross Validation

```
# Create the parameter grid based on the results of random search
C = [.0001, .001, .01, .1]
degree = [3, 4, 5]
gamma = [1, 10, 100]
probability = [True]

param_grid = [
  {'C': C, 'kernel':['linear'], 'probability':probability},
  {'C': C, 'kernel':['poly'], 'degree':degree, 'probability':probability},
  {'C': C, 'kernel':['rbf'], 'gamma':gamma, 'probability':probability}
]

# Create a base model
svc = svm.SVC(random_state=8)

# Manually create the splits in CV in order to be able to fix a random_state (GridSearchCV doesn't have that argument)
cv_sets = ShuffleSplit(n_splits = 3, test_size = .33, random_state = 8)

# Instantiate the grid search model
grid_search = GridSearchCV(estimator=svc,
                           param_grid=param_grid,
                           scoring='accuracy',
                           cv=cv_sets,
                           verbose=1)

# Fit the grid search to the data
grid_search.fit(features_train, labels_train)
```

Fig. 13. Preparing SVM for grid search (used for cross-validation of the hyperparameters chosen)

Figures below give the accuracy, precision, f1 score (parameters which helped us determine which model to use), although all the models have almost same accuracy and other values are similar too, we went ahead with SVM because of its high precision and f1 scores which were consistent in each cycle tried with different ratio of train and test sets.

Model	Training Set Accuracy	Test Set Accuracy
Gradient Boosting	100%	94%
Multinomial Logistic Regression	98%	94%
SVM	95%	94%
Multinomial Naïve Bayes	95%	93%
K Nearest Neighbors	95%	92%
Random Forest	100%	92%

Fig. 14. Accuracy of models

KNN

Classification report

	precision	recall	f1-score	support
0	0.91	0.95	0.93	81
1	0.93	0.88	0.91	49
2	0.97	0.92	0.94	72
3	0.97	0.96	0.97	72
4	0.86	0.92	0.89	60
avg / total	0.93	0.93	0.93	334

SVM

Classification report

	precision	recall	f1-score	support
0	0.88	0.98	0.92	81
1	0.96	0.94	0.95	49
2	0.97	0.89	0.93	72
3	0.99	0.99	0.99	72
4	0.93	0.90	0.92	60
avg / total	0.94	0.94	0.94	334

RANDOM FOREST

Classification report

	precision	recall	f1-score	support
0	0.87	0.95	0.91	81
1	0.90	0.94	0.92	49
2	0.97	0.86	0.91	72
3	0.97	0.96	0.97	72
4	0.95	0.93	0.94	60
avg / total	0.93	0.93	0.93	334

MULTINOMIAL NAÏVE BAYES

Classification report

	precision	recall	f1-score	support
0	0.91	0.96	0.93	81
1	0.96	0.90	0.93	49
2	0.97	0.90	0.94	72
3	0.97	0.99	0.98	72
4	0.87	0.90	0.89	60
avg / total	0.94	0.93	0.93	334

Fig. 15. Classification Report of the SVM model v/s other models

VI. RESULTS

A. Noise suppression

The module shows a significant reduction in sound after a test with clean and noisy samples. There is an observed reduction in noise both from a study of the waveforms and

subjective observation. However, the reduction also tends to cut down some of the speech, so there is scope for improvement in this regard, perhaps by training with more data and fine tuning parameters and modifying the layers of the model.

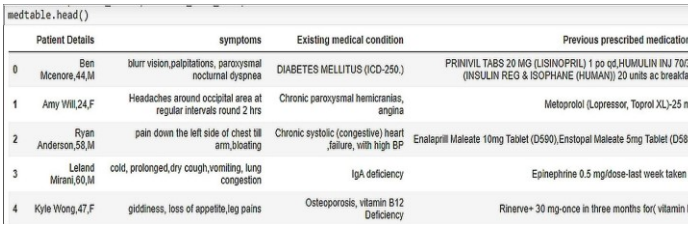
B. Transcription

The module transcribes speech to text fairly well. There is a need to work on the accuracy of the transcription as it is still unable to capture some words, especially if the speaker has a heavy accent.

C. Labelling

It was observed that unigrams worked better than bigrams(for medical conversations and medical terms) in selecting the most used words and ranking them based on their frequency in the document. Finally an acceptable accuracy was obtained from the support vector machine to predict if our classification was done right, (hence SVM was selected), although the scope to improve the whole process and use deep learning models(specifically designed for this scenario) instead of the basic machine learning models is not taken into consideration and thus could be a drawback here since all possibilities to summarize were not thoroughly explored, using deep learning models would help us scale down to picking more parameters form the text instead of the just the basic classification we haddone.

At the end we had finally achieved our end output- a new dataset in tabular form that can be further used in disease prediction or analysis by just using medical transcripts.



medtable.head()

	Patient Details	symptoms	Existing medical condition	Previous prescribed medicatio
0	Ben Mcenore,44,M	blurr vision,palpitations, paroxysmal nocturnal dyspnea	DIABETES MELLITUS (ICD-250.)	PRINIVIL TABS 20 MG (LISINOPRIL) 1 po qd,HUMULIN INJ 70/ (INSULIN REG & ISOPHANE (HUMAN)) 20 units ac breakfa
1	Amy Will,24,F	Headaches around occipital area at regular intervals round 2 hrs	Chronic paroxysmal hemicranias, angina	Metoprolol (Lopressor, Toprol XL)-25 m
2	Ryan Anderson,58,M	pain down the left side of chest till arm,bloating	Chronic systolic (congestive) heart ,failure, with high BP	Enalaprill Maleate 10mg Tablet (D590),Enstopal Maleate 5mg Tablet (D58
3	Leland Mirani,60,M	cold, prolonged,dry cough,vomiting, lung congestion	IgA deficiency	Epinephrine 0.5 mg/dose-last week taken
4	Kyle Wong,47,F	giddiness, loss of appetite,leg pains	Osteoporosis, vitamin B12 Deficiency	Rinerve+ 30 mg-once in three months for(vitamin

Fig. 16. Final Tabular Output, the columns are the tags and information picked from text is put in the matching tag/column

CONCLUSION

Thus, a novel method has been developed for the purpose of suppressing noise in recorded audio, creating medical transcripts, and creating a dataset by filtering out significant words to create a dataset that can be used in several applications like prognosis and diagnosis development. Even though there are several transcription systems, many suffer from a lack of data and do not use the noise suppression technique used in this work. This method can be used to create more data to train any transcription system through its labelling module and it uses an autoencoder convolutional network to suppress noise. The reduction is reasonable, but improvements can be made to avoid cutting the speech along with the noise during suppression. The labelling model is implemented using a variety of machine learning models, out of which SVM proved the most accurate and consistent. However, the accuracy of this module may be further improved by attempting the classification with competitive deep learning approaches, which is the scope for future work. The model displays reasonable accuracy and is one of the few models that incorporates noise suppression, speech to text, and work on labeling medical and non-medical data. Due to the scarce availability of medical data online, especially labeled medical data, the ability to create such datasets is essential for applications that require such data. Some possible future enhancements of this project are testing out other noise suppression methods, training with more data, and trying the same with other classification and labeling techniques. Usage of such a dataset in other applications may yield further insights on the other possible improvements that can be made.

REFERENCES

[1] Wang, J., Lavender, M., Hoque, E., Brophy, P. and Kautz, H., 2021. A patient-centered digital scribe for automatic medical documentation. JAMIA Open.

[2] Valentini-Botinhao, C. and Yamagishi, J., 2018. Speech enhancement of noisy and reverberant speech for text-to-speech. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 26(8), pp.1420-1433.

[3] Quiroz, J.C., Laranjo, L., Kocaballi, A.B., Berkovsky, S., Rezazadegan, D. and Coiera, E., 2019. Challenges of developing a digital scribe to reduce clinical documentation burden. NPJ digital medicine, 2(1), pp.1-6.

[4] van Buchem, M.M., Boosman, H., Bauer, M.P., Kant, I.M., Cammel, S.A. and Steyerberg, E.W., 2021. The digital scribe in clinical practice: a scoping review and research agenda. NPJ digital medicine, 4(1), pp.1-8.

[5] Kanwal, N. and Rizzo, G., 2021. Attention-based Clinical Note Summarization. arXiv preprint arXiv:2104.08942.

[6] Yun, K., Lu, T. and Huyen, A., 2020, May. Transforming unstructured voice and text data into insight for paramedic emergency service using recurrent and convolutional neural networks. In Pattern Recognition and Tracking XXXI (Vol. 11400, p. 1140006). International Society for Optics and Photonics.

[7] Shafran, I., Du, N., Tran, L., Perry, A., Keyes, L., Knichel, M., Domin, A., Huang, L., Chen, Y., Li, G., and Wang, M., 2020. The medical scribe: corpus development and model performance analyses. arXiv preprint arXiv:2003.11531.

[8] van Aken, B., Papaioannou, J.M., Mayrdorfer, M., Budde, K., Gers, F.A. and Löser, A., 2021. Clinical outcome prediction from admission notes using self-supervised knowledge integration. arXiv preprint arXiv:2102.04110.

[9] Finley, G., Edwards, E., Robinson, A., Brenndoerfer, M., Sadoughi, N., Fone, J., Axtmann, N., Miller, M., and Suendermann-Oeft, D., 2018, June. An automated medical scribe for documenting clinical encounters. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations (pp. 11-15).

[10] Quiroz, J.C., Laranjo, L., Kocaballi, A.B., Briatore, A., Berkovsky, S., Rezazadegan, D. and Coiera, E., 2020. Identifying relevant information in medical conversations to summarize a clinician-patient encounter. Health Informatics Journal, 26(4), pp.2906-2914.

[11] Pascual, S., Bonafonte, A. and Serra, J., 2017. SEGAN: Speech enhancement generative adversarial network. arXiv preprint arXiv:1703.09452.

[12] Manoharan, J. Samuel. "Capsule Network Algorithm for Performance Optimization of Text Classification"; Journal of Soft Computing Paradigm(JSCP) 3, no. 01 (2021): 1-9.