**Topic**: Spoken Dialog System

# Spoken Dialogue System

## Introduction

Spoken dialogue systems are considered a breed of interfaces that enable humans to communicate with machines naturally and efficiently using a conversational paradigm. [1](Zue and Seneff, 2008)

the most prominent component in today’s virtual personal assistants (VPAs). Among these VPAs, Microsoft’s Cortana, Apple’s Siri, Amazon Alexa, Google Assistant, and Facebook’s M, have incorporated SDS modules in various devices, which allow users to speak naturally in order to ﬁnish tasks more eﬃciently. [5](Nisar, 2018)

Uses several components related to known NLP applications [1] [3](Papangelis, 2020, 2020; Zue and Seneff, 2008):

1. Speech Recognition
2. Speech Synthesis
3. NLU and NLG (natural language understanding, natural language generation)
4. Discourse modeling
5. Dialogue management
6. Dialogue State Tracking
7. Text To Speech

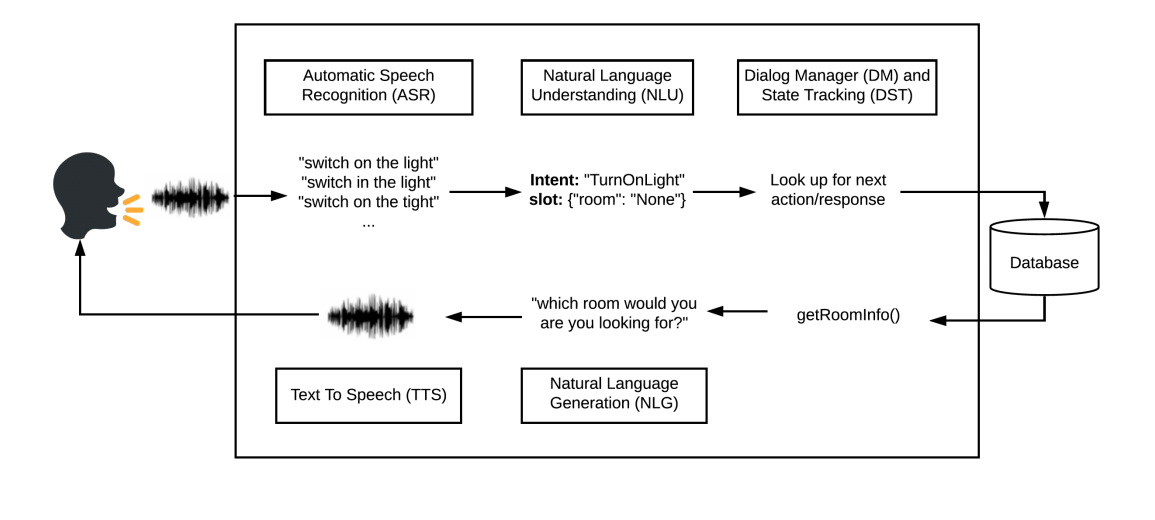


Figure 1: Depiction of a typical dialogue system [3](Papangelis, 2020)

widely used mainly in telephone-based information and transaction services. They are highly susceptible to variations in the speech signal caused by environmental and user variations, it is crucial to increase the robustness of these systems. [2] (McTear, 2010)

One of the difficulties posed by SDSs is that, even when the domain of a task is defined, the vocabulary related to this task may not necessarily be clear to the users. it may be impractical or even impossible to provide users with explicit information about supported vocabulary and out-of-vocabulary (OOV) words. Thus, how to correctly reject the OOVs spoken by users, instead of recognizing them as the closest words in the dictionary, presents another difficult but crucial issue. [2](McTear, 2010)

Most NLG systems in common use employ rules and heuristics and tend to generate rigid and stylized responses without the natural variation of human language. They are also not easily scaled to systems covering multiple domains and languages. [4](Wen, 2015)

Spoken Dialogue Systems are usually divided into two types — Task Oriented and Non-Task Oriented. Task oriented systems are built for specialized use cases such as a banking app using which a user can withdraw or deposit money. Non-Task Oriented Dialogue Systems are built for a general use case, and form a majority of Dialogue Systems including Siri, Alexa. [6](Gupta, 2020)

## Building a Dialogue system

The traditional conversational systems have rather complex and/or modular pipelines. [5](Nisar, 2018)

The advance of deep learning technologies has recently risen the applications of neural models to dialogue modeling. Nevertheless, applying deep learning technologies for building robust and scalable dialogue systems is still a challenging task and an open research area as it requires deeper understanding of the classic pipelines as well as detailed knowledge on the benchmark of the models of the prior work and the recent state-of-the-art work. [5](Nisar, 2018)

a typical Dialogue system looks like this [5](Nisar, 2018):

Diagram

Description automatically generated

Figure 2: depiction of dialog system (2) [5](Nisar, 2018)

In a Dialogue system what we actually get from the user when he uses our system is either speech or text. If it is speech, we can run it through ASR and get the text(utterance). The first thing you need to do when you get the utterance from the user, is to understand what does the user want, and this is the **intent classification problem**. [5] the intent being a **form** that a user needs to fill in, And each intent has a set of **fields** or so-called slots that must be filled in to execute the user request. [5](Nisar, 2018)

So, let’s say we have two slots here like FROM and TO. And we need a slot tagger to extract slots from the user utterance. Whenever we get the utterance from the user. think of it as a sequence tagging and we can solve it as by sequence tagging tasks using BIO Scheme coding. [5](Nisar, 2018)

Another challenge in dialogue systems is to **handle history information** when next utterance is dependent on previous one. Here we need to add context to our intent classifier and slot tagger. [5](Nisar, 2018)

Another challenge in Dialogue systems is when user suddenly changes context. e.g. Imagine that at first user says, **“Give me directions from L.A.”**, and then we ask, **“Where do you want to go?”** and this time, the user says, **“Forget about it, let’s eat some KFC chicken first.”** So, this is where we need to understand that the intent has changed and we should forget about all the previous slots that we had and all the previous information that we had because we don’t need them anymore. [5] Now the intent classifier gives us navigation find and the category, which is a slot and it has the value of KFC chicken. Then, system makes a query to the database or knowledge base like Yelp and dialog manager understands, “Okay, let’s start a new context and find some KFC chicken.” and the assistant outputs, “Okay, here are nearby KFC chicken places” [5](Nisar, 2018)

**dialog manager** is responsible for two tasks. The first one is dialog state tracking. So we need to understand what the user wanted throughout the conversation and track that state. And also, it does dialog policy managing. E.g. there is a certain policy, which says that, okay, if the state is the following then we need to query some information from the user or request some information from the user or we just inform the user about something. And we can also query backend services like Google Maps or Yelp [5](Nisar, 2018)

when we are ready to give users some information, we use **natural language generation(NLG)** box that outputs the speech for the user so that this is a conversation. [5](Nisar, 2018)

It is important to note these systems still make errors, and the individual blocks are trained separately and then joined together. This means that an NLU block is not trained to handle the errors made by the STT block, and errors propagate through the system. [6](Gupta, 2020)

## Speech Recognition

Modern speech recognition systems have come a long way since their ancient counterparts. They can recognize speech from multiple speakers and have enormous vocabularies in numerous languages. [7](Amos, 2018)

The first component of speech recognition is, of course, speech. (1) Speech must be converted from physical sound to an electrical signal with a microphone, and (2) then to digital data with an analog-to-digital converter. (3) Once digitized, several models can be used to transcribe the audio to text. [7](Amos, 2018)

Most modern speech recognition systems rely on what is known as a Hidden Markov Model (HMM). This approach works on the assumption that a speech signal, when viewed on a short enough timescale (say, ten milliseconds), can be reasonably approximated as a stationary process—that is, a process in which statistical properties do not change over time. [7](Amos, 2018)

In a typical HMM, the speech signal is divided into 10-millisecond fragments. The power spectrum of each fragment, which is essentially a plot of the signal’s power as a function of frequency, is mapped to a vector of real numbers known as cepstral coefficients. The dimension of this vector is usually small—sometimes as low as 10, although more accurate systems may have dimension 32 or more. The final output of the HMM is a sequence of these vectors. [7](Amos, 2018)

To decode the speech into text, groups of vectors are matched to one or more phonemes—a fundamental unit of speech. This calculation requires training, since the sound of a phoneme varies from speaker to speaker, and even varies from one utterance to another by the same speaker. A special algorithm is then applied to determine the most likely word (or words) that produce the given sequence of phonemes. [7](Amos, 2018)

One can imagine that this whole process may be computationally expensive. In many modern speech recognition systems, neural networks are used to simplify the speech signal using techniques for feature transformation and dimensionality reduction before HMM recognition. Voice activity detectors (VADs) are also used to reduce an audio signal to only the portions that are likely to contain speech. This prevents the recognizer from wasting time analyzing unnecessary parts of the signal. [7](Amos, 2018)

A handful of packages for speech recognition exist on PyPI. A few of them include:

* apiai
* assemblyai
* google-cloud-speech
* pocketsphinx
* SpeechRecognition
* watson-developer-cloud
* wit

Some of these packages—such as wit and apiai—offer built-in features, like natural language processing for identifying a speaker’s intent, which go beyond basic speech recognition. Others, like google-cloud-speech, focus solely on speech-to-text conversion.

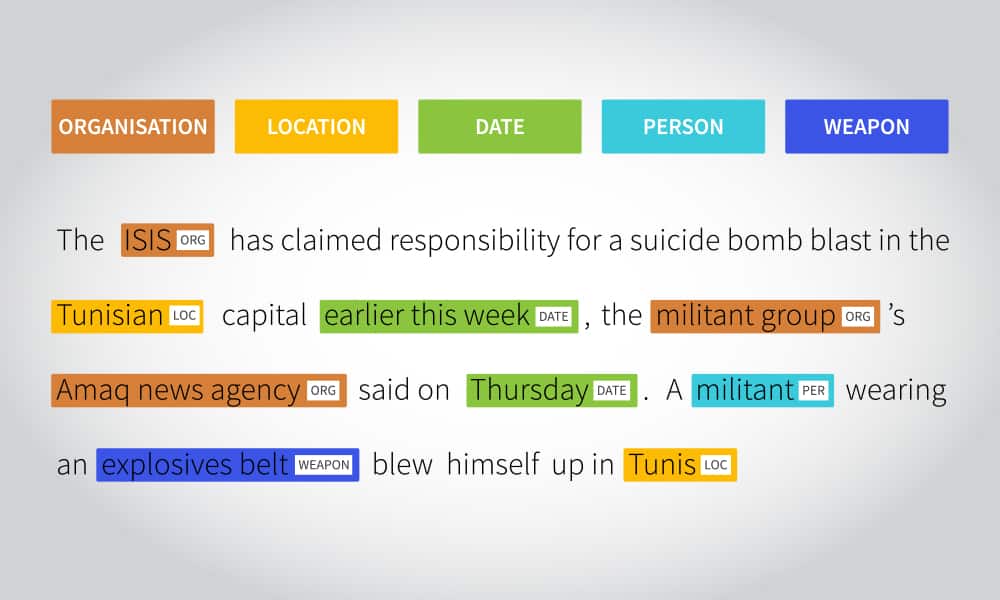
## Natural Language Understanding

NLU is a subset of NLP, and it’s the part that enables machines to understand and comprehend human language and determine the intended meaning of a sentence *(1)* and differentiate between ambiguous words.

By analyzing the data and using different algorithms, NLU can turn normal text into a structured form that helps the machine to understand it*( https://www.techtarget.com/)*.(“Activity Matters,” 2020)

Some of the techniques that are used to understand a text and extract information from it are:

* **Named Entity Recognition (NER)** one of the most basic techniques, and it’s used to identify the tag of the word used in that text, and these tags are: people, dates, location, organization, and much more*(https://blog.aureusanalytics.com/)*(“Aureus Analytics,” 2021).



*NER Applied on a text* (“Aureus Analytics,” 2021)

* **POS Tagging** is categorizing words in a text in correspondence with a particular part of speech, depending and its context, and this task isn’t easy since the language has a lot of ambiguity , and these tags in English are [noun](https://en.wikipedia.org/wiki/Noun), [verb](https://en.wikipedia.org/wiki/Verb), [article](https://en.wikipedia.org/wiki/Article_(grammar)), [adjective](https://en.wikipedia.org/wiki/Adjective), [preposition](https://en.wikipedia.org/wiki/Preposition_and_postposition),

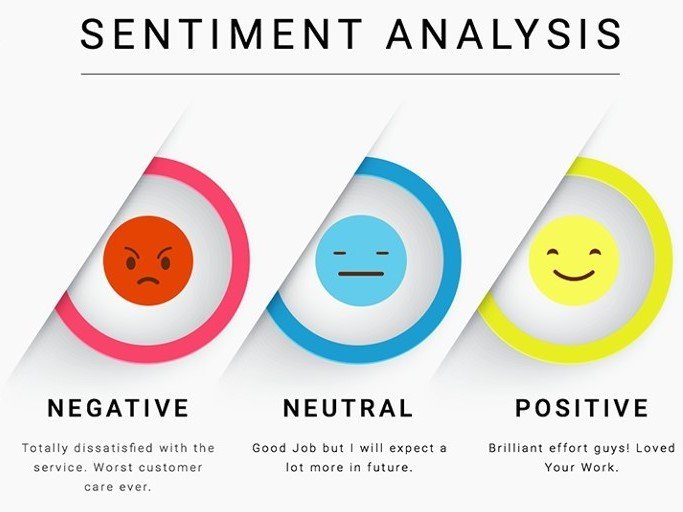
[pronoun](https://en.wikipedia.org/wiki/Pronoun), [adverb](https://en.wikipedia.org/wiki/Adverb), [conjunction](https://en.wikipedia.org/wiki/Grammatical_conjunction), and [interjection](https://en.wikipedia.org/wiki/Interjection)(wlabu, 2020).

**Chart

Description automatically generated with medium confidence**

*POS Tagging example* (wlabu, 2020)

* **Sentiment Analysis** this technique is used to classify a text into feelings and emotions, it could be as simple as positive and negative, or more complex like giving it a (sentiment score) *(https://blog.aureusanalytics.com/)*(“Aureus Analytics,” 2021).

****

*Sentiment analysis*(“Aureus Analytics,” 2021)

We use NLU in many applications, for example in **Question Answering** the computer will answer questions automatically in human language, and in **Emotion Mining** we use sentiment analysis to understand feelings and opinions(“NLU - beginner’s guide,” 2022).

## Zero-Shot Classification

We can assign the proper label to a piece of text using the zero-shot categorization method. This relationship holds true regardless of the aspect and text domain. It might be a subject, feeling, or occurrence that the label describes, for instance. We require a zero-shot model in order to execute zero-shot classification. We can categorize data that was not previously utilized to create the model using a zero-shot model. Simply said, it compares your data to a model created by someone else. With just a few lines of code, Hugging Face enables us to use this pre-trained model to do classification. (Mumbi, 2022)

### What is the process of zero-shot learning? (Kundu, 2023; Mumbi, 2022)

The following data is used in zero-shot learning:

**Seen Classes:** The data classes that were used to train the deep learning model .

**Unseen Classes:** These are the data classes that need to be generalized by the current deep model. These classes' data weren't used at all during training.

**Auxiliary Information:** In order to solve the Zero-Shot Learning issue, some auxiliary information is required because there are no labeled instances of the unseen classes accessible. Such auxiliary information, which may take the form of word embedding’s, semantic information, or descriptions, should provide details about all of the classes that are not visible.

Zero-Shot Learning may be a two-stage process that consists of the following:

**Training:** Information is gathered regarding the designated set of information tests.

**Inference:** The already acquired information is expanded, and the assistant data provided is applied to the unutilized set of classes.

A labeled training set of seen and unseen classes is another component of Zero-Shot Learning. The knowledge from seen classes can be used to unseen classes because both seen and unseen classes are related in a high dimensional vector space called semantic space. **p2**(Kundu, 2023)

Overall, zero-shot learning is a crucial area of research because it enables machine learning models to generalize to previously unseen examples and classes, strengthening their robustness and enabling greater adaptability to practical situations. **p1**(Mumbi, 2022)

## Natural Language Generation

is a subfield of artificial intelligence and natural language processing that focuses on the automatic creation of natural language text or speech by a computer. It can be used for a variety of tasks, such as summarizing data, generating reports, or creating dialogue systems. NLG systems use a combination of machine learning algorithms and linguistic knowledge to generate human-like text or speech. (Wigmore, 2021)

### How NLG works?

NLG is a multi-stage process, with each step further refining the data being used to produce content with natural-sounding language. The six stages of NLG are as follows: (Wigmore, 2021)

* Content analysis
* **Data understanding**
* **Document structuring**
* **Sentence aggregation**
* **Grammatical structuring**
* **Language presentation**

Natural language generation is being used in an array of ways. Some of the many uses include the following: (Wigmore, 2021)

* generating the responses of chatbots and voice assistants such as Google's Alexa and Apple's Siri.
* automating lead nurturing email, messaging, and chat responses.
* personalizing responses to customer emails and messages.
* generating and personalizing scripts used by customer service representatives.
* aggregating and summarizing news reports.

## Text-to-Speech

Text-to-speech (TTS) is a technology that converts written text into spoken words. It is used in many applications such as voice assistants, e-learning, and accessibility for the visually impaired. TTS systems can be integrated into a variety of devices such as smartphones, computers, and smart speakers. There are different TTS engines available, some of which are developed by large tech companies such as Google, Amazon, and Microsoft. They have different features and capabilities, including support for multiple languages and voice options. (“wideo tool,” 2023)

### Text-to-speech (TTS) technology offers several benefits, including:

(“wideo tool,” 2023)

* Accessibility: TTS technology can be used to make written content more accessible to people with visual impairments, learning difficulties, or other disabilities. (“wideo tool,” 2023)
* Convenience: TTS technology can be used to listen to written content while doing other tasks, such as driving or exercising. (“wideo tool,” 2023)
* Efficiency: TTS technology can be used to increase productivity by allowing people to listen to written content at a faster rate than they can read it. (“wideo tool,” 2023)
* Automation: TTS technology can be used in automation systems, such as customer service chatbots, to provide a more natural and human-like conversation experience. (“wideo tool,” 2023)
* Language learning: TTS technology can be used to help people learn a new language by listening to written content in that language. (“wideo tool,” 2023)
* Cost-effective: TTS technology can be an affordable option for creating voice content, compared to hiring human voice actors. (“wideo tool,” 2023)
* Multitasking: TTS technology can be used to listen to written content while doing other tasks, such as cooking, cleaning, or reading. (“wideo tool,” 2023)

### What is the best free text to speech?

(“wideo tool,” 2023)

* Wideo Text to Speech.
* ttsreader.
* ispeech.
* Naturalreaders

## Implementation

### Fluent speech corpus

Link: <https://www.kaggle.com/datasets/tommyngx/fluent-speech-corpus>

Dataset contains 97 speakers saying 248 different phrases. The 248 utterances map to 31 unique intents, that are divided into three slots: action, object, and location.

The goal in preparing this dataset was to provide a benchmark for end-to-end spoken language understanding models.

**/wavs/speakers/**: Contains 16kHz single channel .wav files containing the speech audio recorded by the speakers, files are gathered into directories divided by the randomly generated speaker ID each participant was given.

**/data/speakerdemographics.csv**: A CSV file with the demographics for each speaker. These include: self-reported speaking ability, first language spoken, current language used for work or school, gender, and age range.

/data/traindata.csv, /data/validdata.csv, /data/testdata.csv: A csv file describing each audio file in the training set, validation set, and test set, respectively. Each line contains the following information:

* **path** - Path to the .wav file
* **speakerId** - Anonymized alphanumeric code for the speaker of this audio
* **transcript** - The prompt that the speaker was asked to read
* **action** - One of 'change language', 'activate', 'deactivate', 'increase', 'decrease', 'bring'
* **object** - One of 'none', 'music', 'lights', 'volume', 'heat', 'lamp', 'newspaper', 'juice', 'socks', **'shoes'**, 'Chinese', 'Korean', 'English', 'German'
* **location** - One of 'none', 'kitchen', 'bedroom', 'washroom'

### Open Speech Corpora

Link: <https://github.com/coqui-ai/open-speech-corpora>

A list of open speech corpora for Speech Technology research and development.

### Speech Recognition

Link: <https://realpython.com/python-speech-recognition/#how-speech-recognition-works-an-overview>

Notebook: speech\_recognition\_moh\_1.ipynb

## Implementation

### Fluent speech corpus

Link of data: <https://www.kaggle.com/datasets/tommyngx/fluent-speech-corpus>

The dataset contains 97 speakers saying 248 different phrases. The 248 utterances map to 31 unique intents, that are divided into three slots: action, object, and location.

The goal in preparing this dataset was to provide a benchmark for end-to-end spoken language understanding models.

**/wavs/speakers/**: Contains 16kHz single channel .wav files containing the speech audio recorded by the speakers, files are gathered into directories divided by the randomly generated speaker ID each participant was given.

**/data/speakerdemographics.csv**: A CSV file with the demographics for each speaker. These include: self-reported speaking ability, first language spoken, current language used for work or school, gender, and age range.

**/data/traindata.csv, /data/validdata.csv, /data/testdata.csv**: A csv file describing each audio file in the training set, validation set, and test set, respectively. Each line contains the following information:

* **path** - Path to the .wav file
* **speakerId** - Anonymized alphanumeric code for the speaker of this audio
* **transcript** - The prompt that the speaker was asked to read
* **action** - One of 'change language', 'activate', 'deactivate', 'increase', 'decrease', 'bring'
* **object** - One of 'none', 'music', 'lights', 'volume', 'heat', 'lamp', 'newspaper', 'juice', 'socks', 'shoes', 'Chinese', 'Korean', 'English', 'German'
* **location** - One of 'none', 'kitchen', 'bedroom', 'washroom'

### Open Speech Corpora

Link of data: <https://github.com/coqui-ai/open-speech-corpora>

A list of open speech corpora for Speech Technology research and development.

### Speech Recognition

Link of tutorial: <https://realpython.com/python-speech-recognition/#how-speech-recognition-works-an-overview>

Notebook: speech\_recognition.ipynb

The speech recognition model is used to recognize and transcript the user’s speech using Google Web Speech API

### Zero-Shot Model

Link: <https://huggingface.co/facebook/bart-large-mnli>

Notebook: natural language understanding.ipynb

Facebook’s bart-large-mnli model was used. Which is a checkpoint for bart-large after being trained on the MultiNLI (MNLI) dataset. (Radford \*, 2020)

bart-large BART (large-sized model): BART is a transformer encoder-decoder (seq2seq) model with a bidirectional (BERT-like) encoder and an autoregressive (GPT-like) decoder. BART is pre-trained by (1) corrupting text with an arbitrary noising function, and (2) learning a model to reconstruct the original text. (Brown, 2020; Radford \*, 2020)

MultiNLI (MNLI (Multi-Genre Natural Language Inference)) dataset: MultiNLI corpus is a crowd-sourced collection of 433k sentence pairs annotated with textual entailment information. The corpus served as the basis for the shared task of the RepEval 2017 Workshop at EMNLP in Copenhagen. (Brown, 2020; Radford \*, 2020)

The model is used to classify the order from the transcripted speech and then gives the probability for the three classes ("conversational", "question", "command") (Brown, 2020; Radford \*, 2020)

Models used:

* <https://huggingface.co/facebook/bart-large-mnli>
* <https://huggingface.co/datasets/multi_nli>
* <https://huggingface.co/facebook/bart-large>
* <https://joeddav.github.io/blog/2020/05/29/ZSL.html>

### Conversational model

Link: <https://huggingface.co/facebook/blenderbot-400M-distill>

Notebook: natural language understanding.ipynb

Facebook’s blenderbot-400M-distill model was used. Which is a version of BlenderBot 2.0 that uses 400M parameters, a chatbot that can simultaneously build long-term memory it can continually access, search the internet for timely information, and have sophisticated conversations. (Roller, 2020)

### Question Answering

Notebook: natural language understanding.ipynb

yanekyuk/bert-uncased-keyword-extractor model which is a fine-tuned version of bert-base-uncased. Is used to extract the keywords in the question.

Natural Language Toolkit (nltk) Is used for Part-of-speech tagging

**Wikipedia Python library is used to** access and parse data from Wikipedia to answer the questions

Keyword extractor: <https://huggingface.co/yanekyuk/bert-uncased-keyword-extractor>

### Storytelling

Using pranavpsv/gpt2-genre-story-generator which is GPT2 Genre Based Story Generator. Used to generate stories based on user input genre.

Model link: <https://huggingface.co/pranavpsv/gpt2-genre-story-generator>

### Natural Language Generation

Generative Pre-trained Transformer 2 (GPT-2) model is used to generate text output

Model link: <https://huggingface.co/gpt2>

### Text To Speech

gTTS (Google Text-to-Speech), a Python library and CLI tool to interface with Google Translate's text-to-speech API. Is used to convert the answers.

Library link: <https://pypi.org/project/gTTS/>

## References

Activity Matters, 2020. . TechTarget.

Amos, D., 2018. The Ultimate Guide To Speech Recognition With Python.

Aureus Analytics, 2021. . Aureus Anal.

Brown, T., 2020. Language Models are Few-Shot Learners.

Gupta, A., 2020. Spoken Language Understanding : On Building Spoken Dialogue Systems.

Kundu, R., 2023. The Essential Guide to Zero-Shot Learning [2023].

McTear, M., 2010. Spoken Dialog System topics. ScienceDirect.

Mumbi, W., 2022. How to Implement Zero-Shot Classification using Python.

Nisar, S., 2018. Introduction to Dialogue Systems (Part 1).

NLU - beginner’s guide, 2022.

Papangelis, A., 2020. PLATO DIALOGUE SYSTEM∗ : A FLEXIBLE CONVERSATIONAL AI RESEARCH PLATFORM.

Radford \*, A., 2020. Language Models are Unsupervised Multitask Learners.

Roller, S., 2020. Recipes for building an open-domain chatbot.

Wen, T.-H., 2015. Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems.

wideo tool, 2023.

Wigmore, I., 2021. natural language generation (NLG).

wlabu, may, 2020. Natural Language Understading: getting started.

Zue, V., Seneff, S., 2008. Spoken Dialogue Systems.