MIRPR Project
Vacation Chatbot

General Outline

- Web app
  - Frontend in Elm
  - Backend in Node.js
  - Through Websockets
  - AI part in Python
    - Used by Node.js when replying to messages.
- Intelligent application in
  - Natural Language Processing

Representation of holiday locations

- Each location contains
  - The name of the location
  - Generic data about the location
  - And, relevant for the Intelligent part
    - Sets of tags with a corresponding degree of certainty.
    - The degree can be thought of as
      - “We’ve asked 1000 tourists if they consider the location to be X.”
      - X is a trait, such as “noisy”, “expensive”, “near the sea”.

Process

- Obtain the user’s sentence
- Somehow parse information out of it
- Use the obtained information to make suggestions

Agent 001

- Has three extraction approaches, each imposing more rigidity but allowing more precise suggestions
- Relies on
  - Natural Language Toolkit (https://www.nltk.org/)
- Stanford CoreNLP (https://stanfordnlp.github.io/CoreNLP/)

Approach 1 - flat tags

- Segment input into words
- POS tag
- Lemmatize
- Extract words of interest

Approach 1 - Flat tags

- Upsides
  - Conveys information even if input is chaotic
- Downsides
  - Hard to express very specific preferences

Approach 2 - ranked tags

- Parse into sentence tree
- POS-tag
- Lemmatize
- Extract rules from a supposed sentence structure

Approach 2 - ranked tags

- Upsides
  - Very easy to express complex rules
  - Can be really specific about holiday locations
- Downsides
  - Very opinionated about sentence structure
  - Incomprehensible sentence parts yield useless rules

Approach 3 - tags with dependency graph

- Parse sentence into dependency graph
- Traverse the nodes of the graph
  - For each node, remember the lemmatized form of the word
  - If the word is modified by a negation, negate the extracted rank.

Approach 3 - tags with dependency graph

- Upsides
  - Conveys information even if input is very chaotic
  - Can express negation of holiday characteristics
- Downsides
  - Cannot express more complex rules

Agent 002

- Classifies a query based on the intent it represents
- When the intent is confirmed a predefined message can be sent

Process

1. Data preparation

    1.1. Data cleaning

    1.2. Lemmatization

2. Encoding

    2.1. Input Encoding

    2.2 Output Encoding

3. Train and validation set
4. Results

1. Data preparation

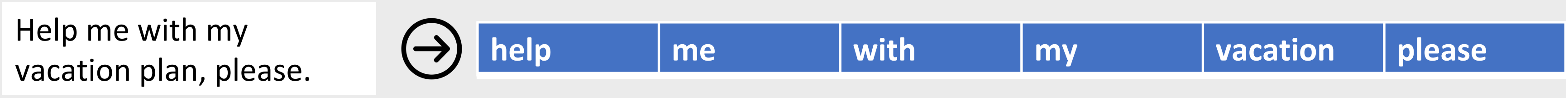
Simplification of data by:

- Building a set of unique intents
- Tagging sentences with their corresponding intents
- Building dataframes for sentences and intents

Sentences	Intents
Yo, dawg, what's up	Greet
I want to go on a trip	Vacation_Search
My last vacation was shit	Vacation_Review
Show me what you got	Vacation_List

1.1. Data cleaning

- Remove punctuations and special characters
- Break sentences into words
- Lowercase words and apply lemmatization



Help me with my vacation plan, please.

→

help	**me**	**with**	**my**	**vacation**	**please**

1.2. Lemmatization

- **NLTK lemmatizer**

  - easy to use python library

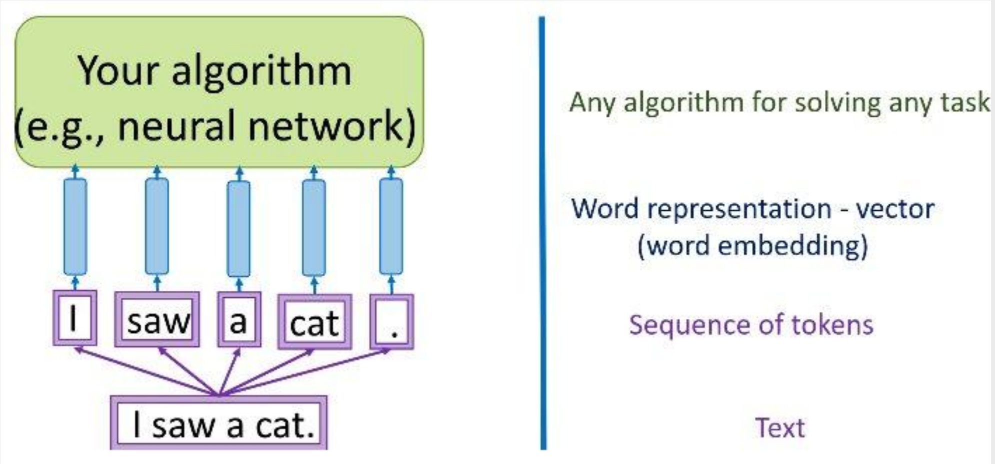
  - large, freely and publicly available english lexical database

  - based on WordNet’s built-in morphy function

Pos	Suffix	Ending
Noun	”s”	””
Adjective	”est”	”e”
Verb	”ies”	”y”

2. Encoding

- Process of turning a set of categorical features in raw (or preprocessed) text to a series of vectors
- 2 phases: Input and Output Encoding

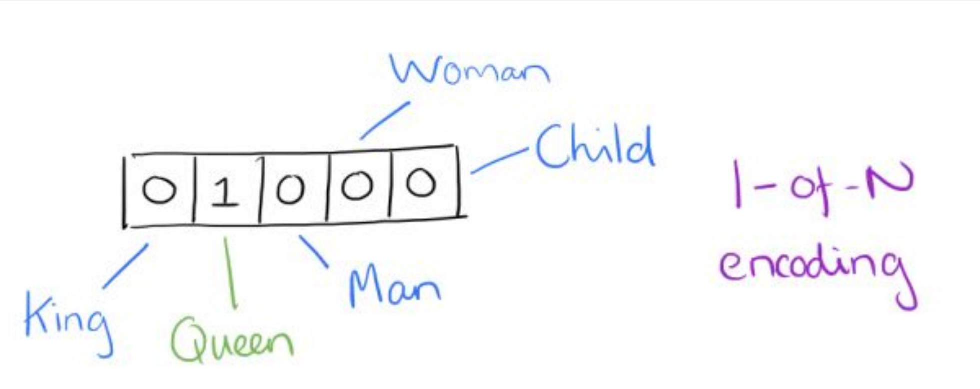


2.1 Input encoding

- Cleaned data is passed to the Keras tokenizer
- The use of padding to make tokenized words the same length

2.2 Output encoding

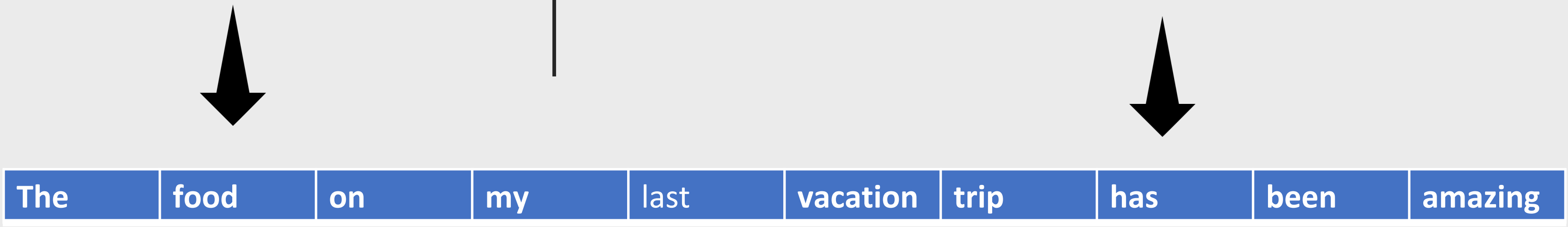
- Apply the same keras tokenize class to the output intents
- One hot encoding of the intents



3. Train and validation set

- Split the dataset into training and validation set (80% training and 20% validation set)
- Build the model and train it on the data
- Bidirectional Gated Recurrent Unit neural network

  120 epochs, batch size of 32



The	food	on	my	last	vacation	trip	has	been	amazing

4. Results

- Achieved 75% training accuracy and 70% validation accuracy
- Some drawbacks: lack of data, hyperparameters not fully optimized
- Confusion due to similarity of intents between vacation_search and vacation_list

Agent 001 + Agent 002

- We first find the intent of the user’s sentence (agent 002)
- We then extract the tags (agent 001)
- If the user’s intent is to search vacation we use the tags to recommend a place from the database

Demo time!