BABEŞ BOLYAI UNIVERSITY, CLUJ NAPOCA, ROMÂNIA
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

Holiday Helper

– MIRPR report –

Team members

Moisil Adriana, Informatică Română, 231

2019

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

1.1 What? Why? How?

Motivate and abstractly describe the problem you are addressing and how you are addressing it.

- What is the (scientific) problem?

  The problem tackles natural language processing, natural language understanding and image classification. The user is going to describe in natural language the basic requirements for their vacation: location, number of rooms, number of kids, dates, or they can provide an image to substitute for a location (and based on it, the AI algorithm will deduce which type of destination is most suitable for them).

- Why is it important?

  It is a holiday helper assistant, meaning that it will help the users with a general idea of their vacation needs to make a first impression of the available accommodation options.

- What is your basic approach?

  We are building a prototype demonstrating the basic functionalities of the app. This includes recognizing a few filters from the user input, and using the Booking API together with our AI algorithm to give hotel recommendations.

Chapter 2

Scientific Problem

2.1 Open text analysis

The first part of the application's flow is extracting the user's requirements from open text. Giving a friendlier experience is the motivation behind building it as a chatbot, so that's why we decided to allow this type of input. We're using the nltk and spacy libraries in order to extract all the information we need:

- location
- time period
- number of rooms
- number of children

2.1.1 Extract number of rooms

For extracting the number of rooms, we build a custom grammar (called COUNT) that is used when parsing the input (using nltk.nltkRegexpParser):
COUNT:{

<CD><NNS>|

<CD><JJ>*<NNS>|

<CD>(<JJ><CC>)*<JJ><NNS>|

<CD><NN>|

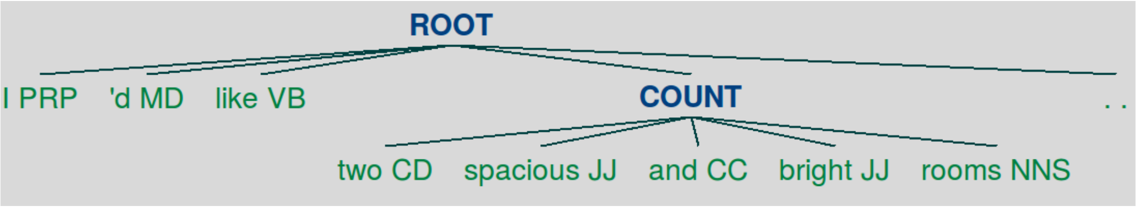
<CD><JJ>*<NN>|

<CD>(<JJ><CC>)*<JJ><NN>

}

The custom grammar can recognize all the groups that start with a cardinal, end with a noun (both singular and plural) and might contain adjectives and conjunctions in between (e.g. 'two spacious and bright rooms').

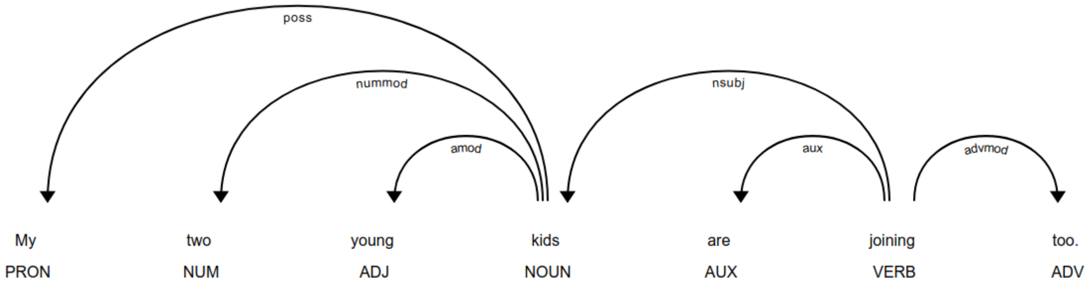
Figure 2.1: Extract number of rooms.



2.1.2 Extract number of children

We tried implementing a different approach for extracting the number of children. We make use of the tokens' dependencies in order to find the cardinal that is associated with the noun children (or similar nouns).

Figure 2.2: Extract number of children.



2.2 Image classifier

We built an image classifier using Keras, which is TensorFlow's high-level API for building and training deep learning models. The user can upload an image (for demonstration purposes only, the current input images are limited to forests, sand beaches and snow mountains) to which the application will offer suggestion of destinations that look similar to what the user wants. The image will be rejected and the application will ask the user again for their desired location (either through an another image or through text) if the probability for the image to fit in one of the three categories is too low.

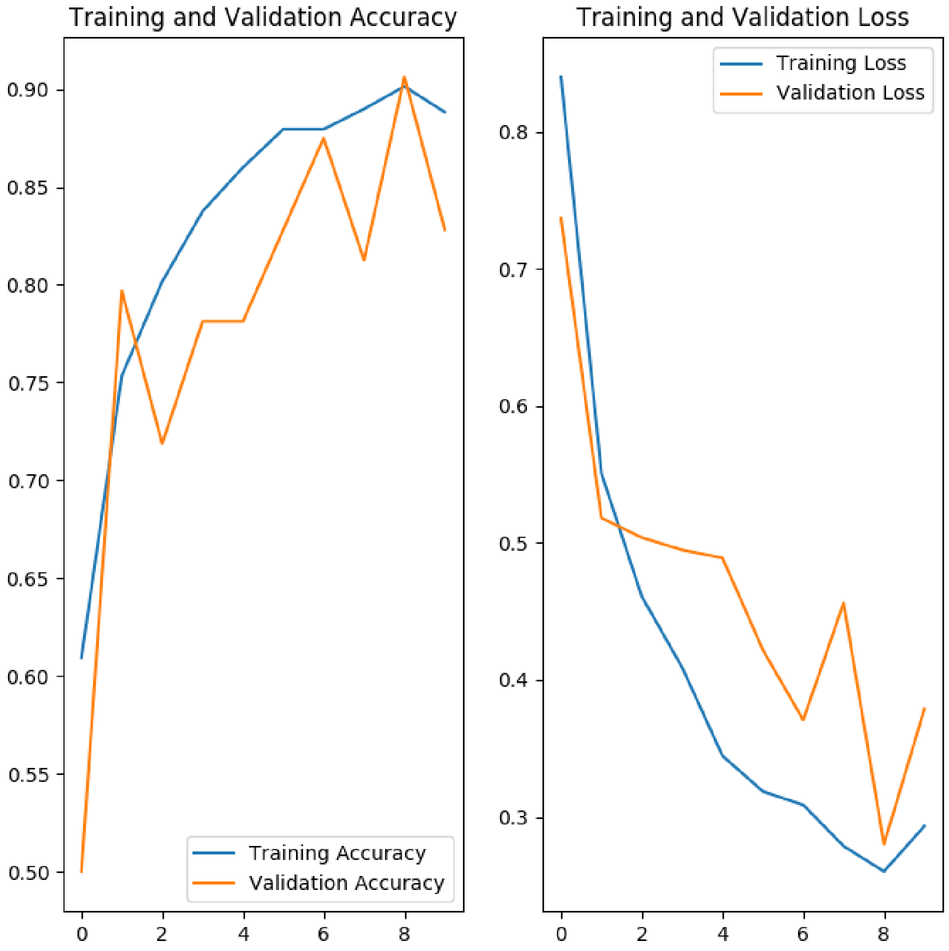
Table 2.1: The parameters used for training the Keras model used for image classification.

Parameter	**Value**
Batch size	64
Number of epochs	10
Image height	150
Image width	150
Learning rate	0.0001

Table 2.2: The layers of the Keras model used for image classification.

Layer type	**Output Shape**	**Param #**
Conv2D	(None, 150, 150, 16)	448
MaxPooling2D	(None, 75, 75, 16)	0
Conv2D	(None, 75, 75, 32)	4640
MaxPooling2D	(None, 37, 37, 32)	0
Conv2D	(None, 37, 37, 64)	18496
MaxPooling2D	(None, 18, 18, 64)	0
Dropout	(None, 18, 18, 64)	0
Flatten	(None, 20736)	0
Dense	(None, 512)	10617344
Dense	(None, 3)	1539

Figure 2.3: Training and Validation values for the final model.



2.3 Reviews analyzer

Apart from the Image Classifier, the tool contains what we call a "Reviews analyzer". Given the raw user reviews extracted from Booking (both the negative and the positive ones), it does the following:

- Clusters data and extracts the topics for each cluster
  - Preprocesses the raw text (remove stopwords, punctuation, spelling correction etc.; no stemmer or lemmatizer is being used here)
  - Encodes the data using Tensorflow's Universal Sentence Encoder, which, for each input text, returns an array of 512 floating points
  - Clusters the data using the clustering algorithms implementations offered by the scikit-learn

  and hdbscan libraries. The clustering methods that were tested are KMeans, DBSCAN, HDBSCAN and Agglomerative, and based on the results, we moved forward with the last one.

  - Extract the keywords for each cluster (Bag of Words)

- Creates word clouds (the preprocessing of the raw text in this case is different as the words are lemmatized WordNetLemmatizer from the nltk library)

- Computes the overall sentiment (a value between -1 and 1, -1 representing a negative sentiment and 1 representing a positive one) for each hotel (using the textblob library)

Chapter 3

State of art/Relate work

Google Assistant

- Virtual Assistant (not for travel planning)
- Stand alone app

Hello Hipmunk

- Virtual Travel planning assistant
- Integrated with Facebook Messenger, Slack, Skype
- Available to group chats
- Includes flights and hotels
- Deals only with traveling (doesn't help with dinner or entertainment)

Mezi

- Stand alone app
- Includes flights, hotels and restaurant reservations
- Provides interactive output (buttons to book a flight)

SnapTravel

- Virtual Travel Planning Assistant
- Integrated with Facebook Messenger and Whatsapp

HelloGBye

- Virtual Travel Planning Assistant
- Text and voice recognition
- Provides interactive output (buttons to book a flight)

Holiday Helper

- Stand alone app
- Includes only hotels, using the Booking API
- The information is asked for in multiple rounds, not all at once, unless the user provides more (e.g. both the location, number of people and number of rooms) at a time.
- Text recognition only

	Hello Hipmunk	Mezi	SnapTravel	HelloGBye	Holiday Helper
Stand alone app	NO	YES	YES	YES	YES
Messenger	YES	NO	YES	NO	NO
Whatsapp	NO	NO	YES	NO	NO
Slack	YES	NO	NO	NO	NO
Skype	YES	NO	NO	NO	NO
Hotels	YES	YES	YES	YES	YES
Restaurants	NO	YES	NO	NO	NO
Flights	YES	YES	YES	YES	NO

Chapter 4

Proposed approach

We're using a client-server model for this solution in order to make it portable. When it comes to the technologies we used, the backend is a Flask server while the frontend is built using Django. All the logic takes place on the server, so the client only sends the data to it and waits for the results. As a result, the application can run on any device (computer, smartphone), as long as it's connected to the Internet (and the server is running).

Chapter 5

Application (numerical validation)

Explain the experimental methodology and the numerical results obtained with your approach and the state of art approache(s).

Try to perform a comparison of several approaches.

Statistical validation of the results.

5.1 Methodology

- What are criteria you are using to evaluate your method?
- What specific hypotheses does your experiment test? Describe the experimental methodology that you used.
- What are the dependent and independent variables?
- What is the training/test data that was used, and why is it realistic or interesting? Exactly what performance data did you collect and how are you presenting and analyzing it? Comparisons to competing methods that address the same problem are particularly useful.

5.2 Data

5.2.1 Image classifier

The data used for training the Image Classifier was collected from Google Images using a script, obtaining around 800 entries for each section (forest, sand beach and snow mountain). After filtering, the remaining number is around 450 for the first two categories, and 680 for the last one, which were later divided in two sections: training and validation. In total, there were used 1496 images for training and 150 for validation.

5.3 Results

One way to improve the results of our model used for image classification would consist of gathering more training data for all the three existing categories.

Chapter 6

Conclusion and future work

6.1 Open text analysis

6.2 Image classifier

Our current dataset for training and validation is quite small, so one way to improve this component is gathering more images for the three categories in order to train the model better. It would be also worth looking into ways of adjusting the ImageDataGenerator's parameters for the training and validation images so we can generate more data from the current one.

6.3 Reviews analyzer

As mentioned before, we consider the cluster's keywords to be their most frequent words. This might not always lead to the best result (e.g. a cluster with positive reviews where the main topic is breakfast, but we don't know what other words are associated with it: it could be 'large breakfast' or 'delicious breakfast' etc.), so another option would be extracting pairs of two words as keywords.