Chatbot

CSC 466 Spring ‘21 Project

by Nithya Chandran, Casey Ng, and Brandon Chan

The Goal

Create a retrieval-based chatbot to help users (CS students) decide what classes to take

The Process

1. Create a dataset with expected user inputs and responses
2. Create training set with unigram, bigram, and trigram approaches
3. Train a “special recurrent neural network (LSTM)” to predict correct responses to user input
4. Compare chatbot behavior and performance from different training set approaches
5. Have fun with Chatbot!

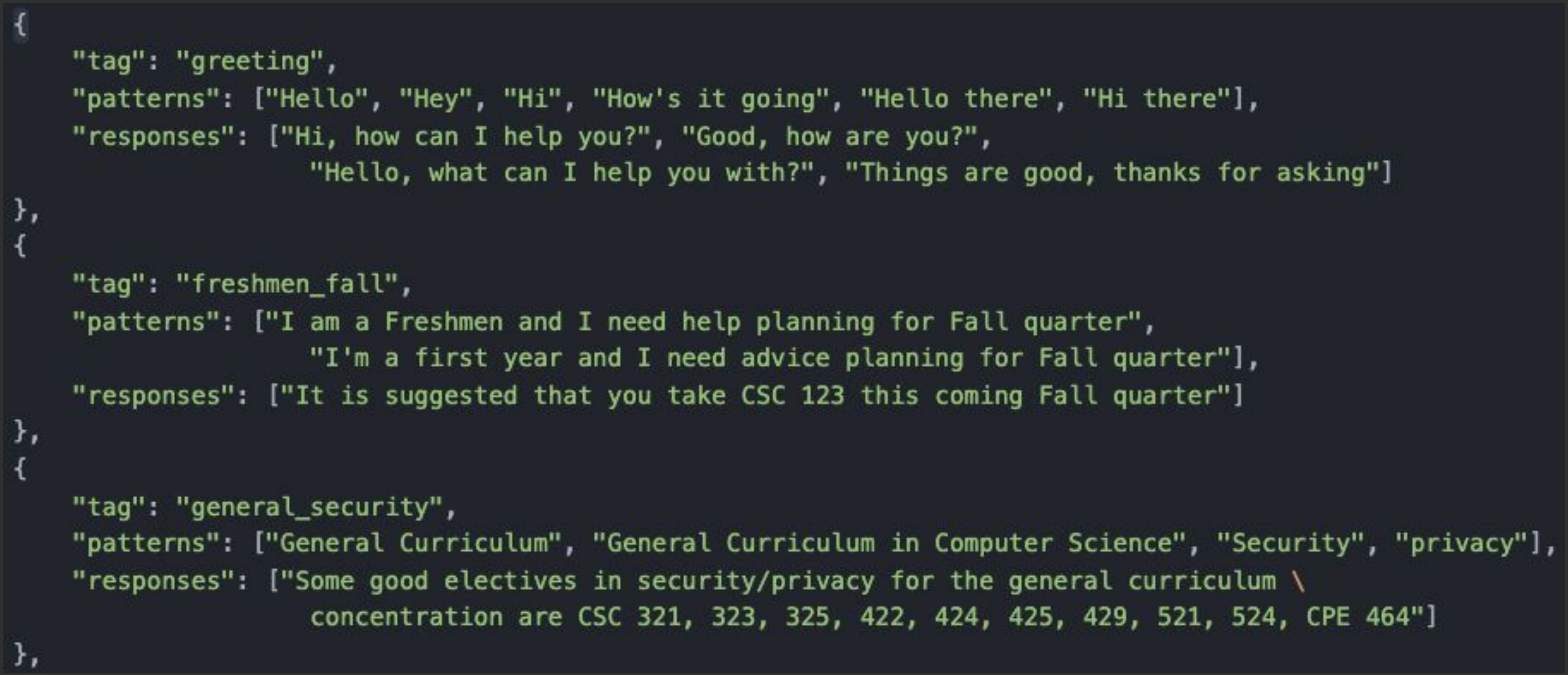
The Data

The data was generated from scratch. How?

We have a total of 80 intent categories which includes:

- General conversation
- Class recommendations based on:
    - Class standing
    - Quarter
    - Concentration
    - Interests

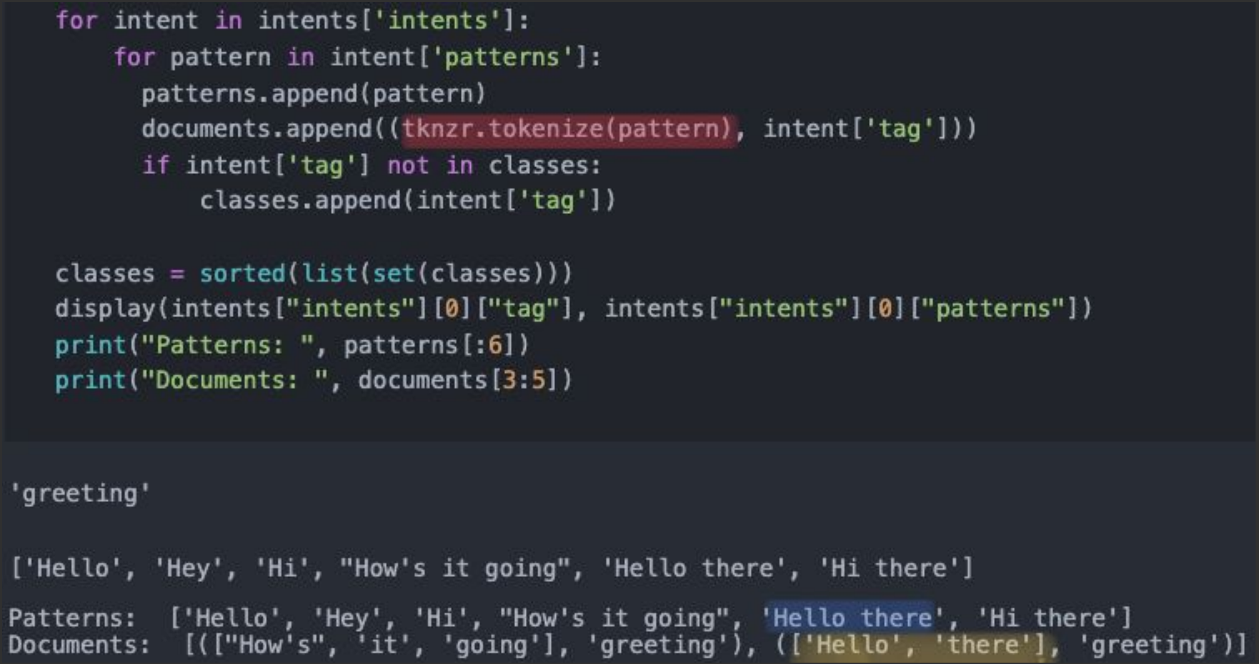
Snippet of the Data



```
{
    "tag": "greeting",
    "patterns": ["Hello", "Hey", "Hi", "How's it going", "Hello there", "Hi there"],
    "responses": ["Hi, how can I help you?", "Good, how are you?",
                  "Hello, what can I help you with?", "Things are good, thanks for asking"]
},
{
    "tag": "freshmen_fall",
    "patterns": ["I am a Freshmen and I need help planning for Fall quarter",
                  "I'm a first year and I need advice planning for Fall quarter"],
    "responses": ["It is suggested that you take CSC 123 this coming Fall quarter"]
},
{
    "tag": "general_security",
    "patterns": ["General Curriculum", "General Curriculum in Computer Science", "Security", "privacy"],
    "responses": ["Some good electives in security/privacy for the general curriculum \
                  concentration are CSC 321, 323, 325, 422, 424, 425, 429, 521, 524, CPE 464"]
},
```

Data Preprocessing

- Tokenized patterns for each intent
  - “Hello there” -> [“Hello”, “there”]



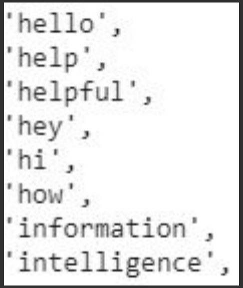
```
for intent in intents['intents']:
    for pattern in intent['patterns']:
      patterns.append(pattern)
      documents.append((tknzr.tokenize(pattern), intent['tag']))
      if intent['tag'] not in classes:
          classes.append(intent['tag'])

classes = sorted(list(set(classes)))
display(intents["intents"][0]["tag"], intents["intents"][0]["patterns"])
print("Patterns: ", patterns[:6])
print("Documents: ", documents[3:5])
```

```
'greeting'

['Hello', 'Hey', 'Hi', "How's it going", 'Hello there', 'Hi there']

Patterns:  ['Hello', 'Hey', 'Hi', "How's it going", 'Hello there', 'Hi there']
Documents:  [(["How's", 'it', 'going'], 'greeting'), (['Hello', 'there'], 'greeting')]
```

Training Data

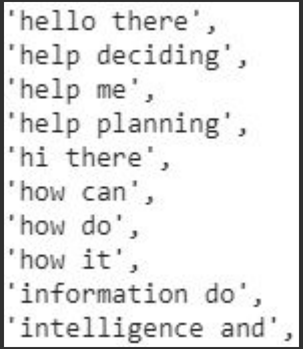
- Convert array of patterns to a matrix of unigram counts, bigram and trigram counts using CountVectorizer()

Unigram Feature Examples



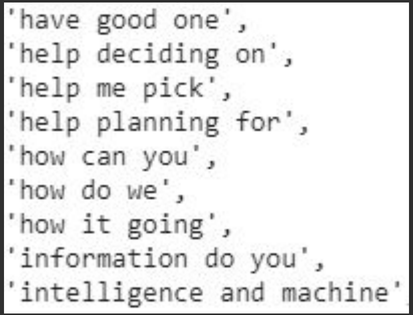
```
'hello',
'help',
'helpful',
'hey',
'hi',
'how',
'information',
'intelligence',
```

Bigram Feature Examples



```
'hello there',
'help deciding',
'help me',
'help planning',
'hi there',
'how can',
'how do',
'how it',
'information do',
'intelligence and',
```

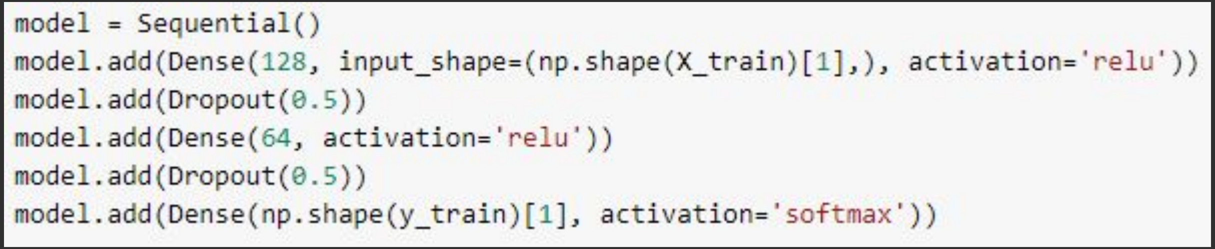
Trigram Feature Examples



```
'have good one',
'help deciding on',
'help me pick',
'help planning for',
'how can you',
'how do we',
'how it going',
'information do you',
'intelligence and machine'
```

The Models

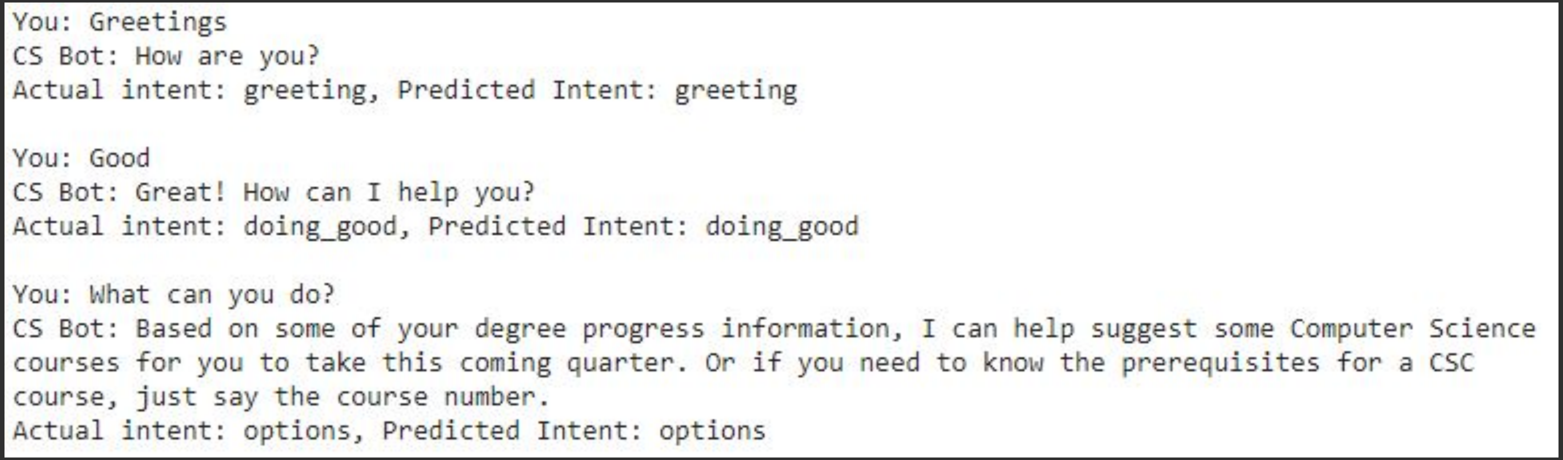
- Three Keras Sequential models
  - One model for each training dataset (unigram, bigram, trigram)
- Layers:
  - Three Dense Layers
  - Two Dropout



```
model = Sequential()
model.add(Dense(128, input_shape=(np.shape(X_train)[1],), activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(np.shape(y_train)[1], activation='softmax'))
```

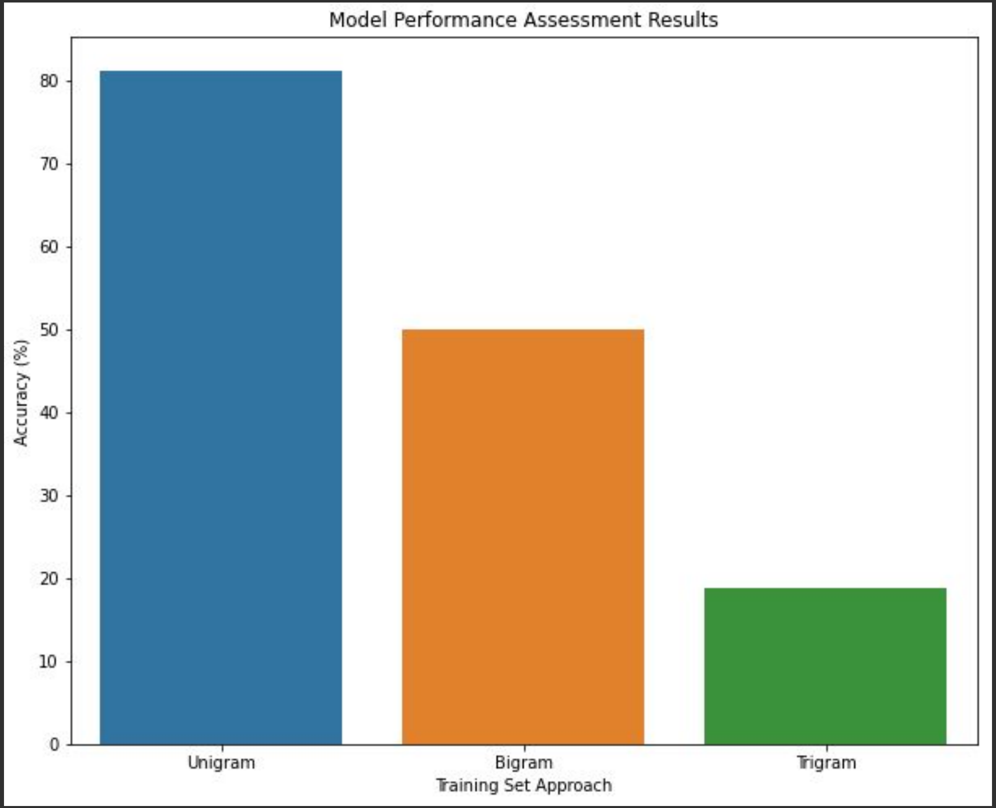
Model Performance Assessment

- Have the same conversation with each of the three models
- Performance is measured by accuracy
    - Accuracy = Number of correct response / Total number of responses



```
You: Greetings
CS Bot: How are you?
Actual intent: greeting, Predicted Intent: greeting

You: Good
CS Bot: Great! How can I help you?
Actual intent: doing_good, Predicted Intent: doing_good

You: What can you do?
CS Bot: Based on some of your degree progress information, I can help suggest some Computer Science
courses for you to take this coming quarter. Or if you need to know the prerequisites for a CSC
course, just say the course number.
Actual intent: options, Predicted Intent: options
```

Model Performance Assessment Results



Potential Additions

- Filter out stopwords in patterns when training model (can be added to CountVectorizer)
- Lemmatize patterns
  - Securities –> security
  - Graphics –> graphic
- TF-IDF training set
- Remembering previous input to improve current responses
  - Specific class recommendations based on classes taken, standing and interests

Demo