

# **CLASSIFYING QUESTIONS**

**SHRI SHRUTHI SHRIDHAR**

**SOWMYA S SUNDARAM**

# PROBLEM STATEMENT

- Identify the type of question being asked by the user with these categories - Who, What, When, Affirmation (Yes / No) or Unknown.

# PROBLEM STATEMENT

- Identify the type of question being asked by the user with these categories - Who, What, When, Affirmation (Yes / No) or Unknown.
- Build an interface to display it to the user, and validate your model's response with that of the user's actual intent.

# PROBLEM STATEMENT

- Identify the type of question being asked by the user with these categories - Who, What, When, Affirmation (Yes / No) or Unknown.
- Build an interface to display it to the user, and validate your model's response with that of the user's actual intent.
- Show analytics of performance

# PROBLEM STATEMENT

- Identify the type of question being asked by the user with these categories - Who, What, When, Affirmation (Yes / No) or Unknown.
- Build an interface to display it to the user, and validate your model's response with that of the user's actual intent.
- Show analytics of performance
- Improve performance dynamically

# DATASET ANALYSIS

- No of samples – 1483
- Distribution
- Missing class labels – taken as unknown

Category	Number
What	574
Who	371
When	89
Affirmation	103
Unknown	342

# CHALLENGES

- Key challenge is differentiating affirmation and unknown
- If we take conventional word2vec approaches all question words such as who, what, when, why, how etc may have similar values. Hence classifying between them might be difficult.
- Class imbalance may make the model weak for affirmation

# APPROACHES ATTEMPTED

- Multiple Classifiers after applying mean of glove
- Using first two words of sentence and building glove
- Applying over-sampling to balance training
- Rules based on Stanford parse trees



# APPROACH - 1

- Multiple Classifiers after applying mean of glove
- After splitting dataset into 80-20% for training and test, we evaluated various models

Classifier	Test Accuracy
Logistic Regression	0.68
SVM	0.67
Random forest classifier	0.52
Multi Layer Perceptron	0.64

# APPROACH - 2

- Multiple Classifiers after applying mean of glove on first 2 words
- After splitting dataset into 80-20% for training and test, we evaluated various models

Classifier	Test Accuracy
Logistic Regression	0.89
SVM	0.89
Random forest classifier	0.74
Multi Layer Perceptron	0.91

## **APPROACH - 3**

- Sampled all datasets equally to minority class
- Performed badly, all accuracies  $\sim 0.2$

## **APPROACH - 4**

- Wrote rules on POS tags
- Moderate performance – human intensive approach

# POST MODEL BUILD

- Chose multi layer perceptron with 2 hidden layers – 150 and 10 nodes
- Trained on entire dataset before launching interface

# INTERFACE

- Desktop interface
- Trained on entire dataset before launching interface
- Functionalities
  - Query Model
  - Check Model against User Intent
  - See statistics of queries in this session
  - Retrain model

# TECH DETAILS

- Python 3
- Sklearn
- NumPy
- Tkinter
- Gensim

# SCREENSHOTS

A screenshot of a Tkinter window titled "tk". The window has a light blue title bar with standard window controls (minimize, maximize, close). The main content area is light gray and contains three input fields stacked vertically, each with a label to its left: "Question", "Type", and "User Intent". Below these fields are four buttons arranged in two groups. The first group on the left contains "Compute" and "Check" buttons. The second group on the right contains "Analytics" and "Re-Train" buttons.

Question	<input type="text"/>		
Type	<input type="text"/>		
User Intent	<input type="text"/>		
Compute	Check	Analytics	Re-Train

# SCREENSHOTS

## Compute

The screenshot shows a web application window titled "tk". The interface includes a form with the following elements:

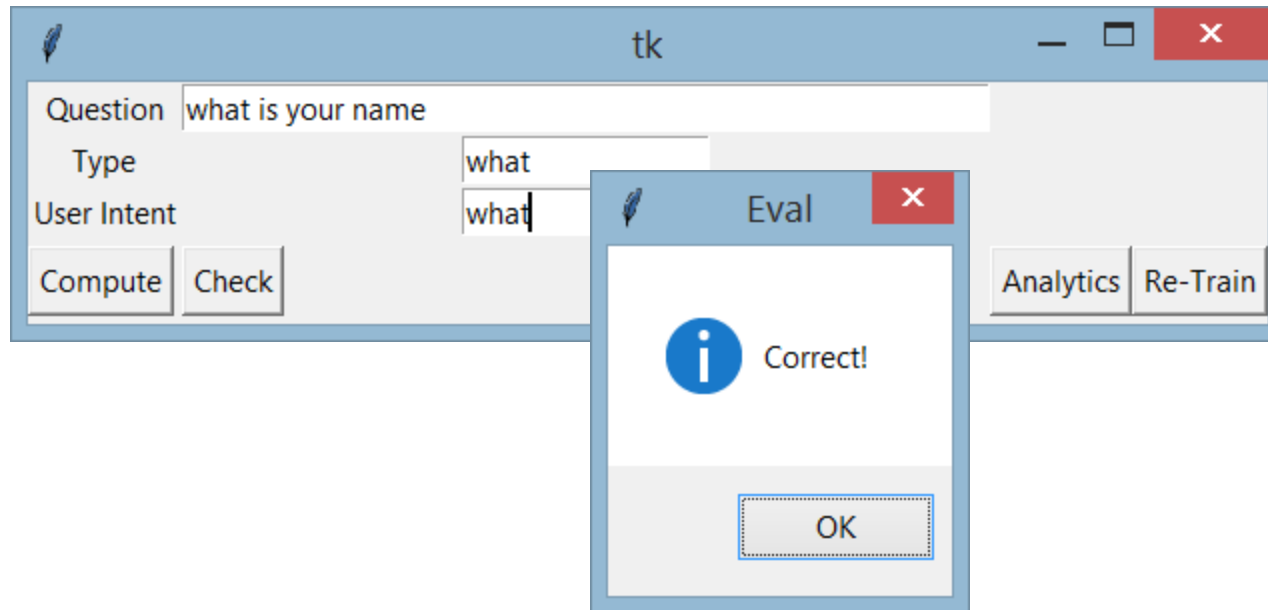
- Question:** A text input field containing "what is your name".
- Type:** A dropdown menu with "what" selected.
- User Intent:** An empty text input field.
- Buttons:** "Compute", "Check", "Analytics", and "Re-Train".

The "Compute" button is highlighted, indicating it is the active action.



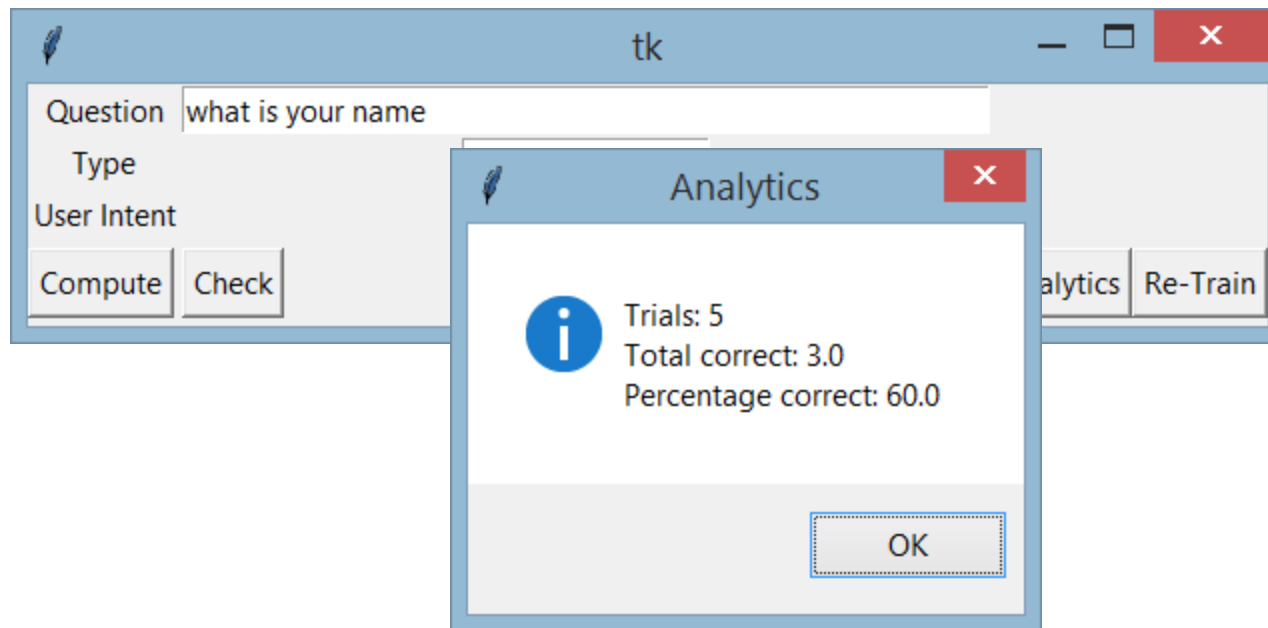
# SCREENSHOTS

## Check



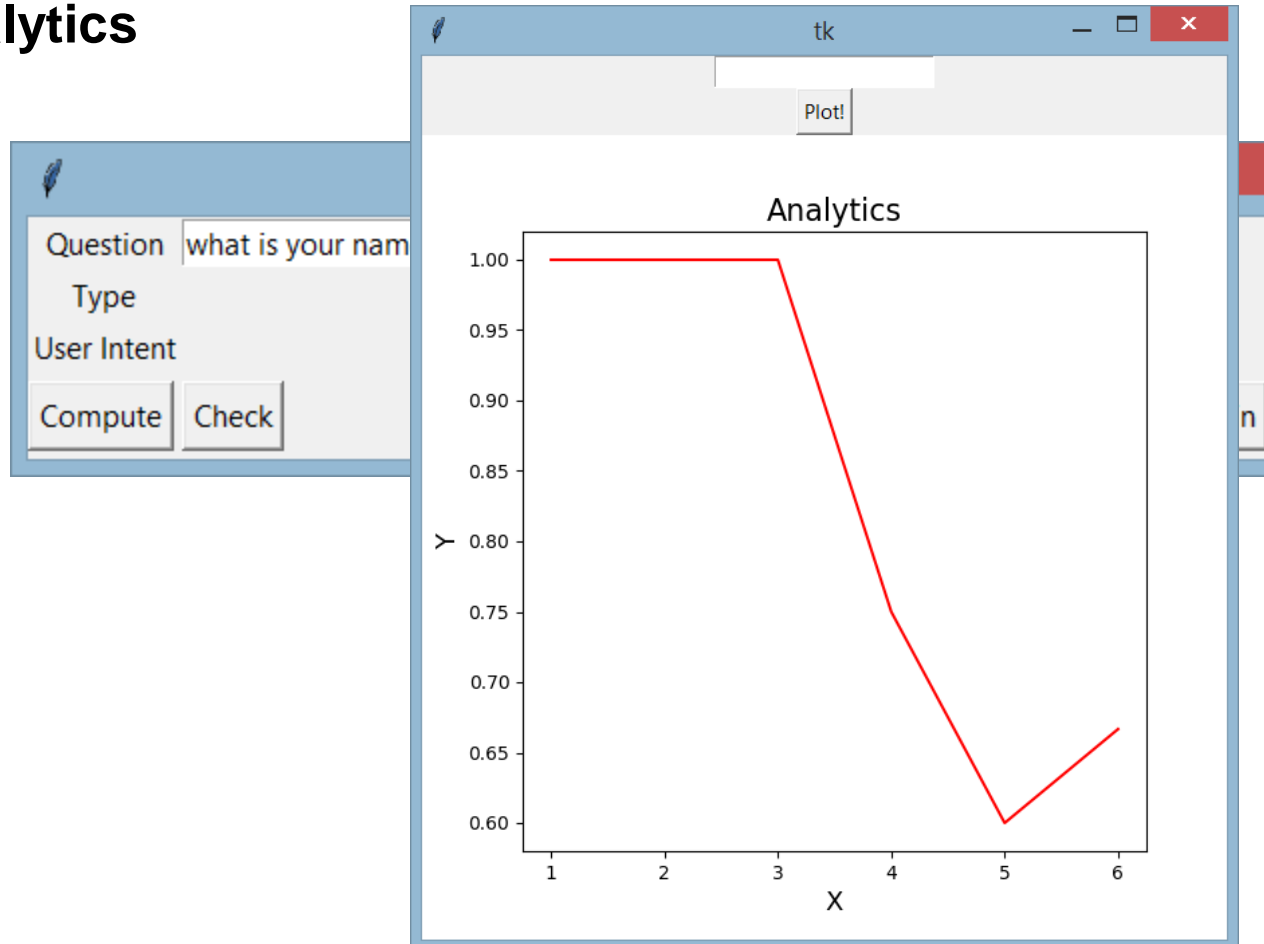
# SCREENSHOTS

## Analytics



# SCREENSHOTS

## Analytics



# FUTURE WORK

- Experiments with doc2vec
- Partially rule based and partially trained
- Associate a control logic for re-training
- With more data – deep architectures can be trained

# APPLICATIONS

- Chatbot pre-processing
- Automated feedback design