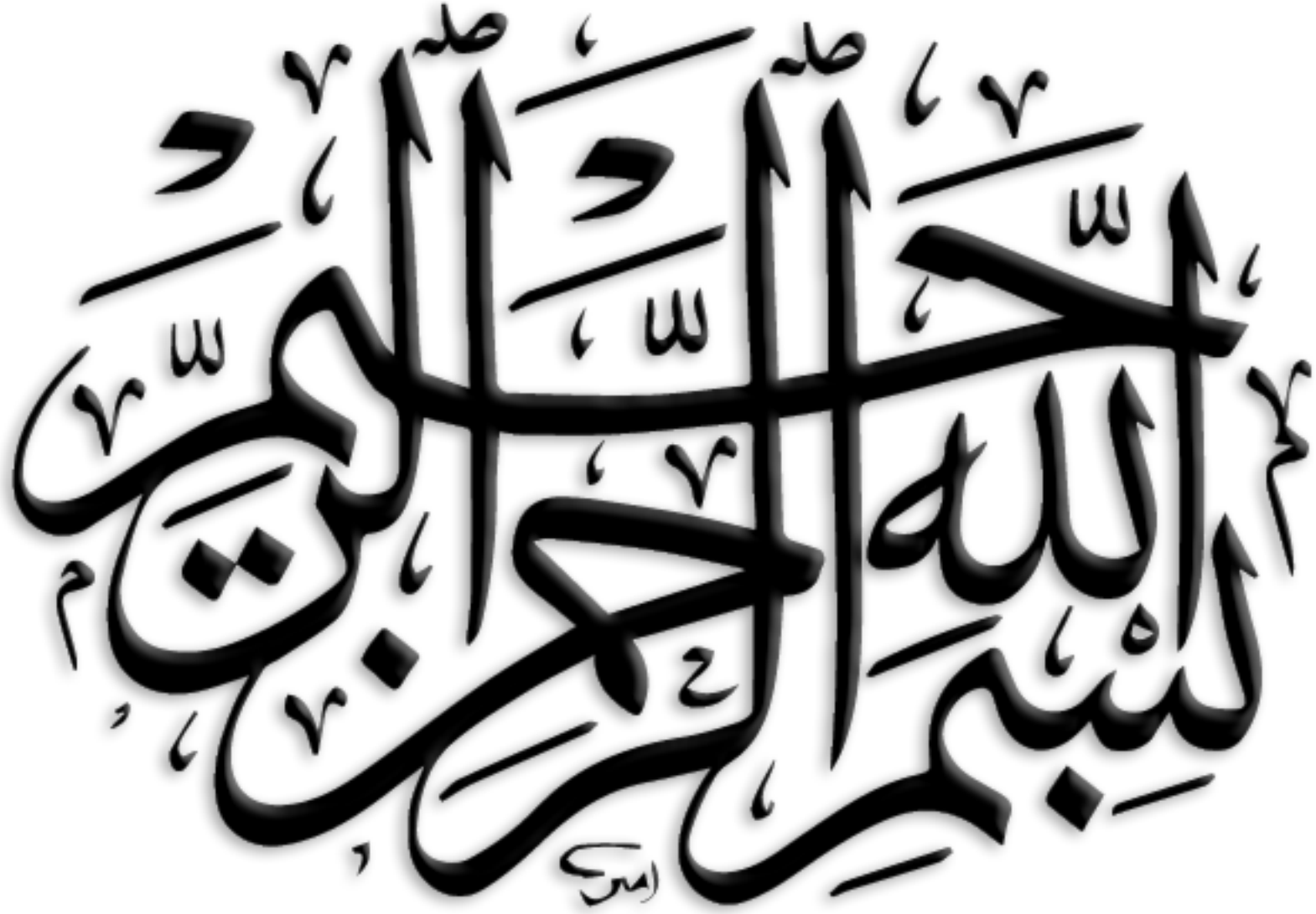

بسم الله الرحمن الرحيم

Customer support ChatBot

Project

SCAN ME

October 2024

Generative Ai ONL1_AIS2_S9e

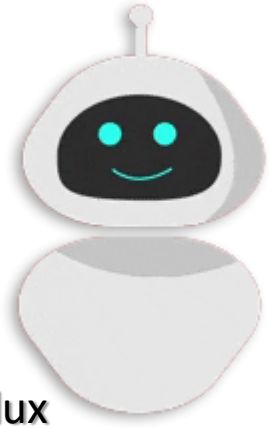
Agenda

Project team:

Name	Initials
Khaled El Sherif	K.E.
Mohamed Yousef	M.Y.
Abdoh Elgazar	A.E.
Farrah Tharwat	F.T.

What are ChatBots?

- ChatBots are artificially intelligent conversation agents.
  They can be used for various things including customer support agents replying to frequently asked questions.



  - For business ChatBot can perform the role of human customer agent handling common queries.

- Since ChatBots can be used simultaneously in multiple instances, they can scale to a potentially large influx of customers.

  - We can use existing chat transcripts between an agent and a customer.
    However Curating data is an expensive task in terms of time and cost both manpower and financial.

What are ChatBots?

ChatBots

Generative

- More beneficial to open domain tasks.
- Sounds a little more natural. Responses cam be different each time.
- The model learn to adopt the responses to the user.
- There are a couple of drawbacks from this:
  - We might not always know if it is providing correct info.
  - User might be inclined to feel they are speaking to human which may make them frustrated about inappropriate responses.
  - Tend to be computationally expensive.

Retrieval ✓

- More beneficial to close domain tasks.
- Responses may be easy to identify as a bot. So the user may phrase their questions in a way that is easy to understand therefor to predict.
- The disadvantage is that it doesn't sound very natural , however it is relatively less computationally expensive and takes less time to create and deploy.

In our use-case it is close domain, specific business setting for specific type of products.
The types of responses are limited.
Moreover the]need to maintaining a conversation is minimum.

Dataset

In order to create a bespoke and robust chatbot it's beneficial for the business to tailor its chatbot for their use case.

One way to achieve this is to use in-house data.

Training a chatbot system using the data ensures that queries and responses are business specific (in-domain).
In order for the model to be able to accurately predict an intent given an utterance; typically a large amount of data would be required.
This may not always be possible from a business perspective partly because of the dearth of available chat transcripts from which the data is created or even with a sufficient amount of chat transcripts there may not be enough resources to be able to convert that into training data.

Another way is to use a pre-trained model

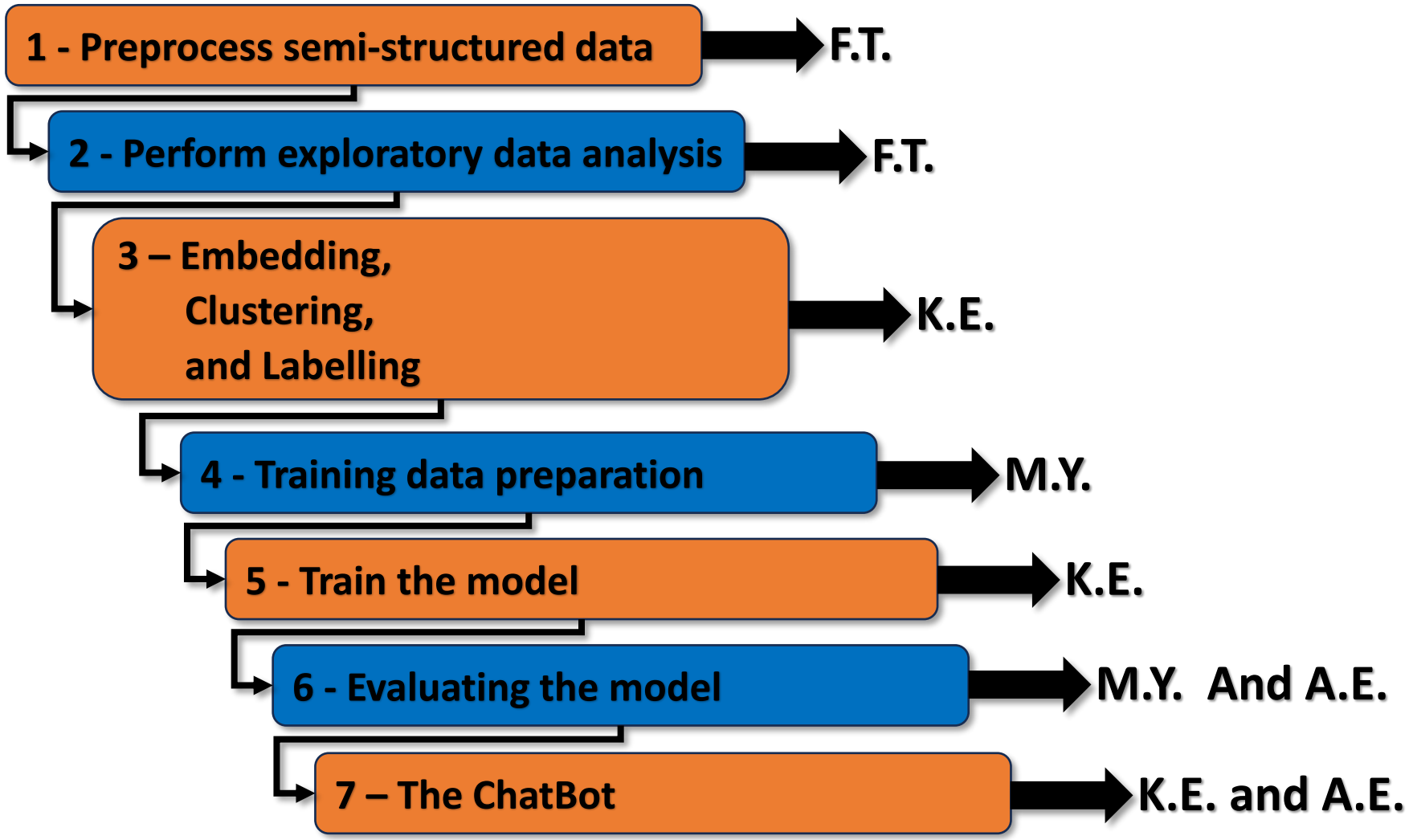
By utilizing existing pre-trained models trained on similar tasks to then optimize towards a business specific task using fine-tuning.
There are numerous advantages to using a pre-trained model some of which are:

- Reducing the cost of computational resources and time
- Being able to leverage a large amount of data that the pre-trained model had been trained on
- Ramp up training and deployment relatively quickly compared to building a state-of-the-art intent recognizer from scratch.

From a business perspective these are meaningful factors that make utilizing such an approach a compelling option.

One drawback is that the pretrained model outsize the influence against parameter weights introduced during finetuning with our data set.

Project tasks and roles:



Step 01

Preprocess semi-structured data

1 Slide

Step 01

Preprocess semi-structured data

- We are using data provided by a company for internal chat transcripts between customer agents and users inquiring about the products. We have a total of 75 transcript logs in .txt format.
- The data can be considered **semi-structured** this is because there are features within the transcript that allow us to extract relevant data. For example the time standard format only goes alongside chatter identity, and then the utterances comes alongside the chatter identity. `(2022-03-31 13:10:17) Agent: shall i give you the discounted link`
- Parsing the files by taking a directory containing raw chat logs as input and returns data frame of preprocessed text with speaker number.
  - The data is standardized using **normalization**, **tokenization** and **lemmatization**.
  - Utterances are stripped from any character except for alphabetic characters and single occurrence space.
  - URLS are removed

```
[6]: data = parse_files(DATA_DIR)

[7]: data
```

[7]:

	participant	original_text	text
0	2	Hi, please let me know how I can he...	hi please let I know how can help yo...
1	2	View a list of 120+ end-to-end Mach...	view list of end to end machine lear...
2	2	Solution code + videos + tech suppo...	solution code video tech support moc...
3	1	Hello Sure	hello sure
4	2	hi	hi
...	...	...	...
9748	1	ok.. please inform them.. I am alre...	ok please inform they be already wor...
9749	2	please provide your contact number	please provide your contact number
9750	2	will arrange a call back	will arrange call back
9751	1	I think fees is much on higher side...	think fee be much on high side for t...
9752	1	*videos	video

9753 rows × 3 columns

Step 02

Perform exploratory data analysis

1 Slide

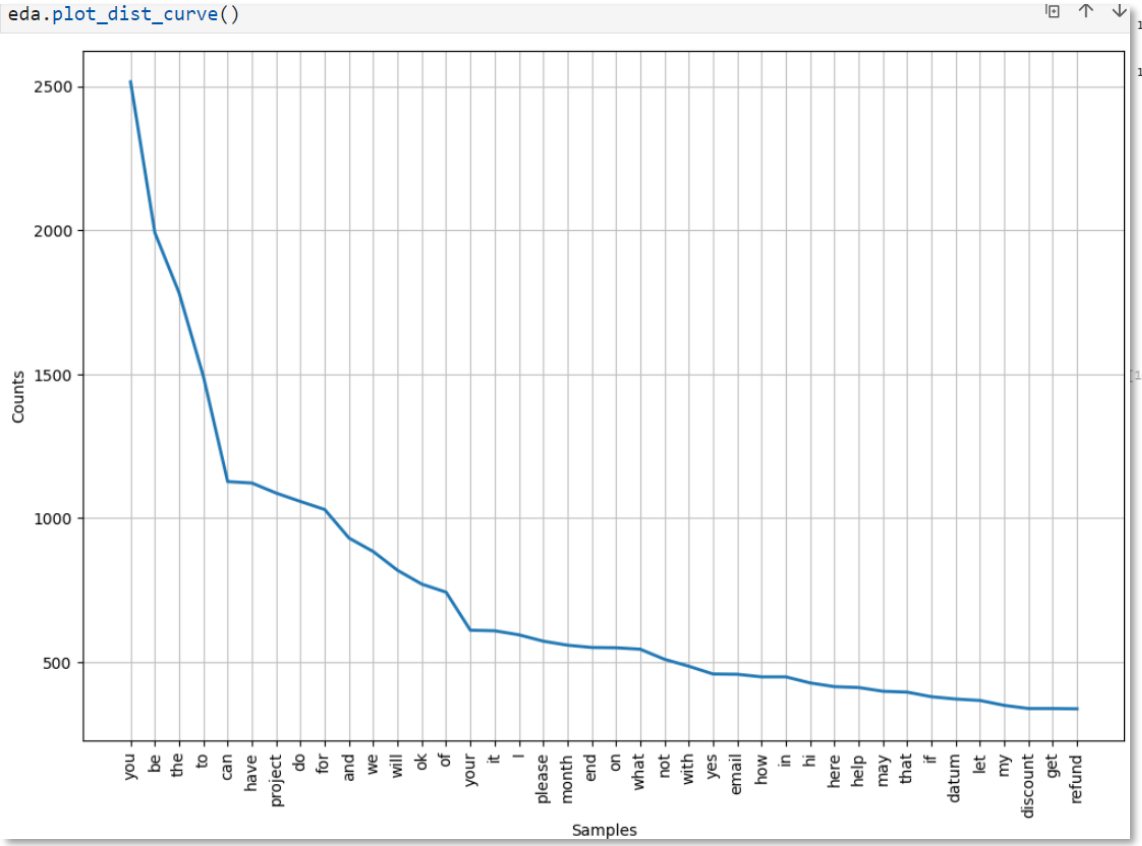
Perform exploratory data analysis

Checking features of the after initial preprocessing.

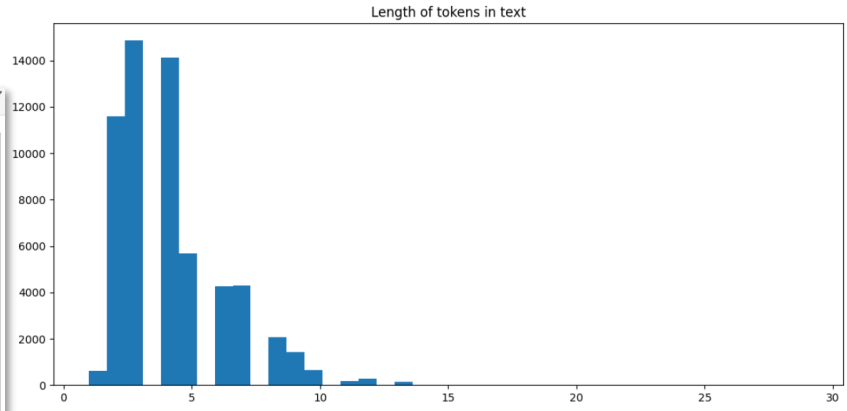
```
[11]: eda.get_top_n_tokens(20)
```

```
[11]: [('you', 2515),
       ('be', 1991),
       ('the', 1782),
       ('to', 1492),
       ('can', 1127),
       ('have', 1122),
       ('project', 1087),
       ('do', 1058),
       ('for', 1030),
       ('and', 931),
       ('we', 884),
       ('will', 819),
       ('ok', 771),
       ('of', 743),
       ('your', 611),
       ('it', 609),
       ('I', 595),
       ('please', 573),
       ('month', 559),
       ('end', 551)]
```

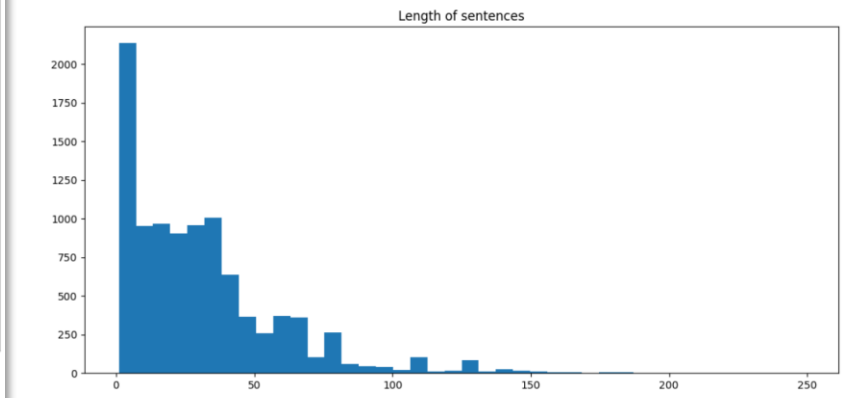
```
eda.plot_dist_curve()
```




```
[13]: eda.get_token_length_visualisations()
```




```
[14]: eda.get_sent_length_visuals()
```



Step 03

Start the process of labeling the data

5 Slides

Start the process of labeling the data

- Humans can effortlessly classify different utterances into different intent labels.
  In our project machine learning methods such as clustering are used to automate some of this process.

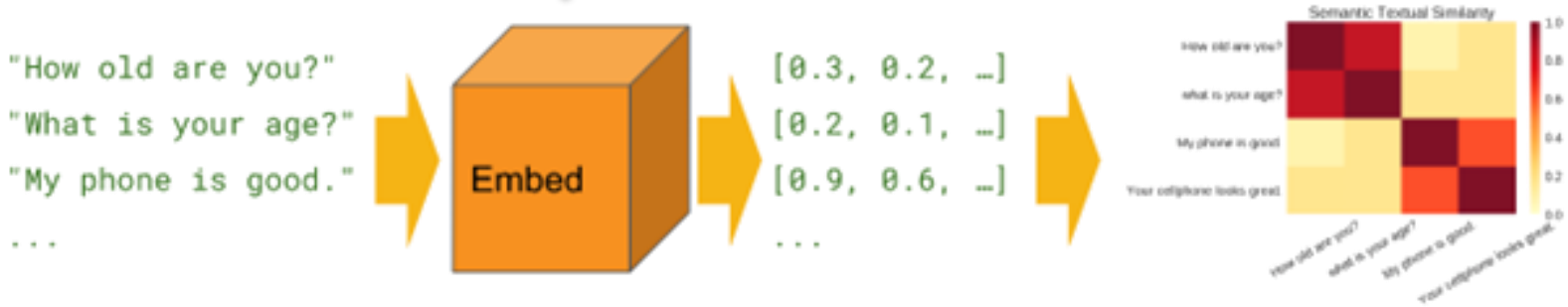
- Typically human labeling is described as **supervised** and the automated method is described as **unsupervised**.

- Some of the disadvantages of unsupervised methods are that the labels might not accurately represent the utterances. However this approach can provide some meaningful assistance to the labeling task effectively bootstrapping the human labeling process.

- In order to complete the labeling we will go through some steps:
  - Collecting all the utterances → *all_intents* [9753 utterances]
  - Embedding the *all_intents* using our pre-trained model → *embeddings* = embedder(*all_intents*) [Shape 9753,512]
  - Clustering using **chatintents( )** package.
  - Once the best clusters are determined labels of the clusters are generated using some syntactic information from common utterances in the clusters.

Embedding

Semantic Similarity



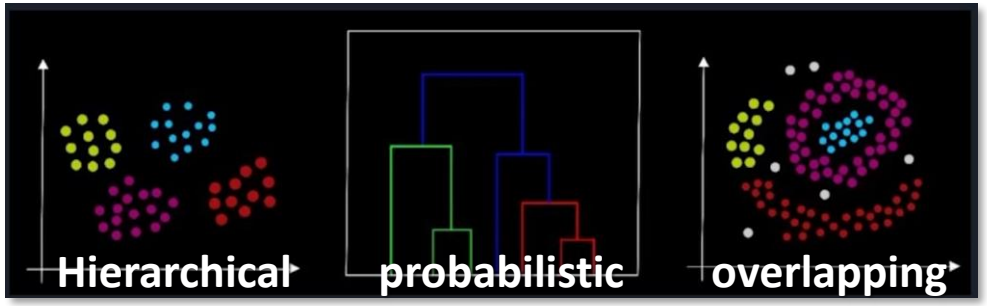
Our pre-trained model **universal-sentence-encoder** can perform sentence embedding. It can take and convert sentence into a vector of representation using information from hidden layers to derive the values of the vector

Clustering

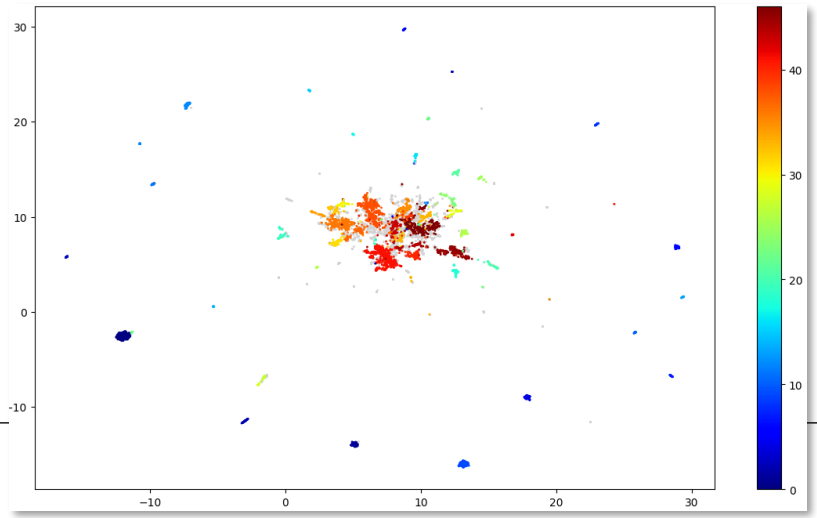
- The existing **chatintents( )** clustering package is used. The authors of the package designed based on motivation.

- Convert embeddings array to sentence embeddings using USE

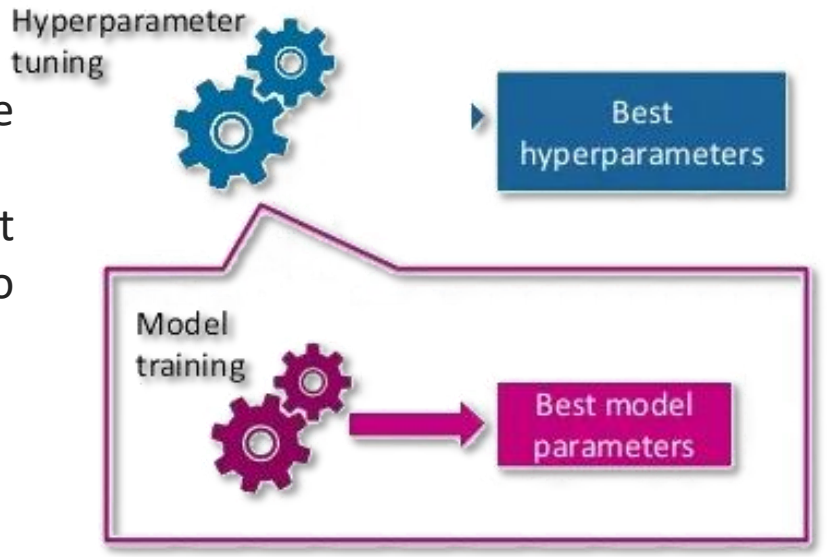
  ChatIntents(*embeddings*, 'use')



- The existing **chatintents( )** clustering package is used. The authors use a unique approach to directly hyperparameter tune both the dimensionality of algorithm in this case UMAP and the clustering algorithm HVSCAN to identify the optimal clusters. This is beneficial to small data sets like ours.



Hyperparameter tuning



Hyperparameters are the variables of the algorithm that control its whole behavior.

A good example is the **learning rate**. When it is too large, the learning isn't sensitive enough, and the model results are unstable. But when it is too small, the model has trouble learning and might stuck.

Types of Hyperparameter Search

There are three main methods to perform hyperparameters search:

1.Grid search
2.Randomized search
3.Bayesian Search

The tuning is bayesian_search algorithm which is showing to perform better than just random search.

The main difference between Bayesian search and the other methods is that the tuning algorithm optimizes its parameter selection in each round according to the previous round score. Thus, instead of randomly choosing the next set of parameters, the algorithm optimizes the choice, and likely reaches the best parameter set faster than the previous two methods. It can be beneficial to **minimize the tuning time**.

Labeling

- Once the best clusters are determined labels of the clusters are generated using some syntactic information from common utterances in the clusters.

count	label
2593	let_project_month
744	want_project_data
573	ok_ya
397	let_time_pm_tomorrow
310	yes
304	send_email_team
303	pay_plan_month_subscription
302	help_discount_price
219	hi_day
199	learn_machine_end_view
195	sure_giron_welcome
180	resume_guarantee_solution_code
179	let_today
148	demo_demo_session
142	price_price_fee
140	access_access_month
131	pay_card_emi_cost
123	work_kind_technology_training
122	enrol_access_student_transition
118	click_address_email_password
118	refund_refund_day
118	_email
116	anjali_office_indian_company
99	hello
96	send_invite_team_calendar
92	pay_link_payment_time
92	like_online_line_note
90	nope_mam_affiliate_universit
87	following_project
81	provide_plan_month_day
81	check_faq_enrolment_detail
76	unlimited_mentor_expert_project
75	great
74	provide_number_contact
72	okk_studnet
71	schedule_understanding_demo_min
66	thank_thank_ohkk
63	plan_plan_discount
59	okay
57	help_process_job_procedure
49	use_card_credit
43	regard_project
42	oh
42	thank
40	let_link_time_list
38	reply_help_mam
37	welcome
35	provide_option_placement_minnesota

48 clusters

```
cluster.get_model_best_params()
```

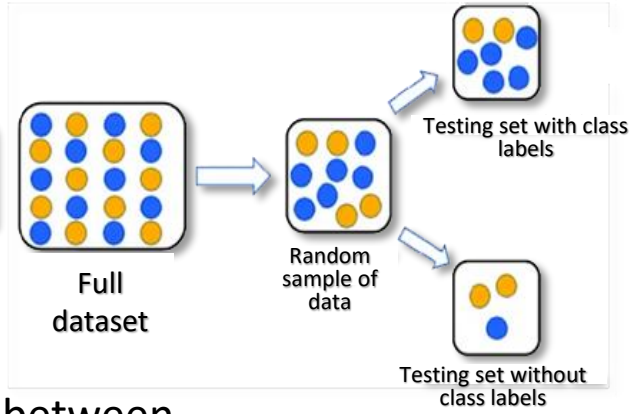
```
{'state': 2,
 'tid': 0,
 'spec': None,
 'result': {'loss': 0.2677124987183431, 'label_count': 48, 'status': 'ok'},
```

Step 04

Training data preparation

2 Slides

Training data preparation



- The data is splitted into: training, test and validation
- As a standard the training tends to be the largest proportion of the data between 70 to 80 present.
- We are using 80% 10% 10% split
  Since we have small amount of data.
- Once the data is splitted the data is converted into numpy arrays that will be used as inputs to the model which will convert them into sentence embeddings.
- The labels or the output will be one hot encoded.

id	color
1	red
2	blue
3	green
4	blue

One Hot Encoding

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

'monthly_payments': {**'1'**: ['is there any plan for a monthly subscription ?', 'do you have monthly payment option', 'so is it full payment or every month I need to pay?', 'can i pay month wise?', 'but still I want to pay it montly', 'is the payment monthly or upfront', 'can pay this per month or have to be one time payment', 'ok can we pay per month basis?', 'So I will have to pay full money in one go?', 'is it per month?', 'initially, 1 month amount neds to be paid right?', 'per month?', 'do i have to pay upfront and can i pay monthly?', 'Okay but i want to play monthly', 'how much i should pay for a month', 'Do we need to pay monthly basis?', 'I asked a question regarding payment. will it be monthly payment or one time payment?', 'was it monthly payment allowed or just pay at once?', 'i mean is it monthly charge or one at a time?', 'I can pay monthly charges', 'Or monthly charges', 'what is per month cost?', 'Hey, May I know what will be the cost of 1 month?', 'do I need to pay per month?', 'whats the monthly price', 'i want to ask when I subscribe to a plan you debit all the amount of subscription or you debit monthly ?'],

'2': ['if you want monthy payment options we have collaboration with Affirm and could help you with monthly part payments', 'yes you can pay monthly', 'you need to enter your details ... and based on your credit history there will be monthly .. quarterly and 6- month part payment plans', 'We have a collaboration with Affirm ... where you can pay monthly ..', 'we have collaboration with Affirm for monthly part payment options', 'and you have to pay as per your due date for the monthly credit card payment..', 'where you can pay monthly.. quaterly or 6- monthly', 'you can pay thru monthly part payments .. but it depends on your credit history ...', 'for monthly payments we have a collaboration with affirm', 'you can pay monthly', 'you could go for monthly payment plan', 'if you want you can pay monthly', 'you can go for monthly ... quaterly or 6 months payment']},

'1'→ Query

'2'→ Response

Training data preparation

- The data set has been significantly reduced to close to 459 utterances from 9,753.
- This is due to a number of duplicate types of queries and a lot of out of context and out of scope queries.
- **'2' Response** is being dropped since it will not be needed for the training.

	intent	query	query_preprocessed
39	i_get_back	right now cant make teh payment	right now can not make teh payment
39	i_get_back	I will think about it. Thanks for yo...	will think about it thank for your t...
...	...	...	...
39	i_get_back	Hey I need some time to think	hey need some time to think
39	i_get_back	okay ..will contact you tomorrow	okay will contact you tomorrow
40	you_get_back	Can you send me an email, I will thi...	can you send I an email will think a...

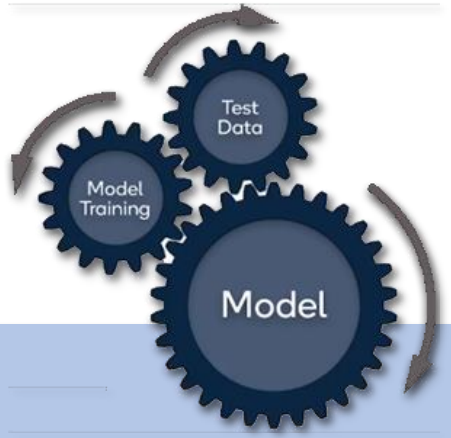
459 rows × 3 columns

Step 05

Train the model

3 Slides

Train the model



universal-sentence-encoder

https://www.kaggle.com/models/google/universal-sentence-encoder/tensorFlow2/universal-sentence-encoder/2?tfhub-redirect=true

The Universal Sentence Encoder encodes text into high-dimensional vectors that can be used for text classification, semantic similarity, clustering and other natural language tasks.

The model is trained and optimized for greater-than-word length text, such as sentences, phrases or short paragraphs. It is trained on a variety of data sources and a variety of tasks with the aim of dynamically accommodating a wide variety of natural language understanding tasks. The input is variable length English text and the output is a 512-dimensional vector. The universal-sentence-encoder model is trained with a deep averaging network (DAN) encoder.

The encoder differs from word level embedding models in that it is trained on a number of natural language prediction tasks that require modeling the meaning of word sequences rather than just individual words.

Train the model

- The pre-trained model is provided without the output layer, we provide our inputs to the final layer.
- It is then optimized using back propagation and the pretrained weights help to classify the output.
- Using a pre-trained model reduces the amount of time for the loss to be reduced and accuracy to be increased, which reduces the training time overall and computational processing.

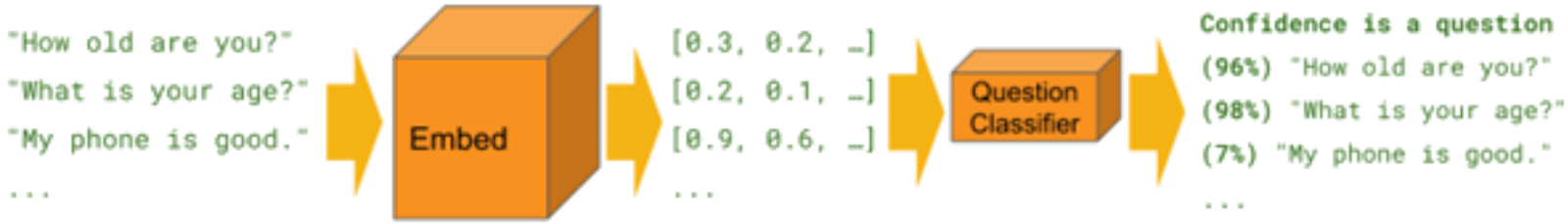
```
Search: Running Trial #1

Value             |Best Value So Far |Hyperparameter
2                 |?                 |num_layers
384               |?                 |units
sigmoid           |?                 |activation
0.5               |?                 |dropout_rate
0.001             |?                 |lr

12/12 [==============================] - ETA: 0s - loss: 3.6407 - categorical_accuracy: 0.1117
```

Train the model

Classification



The Universal Sentence Encoder was partially trained with custom text classification tasks in mind. These kinds of classifiers can be trained to perform a wide variety of classification tasks often with a very small amount of labeled examples.

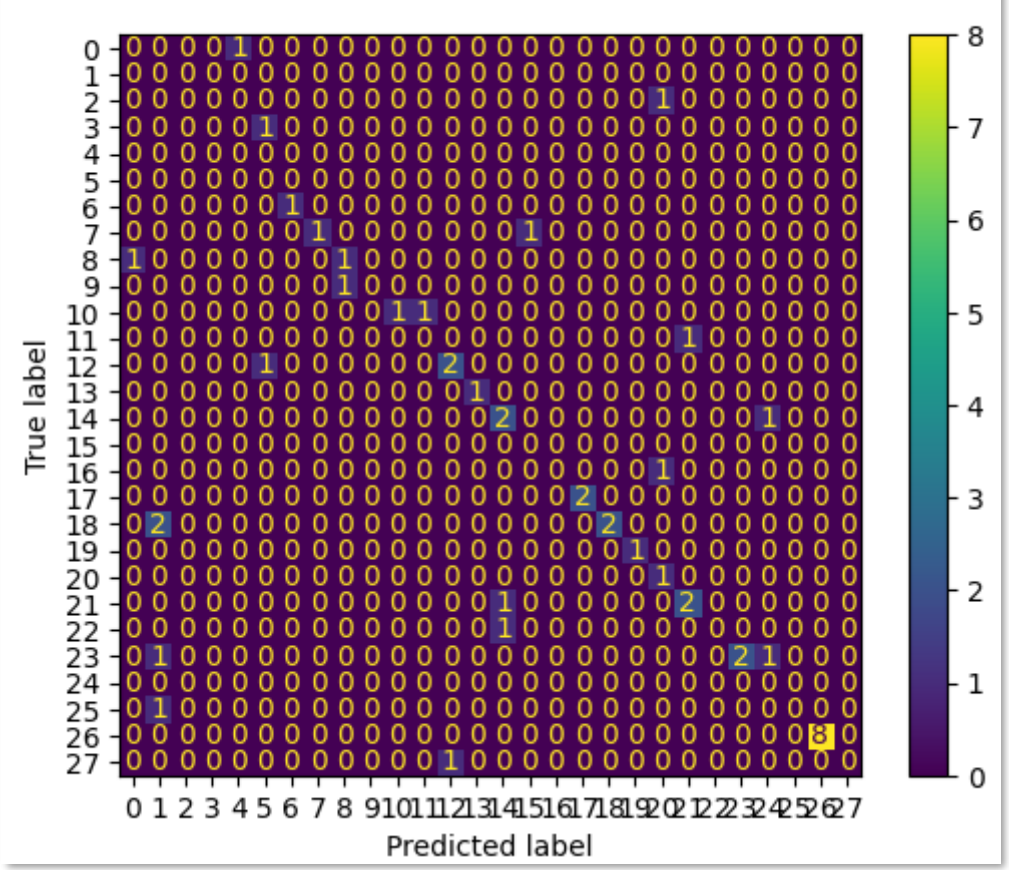
Step 06

Evaluating the model

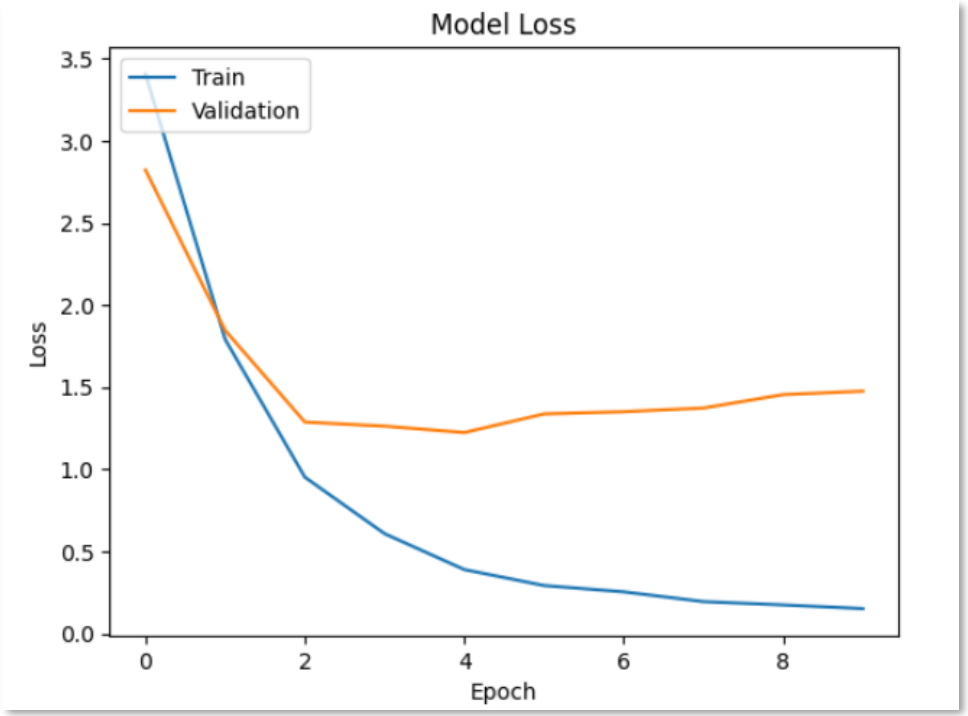
2 Slides

Evaluating the model

```
evaluate.get_confusion_matrix()
```




```
2/2 - 1s - loss: 2.5519 - accuracy: 0.5870 - 536ms/epoch - 268ms/step
Test loss: 2.551934003829956
Test accuracy: 0.5869565010070801
```



Evaluating the model

```
evaluate.compare_predicted_intents()
```

	intent	predicted_intent
39	i_get_back	i_get_back
30	project_want	i_get_back
26	thanking	thanking
9	payment_link	payment_link
34	give_email	ask_email
8	location	payment_link
25	demo_discussion	live_session_mentoring
26	thanking	thanking
39	i_get_back	refund_process_time
26	thanking	thanking
25	demo_discussion	demo_discussion
10	Paypal	ask_email
7	MLOPS	payment_link
31	refund_info	refund_info
26	thanking	thanking
26	thanking	thanking
2	download_can	enrollment
30	project_want	project_want
24	projectpro	i_get_back
28	discount_student	discount
26	thanking	thanking
36	noise	noise
29	enrollment	project_want
37	greetings	greetings
31	refund_info	refund_process_time

27	discount	discount
30	project_want	project_want
18	monthly_payments	monthly_payments
18	monthly_payments	monthly_payments
19	12_month	annual
34	give_email	give_email
17	monthly_subscription	monthly_subscription
34	give_email	give_email
15	cost	cost
31	refund_info	6_months
26	thanking	thanking
17	monthly_subscription	6_months
40	you_get_back	give_email
2	download_can	download_can
26	thanking	thanking
13	renew	6_months
17	monthly_subscription	monthly_subscription
27	discount	12_month
31	refund_info	refund_info
39	i_get_back	i_get_back
17	monthly_subscription	6_months

```
You: Can I have a trial?
Kha ChatBot: shall i give you the payment link
You: Can I have a trial?
Kha ChatBot: shall i give you the payment link
You: where is your location?
Kha ChatBot: This link will be valid for couple of hours .. so please ensure to complete the enrollment as soon as possi
ble
You: where is your address?
Kha ChatBot: Please complete the enrollment using the abouve link
You:
```

Step 07

The ChatBot

3 Slides

The ChatBot

```
How may we help you? (type 'quit' to exit)
You: hi
Kha ChatBot: Hi
```