GitHub link

AIML Online Capstone Project
AUTOMATIC SUPPORT TICKET ASSIGNMENT

- Prasad Darbhamulla
- Anil Potukuchi
- Niranjan
- Ram R
- Atanu Sinha
- Bruno Suresh Lucas

Date of Submission(Interim) : 05-Jul-2020

Business Case

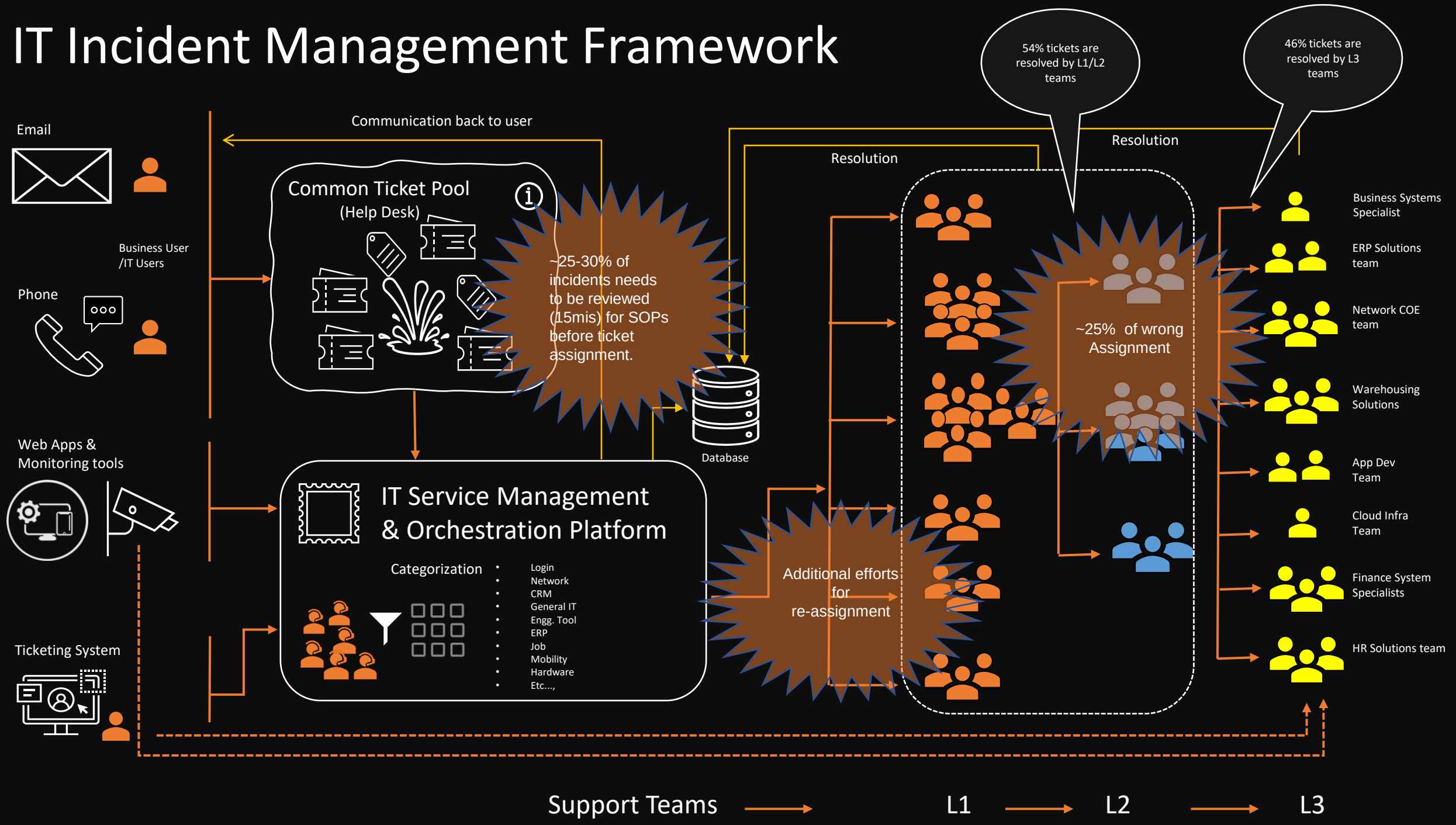
Background:

- In the support process, incoming incidents are analyzed and assessed by organization’s support teams to fulfill the request.
- Better allocation and effective usage of the valuable support resources will directly result in substantial cost savings.
- Currently, the incidents are created by various stakeholders and are assigned to L1/L2 teams.
- Incase L1 / L2 is unable to resolve, they will then escalate / assign the tickets to Functional teams from Applications and Infrastructure (L3 teams).

Problem Statement:

- L1 / L2 needs to spend time reviewing Standard Operating Procedures (SOPs) before assigning to Functional teams (Minimum ~25-30% of incidents needs to be reviewed for SOPs before ticket assignment).
- 15 min is being spent for SOP review for each incident. Minimum of ~1 FTE effort needed only for incident assignment to L3 teams.
- Multiple instances of incidents getting assigned to wrong functional groups (Around ~25%)
- Additional effort needed for Functional teams to re-assign to right functional groups.
- During this process, some of the incidents are in queue and not addressed timely resulting in poor customer service.

IT Incident Management Framework



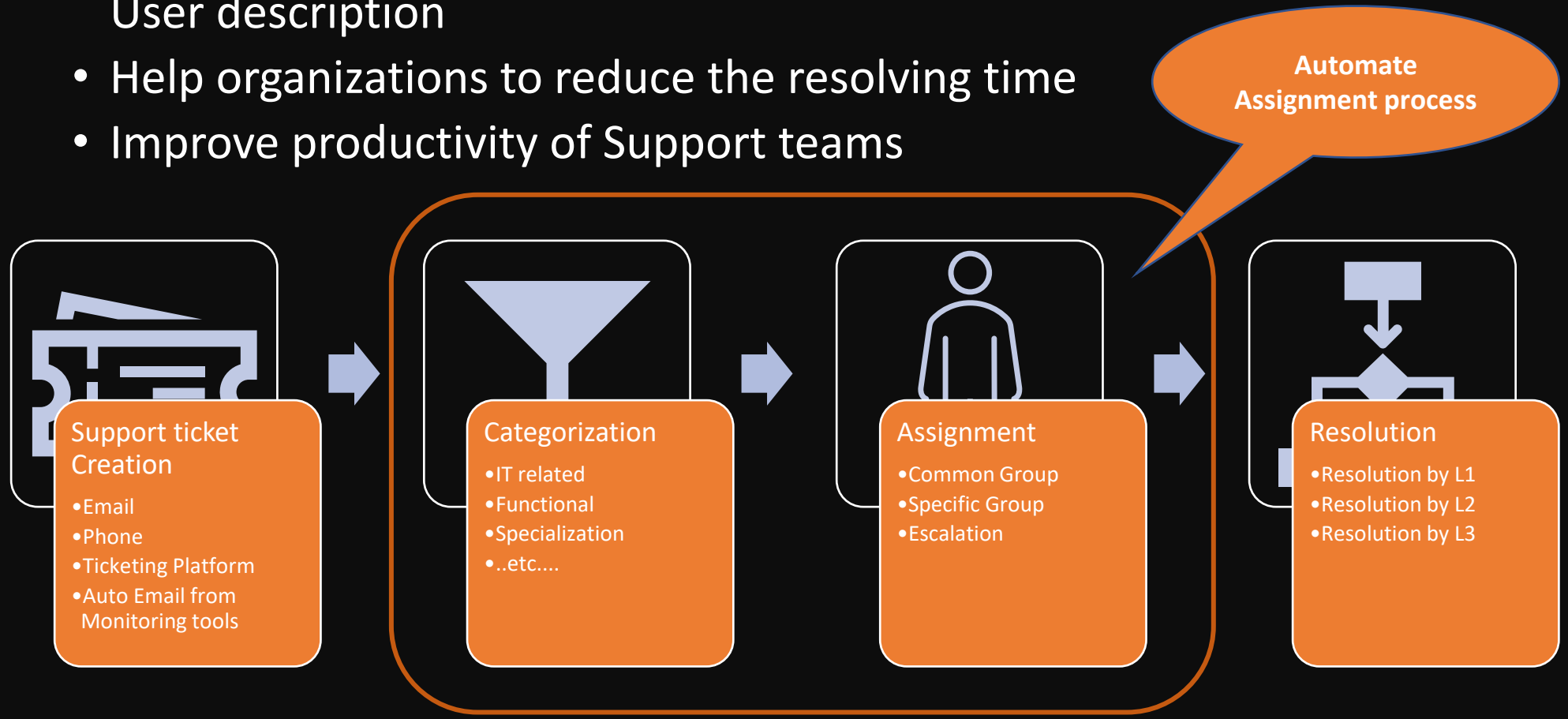
Challenges

- Multiple Languages
- Multiple reception modes
  - Email, Call, Chat, Monitoring tools, Direct entry into system
- Length of the sentences is too sparse (description ranging from few words to several hundreds)
- Too many resolution groups – 74 assignment groups, some of them having very few tickets assigned

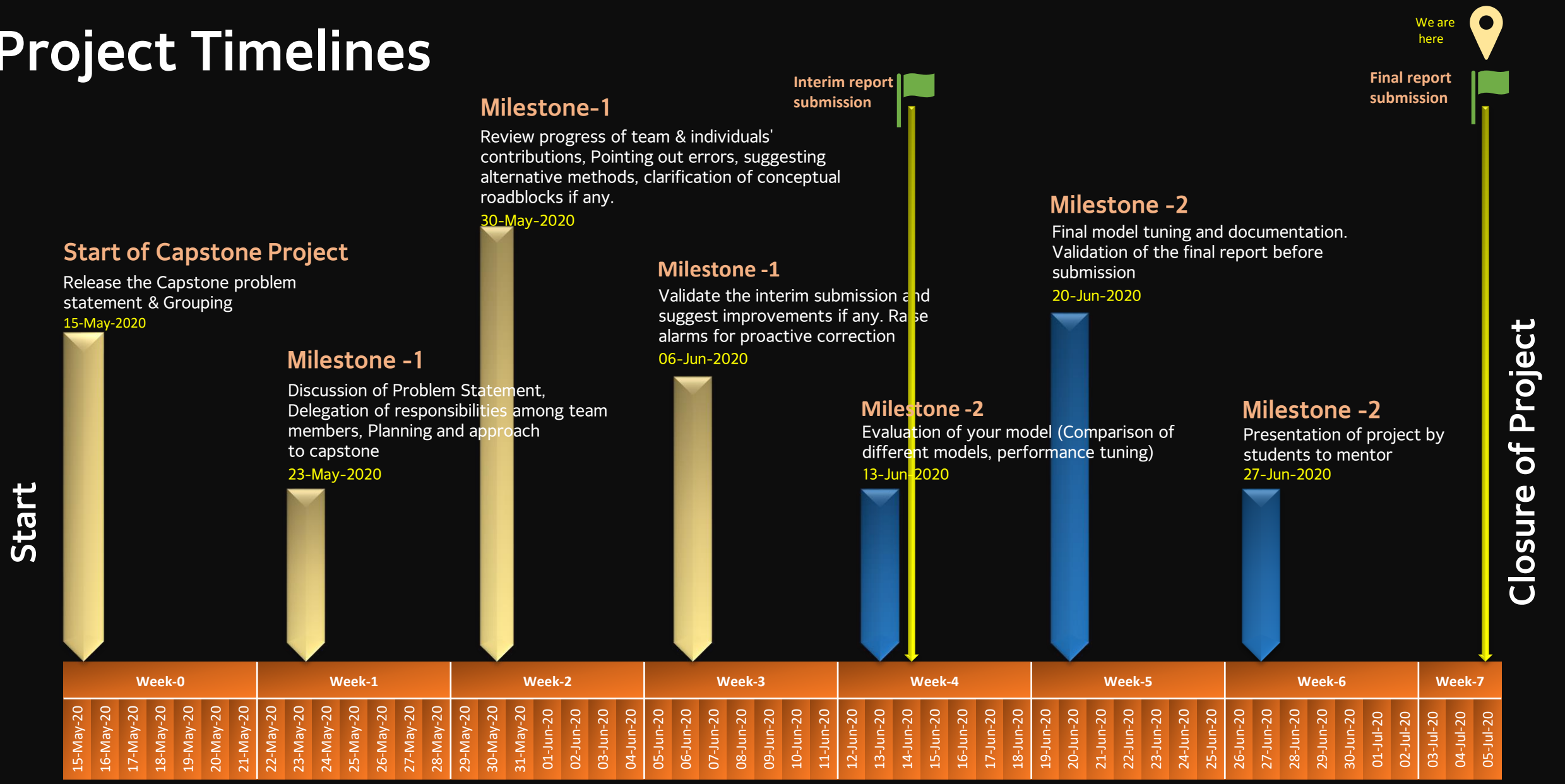
Project Objective

Guided by powerful AI techniques

- Classify incidents to right functional groups based on User description
- Help organizations to reduce the resolving time
- Improve productivity of Support teams



Project Timelines



Milestone1 : Understanding Business case, Project Objective, Data Pre-Processing, EDA, Basic model Building

Milestone2 : Model Evaluation, Model Tuning, Final Document preparation and Report submission

Input Data Analysis

There are total 8500 records in the given dataset and 4 columns namely

- Short description
- Description
- Caller
- Assignment group

It seems like GRP_0 and GRP_8 are L1/L2 Groups

Count of Short description	Count of Description	Count of Caller	Count of Assignment group
8498	8499	8500	8500

Top 11 Short Description	Count of tickets
password reset	38
windows password reset	29
account locked in ad	24
windows account locked	23
erp SID_34 account unlock	19
unable to connect to vpn	18
login issue	18
blank call	18
account locked.	18
erp SID_34 password reset	17
erp access issue	17

Top 11 Assignment Groups	Count of tickets	%
GRP_0	3976	46.78%
GRP_8	661	7.78%
GRP_24	289	3.40%
GRP_12	257	3.02%
GRP_9	252	2.96%
GRP_2	241	2.84%
GRP_19	215	2.53%
GRP_3	200	2.35%
GRP_6	184	2.16%
GRP_13	145	1.71%
GRP_10	140	1.65%

Data – Pre-Processing

Summary :

- There are 8500 records in the dataset
- Multiple Languages are detected. However, the top Languages are **English and German**
- Both the short and long description translated to English
- Special Characters were counted. Records with the no of spl characters more than 50% are deleted.
- 8 records with null values in short description were deleted.
- It was found that the tickets are created through Call, Email, Monitoring Systems, by users directly in the ticketing system or by the agents..
- Email id’s were identified from the description column and was stored separately.
- Both description and short description are processed and removed of the following things
  - Sentence trimmed up to “Received From : ”
  - Email ID
  - Special characters and excess spaces were removed
  - Digits, punctuation & HTML tags removed
  - If caller ID present in the description, it was removed

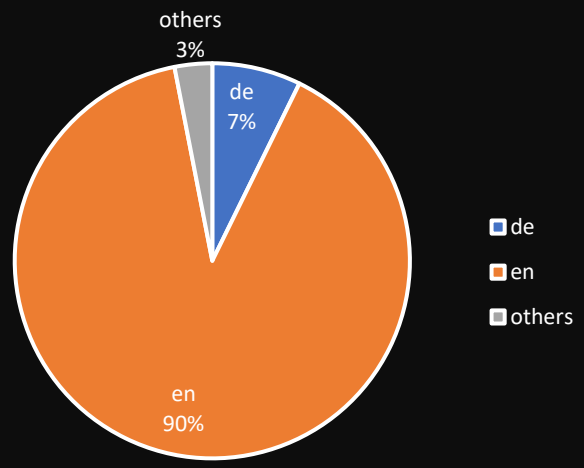
Data – Pre-Processing

New Columns Created for Analysis :

Column Name	Description	Column Name	Description
Short description	**Original**	trimmed_words_len	length of trimmed_words
Description	**Original**	trimmed_words_short	stop words removed from short_desc_en_spl
short_desc_lang	Language of the Short Desc	trimmed_words_long	stop words removed from description_en_spl
short_desc_en	Short Desc translated to English	mark_for_delete	flagged for delete
description_lang	Language of Description	flg_account_related	flg_account_related
description_en	Description Translated to English	flg_job	flg_job
Caller	**Original**	flg_vpn	flg_vpn
Assignment group	**Original**	flg_network	flg_network
emails	Email Id from description	flg_erp	flg_erp
email_type	type of email /call	flg_crm	flg_crm
short_desc_spl	Short Desc after removal of spl characters	flg_engineering	flg_engineering
description_spl	Description after removal of spl characters	flg_install	flg_install
short_desc_len	length of Short Desc	flg_laptop	flg_laptop
description_len	lebgth of Description	flg_mobile	flg_mobile
no_of_splch_removed1	no of spl characters removed from Short desc	flg_outlook	flg_outlook
no_of_splch_removed1_pct	% no of spl characters removed from Short desc	flg_printer	flg_printer
no_of_splch_removed2	no of spl characters removed from Desc	flg_skype	flg_skype
no_of_splch_removed2_pct	% no of spl characters removed from Desc	flg_ticket_update	flg_ticket_update
short_desc_en_spl	Spl characters removed (short_desc_en)	flg_windows	flg_windows
description_en_spl	Spl characters removed (description_en)	host_related	host_related
short_desc_en_len	length of short_desc_en_spl	blank_call	blank_call
description_en_len	length of description_en_spl	setup_ws	setup_ws
trimmed_words	stop words removed and short_desc_en_spl, description_en_spl combined	flg_help	flg_help
		flg_failed	flg_failed

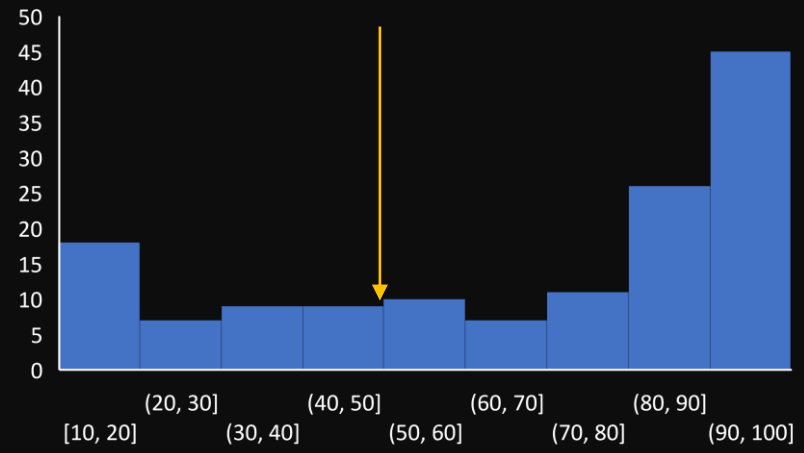
EDA – Exploratory Data Analytics

Ticket Distribution by Language



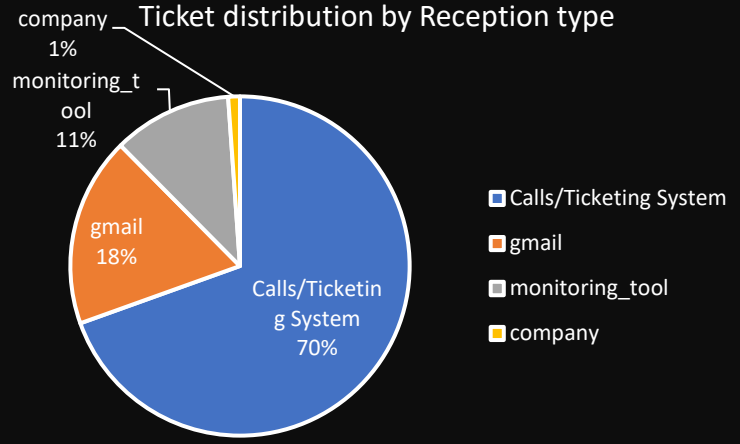
97% of the records belong to English and German

Records where Spl characters > 10%



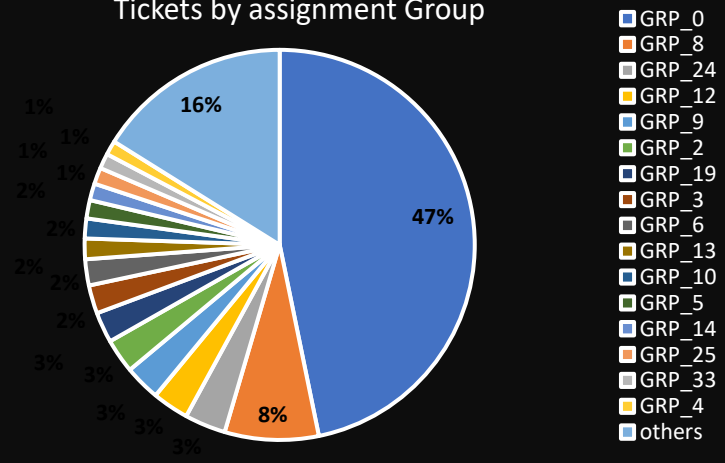
Records with >=50% spl characters (118) were deleted

Ticket distribution by Reception type



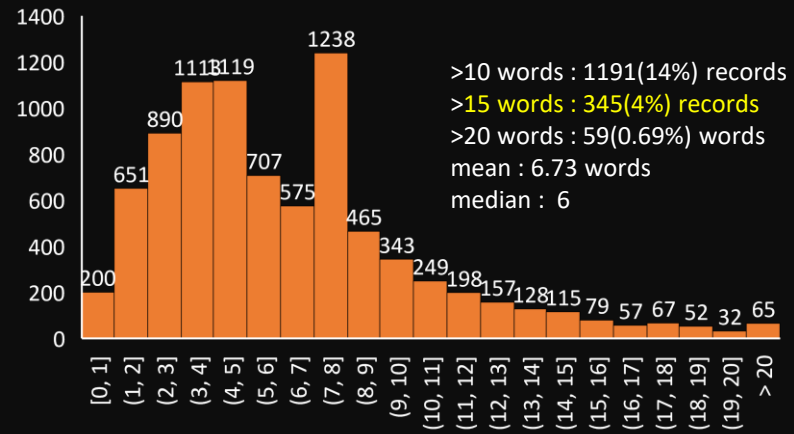
- Majority of the tickets are through calls/direct entry
- 18% tickets through email (external)
- 11% tickets are created by monitoring systems

Tickets by assignment Group



54% of the tickets are resolved by Grp_0 and Grp_8. They must be L1/L2 groups. There are 58 Groups in the Others category

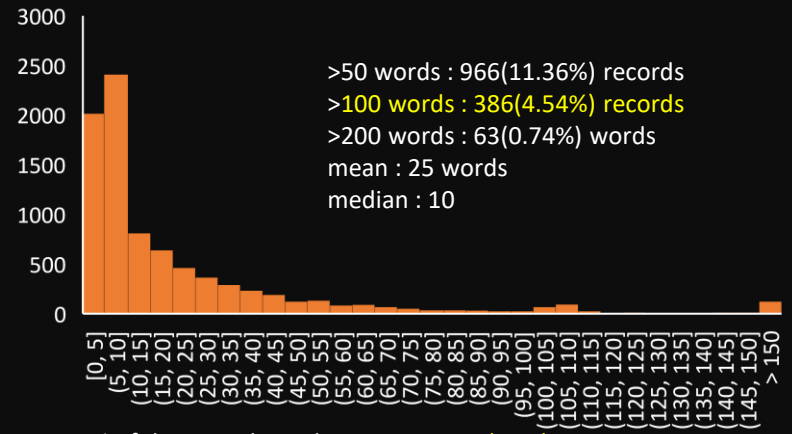
No of words in Short Description - Distribution



>10 words : 1191(14%) records
>15 words : 345(4%) records
>20 words : 59(0.69%) words
mean : 6.73 words
median : 6

95% of the records are having 15 words or less. We can use this as the baseline to trim the sentence

No of Words in Description - Distribution



>50 words : 966(11.36%) records
>100 words : 386(4.54%) records
>200 words : 63(0.74%) words
mean : 25 words
median : 10

95% of the records are having 100 words or less. We can use this as the baseline to trim the sentence

EDA – With Key words

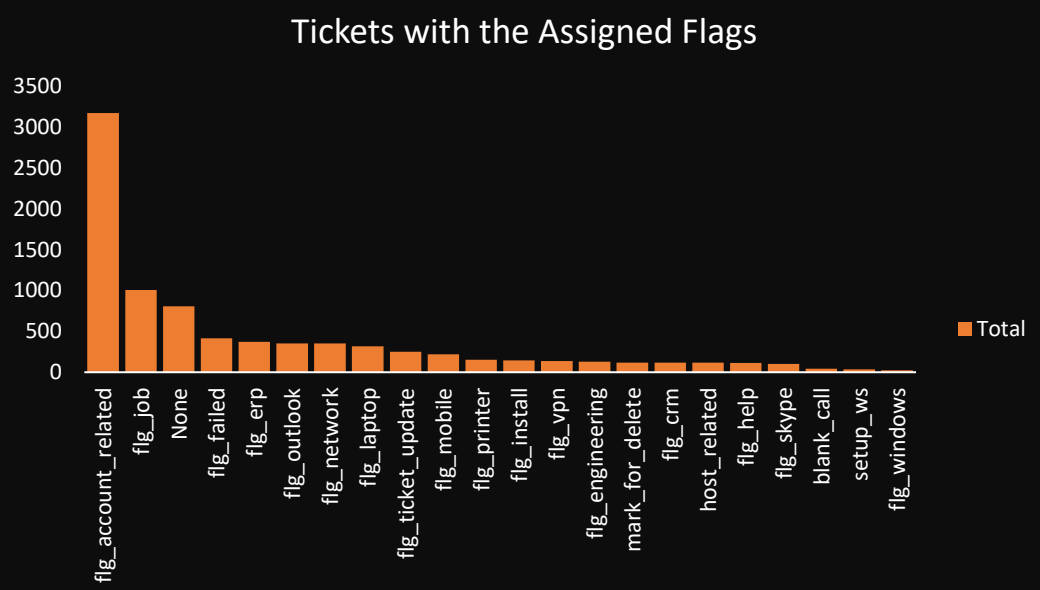
Key word search done in the Short and Long description and the words are assigned to categories

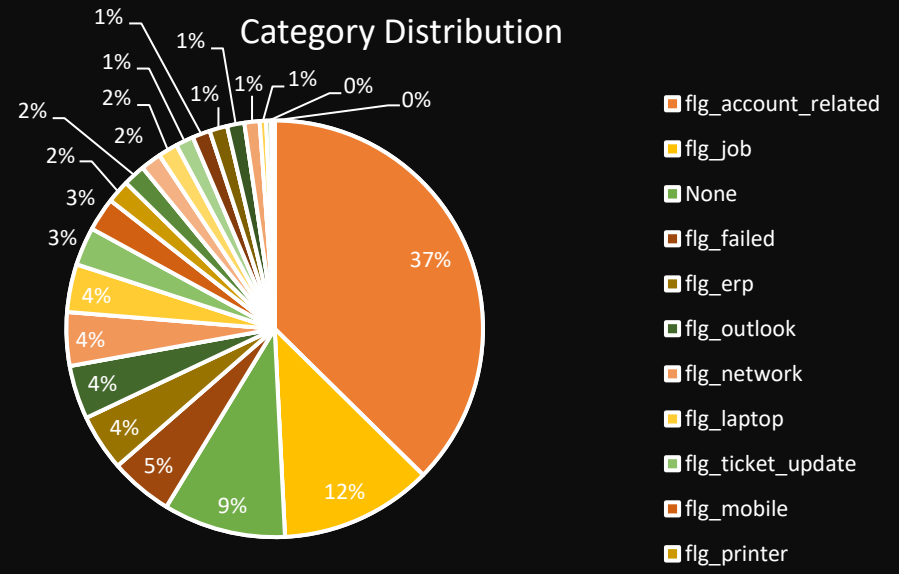
word	category
access	account/reset/password/login/reset
account	account/reset/password/login/reset
circuit	network
computer	laptop
connection	network
crm	crm
email	outlook
emails	emails
engineering	engineeringtool
engineeringtool	engineeringtool
erp	erp
error	failed
failed	failed
help	help
install	install
issue	failed
job	job
jobscheduler	job
laptop	laptop
locked	account/reset/password/login/reset
login	account/reset/password/login/reset
logon	account/reset/password/login/reset
network	network
outage	network
outlook	outlook
password	account/reset/password/login/reset
passwordmanagementtool	account/reset/password/login/reset
passwords	account/reset/password/login/reset
phone	Mobile
print	printer
printer	printer
problems	failed
reset	account/reset/password/login/reset
server	network
skype	skype
telephonysoftware	mobile
ticket	ticket update
tologin	account/reset/password/login/reset
unable	failed
unlock	account/reset/password/login/reset
update	ticket update
user	account/reset/password/login/reset
vpn	vpn
vpncompanycom	vpn
windows	windows

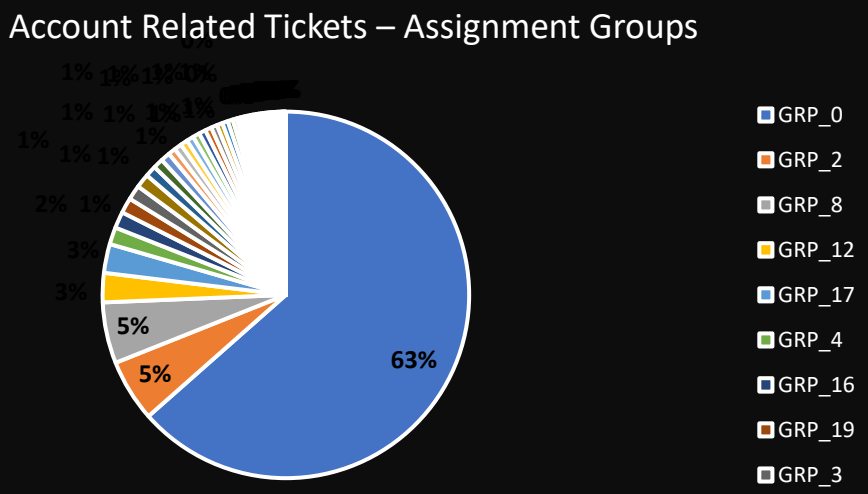
Dataset is enriched with the following flags to understand the correlation between the categories and the Assignment Groups

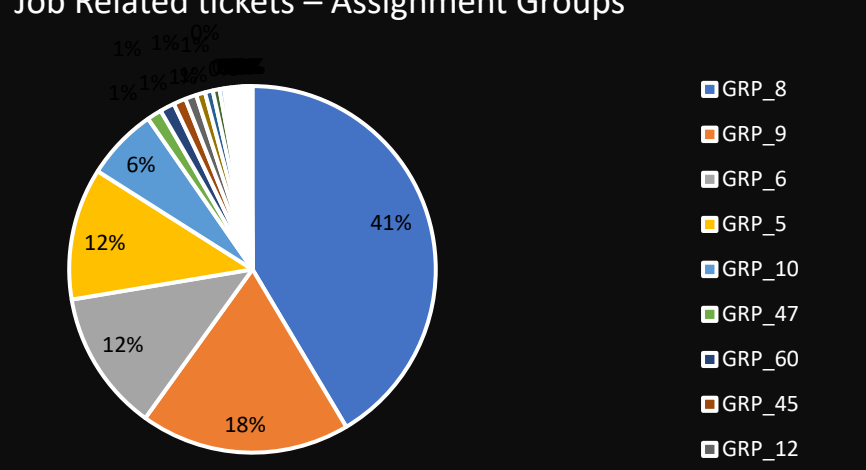
Row Labels	Count of Final_Flag	Count of Final_Flag2
flg_account_related	3177	37.38%
flg_job	1006	11.84%
None	807	9.49%
flg_failed	415	4.88%
flg_erp	373	4.39%
flg_outlook	354	4.16%
flg_network	353	4.15%
flg_laptop	316	3.72%
flg_ticket_update	252	2.96%
flg_mobile	220	2.59%
flg_printer	152	1.79%
flg_install	146	1.72%
flg_vpn	136	1.60%
flg_engineering	129	1.52%
mark_for_delete	118	1.39%
flg_crm	117	1.38%
host_related	117	1.38%
flg_help	114	1.34%
flg_skype	100	1.18%
blank_call	43	0.51%
setup_ws	33	0.39%
flg_windows	22	0.26%
Grand Total	8500	100.00%

EDA – With Key words









EDA – Summary

- **Language :**
  - English and German contribute to >97% of all tickets. Google translate is used for English translation
- **Special Characters :**
  - 10% of the records are having at least one Special Character.
  - Using EDA it was decided that any record having more than 50% of spl characters is not useful for training and hence deleted. For all others, the special characters were removed
- **Ticket Reception Mode:**
  - Multiple ticket reception modes were detected. Call, direct entry ,Email(internal and external) and monitoring tools.
  - It is important to note that monitoring tools can directly assign a ticket to L2/L3 based on the severity/priority.
- **Assignment Groups :**
  - As detected in the beginning 54% of the tickets are resolved by L1/L2 groups (Grp_0 and Grp_8).
  - There are total 74 groups in the dataset.
  - 9 out of 74 groups manage 75% of the tickets. So the distribution is very sparce.
  - At the tail end there are 41 groups managing 5% of the total tickets.
  - We need to club all these groups into one group called “Others” as it is impossible to train using this dataset.
- **Word Count :**
  - No of words in both the short description and Description has wide distribution. Using EDA we concluded to limit the no of words in Short description and Description as 15 and 100
  - Short description : >15 words : 345(4%) records
  - Description : >100 words : 386(4.54%) records

EDA – Summary (Contd..)

- **Key words :**
  - Key words were identified using word frequency (tokenization). Tickets are categorized based on the key words Interesting results were observed
    - Top categories are “account/password related” and “job related”. Both of them account for 49% of the tickets followed by **“erp”, “outlook”, “network related”**..etc.....,
    - Analysis of Assigned group vs the ticket category **“flg_account related”** resulted in 73% of the user account related tickets are handled by 3 groups (GRP_0, GRP_2, GRP_8).
    - 88% of job-related tickets are handled by 5 groups (Grp_8, Grp_9, Grp_6, Grp_5, Grp_10)
    - There were 9% of tickets where none of the key words were available

Key word analysis gave us interesting insights. Traditional automations uses the key word search to find the best fit category for a ticket assignment. This method has its own limitations. Unless the context is understood, it is not possible to assign a ticket to a particular group.

We can see this in the **“flg_account_related”** tickets. In the outset it may look like the tickets need to be assigned to L1/L2 but, only 73% of the tickets were resolved by them and the rest by L3.

It’s time we use advanced analytical techniques. New forms of AIML will go beyond the simple keyword search used by traditional methods. Using NLP and deep learning, tickets can be tagged/classified based on topic, issues, intent, sentiment or priority. We can use the tags within support software in combination with routing and prioritization rules.

It’s important that we make this leap to truly intelligent ticket classification because we are quickly dealing with lot of tickets and different varieties.

Basic Model Building – Comparing different Options

Model	Advantages	Limitations	Remark
Simple Rule Based Automation (Key word search)	• Easy to automate • Easy to understand the logic of classification • Can be incorporated in the existing	• Key words and rules keep changing • Complex when reception modes are multiple • Multiple languages makes it even more complicated • Context is not considered	• This is a traditional automation technique and is not very efficient ✗
Basic Machine Learning (Extract features using NLP and build an ML model using ensemble or other techniques)	• Simple models can be built using past data • Faster training	• Feature extraction makes the model rigid. • Complicated when reception modes are multiple (Email/Chat/Call..Etc.....) • Context is missing	• This is not suitable since the context is missing in the model and new features need to be added frequently ✗
Neural Networks with NLP (Bag of words)	• Perform better than simple ML models. • Can be used to train deeper	• Ordinary Neural Networks don't perform well in cases where sequence of data is important.	• Since the sequence of data is important (especially with multiple reception modes), we can not get better results ✗
Simple RNN's (Recurring Neural Networks)	• An RNN cell not only considers its present input but also the output of RNN cells preceding it, for its present output. • Understands the context	• Vanishing gradients problem • Exploding gradients problem • Long-Term Dependencies problem	• RNN is the answer for our problem. However, simple RNN may not work due to the mentioned disadvantages ✗
Special RNN's (LSTM) 	• capability of handling Long-Term dependencies. • Provide solution to Vanishing/Exploding Gradient problem	• Difficult to train because they require memory-bandwidth-bound computation, which is the worst nightmare for hardware designer • Limits the applicability of neural networks solutions based on Hardware	• We are going to start building a basic model using LSTM and establish baseline accuracy • Do simulations with different combinations to get the best possible accuracy ✓
Advanced Architectures (BERT, ELMO..etc....)	• More efficient than LSTM		• Need to be explored

Basic Model Building - LSTM

Ordinary Neural Networks don't perform well in cases where sequence of data is important. To overcome this, RNNs were invented. RNN stands for "Recurrent Neural Network". An RNN cell not only considers its present input but also the output of RNN cells preceding it, for its present output.

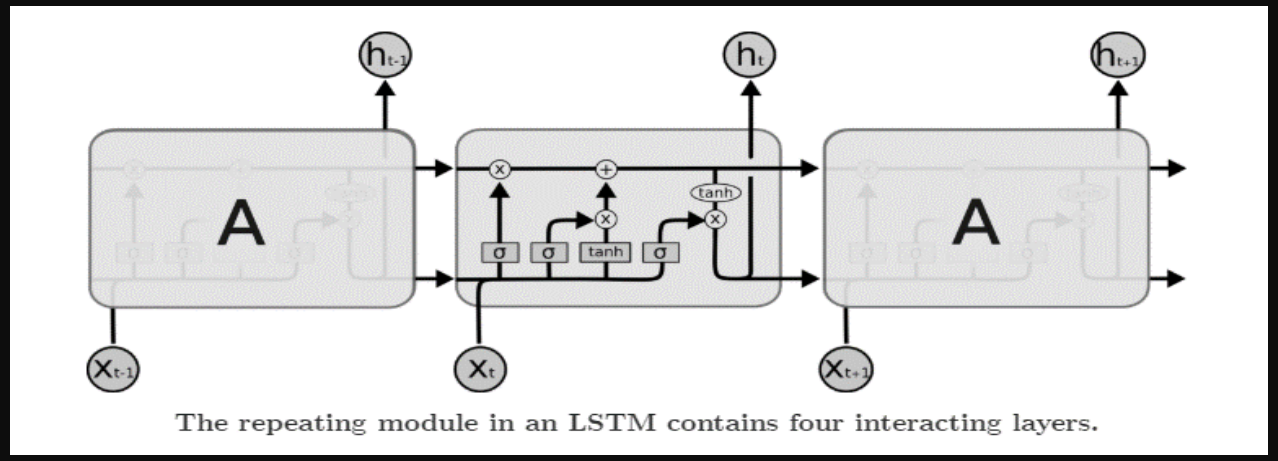
It is proved that RNNs perform very well on sequential data and on tasks where sequence was important. In ticket classification, sequence of the words is important to understand the right classification and severity of the issue so that the ticket can be assigned to the right resolution group (L1/L2/L3)

But there exists many problems with ordinary RNNs

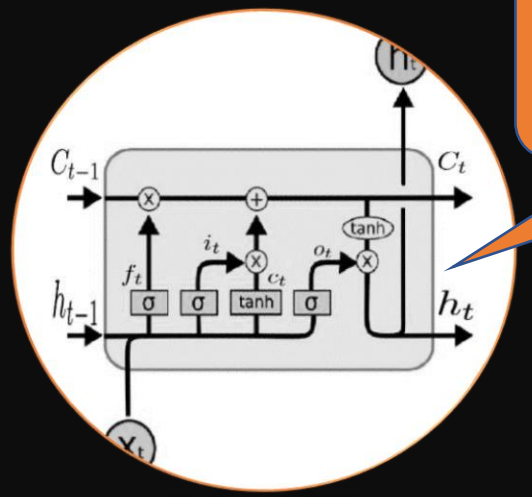
- **Vanishing gradients problem** : During backpropagation, as gradient is calculated by chain rule, it has an effect of multiplying these small numbers which squeezes the final gradient to almost zero and hence subtracting gradient from weights doesn't make any change to them which stops the training of model.
- **Exploding gradients problem :** Opposite to vanishing gradient problem, while following chain rule we multiply with the weight matrix too at each step, and if the values are larger than 1, multiplying a large number to itself many times leads to a very large number leading to explosion of gradient.
- **Long-Term Dependencies problem :** RNNs are good in handling sequential data but they run into problem when the context is far away.

Long Short Term Memory Networks : LSTMs are special kind of RNNs with capability of handling Long-Term dependencies. LSTMs also provide solution to Vanishing/Exploding Gradient problem. A simple LSTM cell looks like this

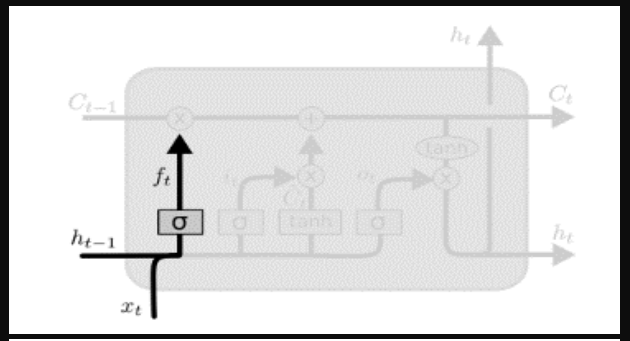
Basic Model Building – LSTM Architecture



The repeating module in an LSTM contains four interacting layers.

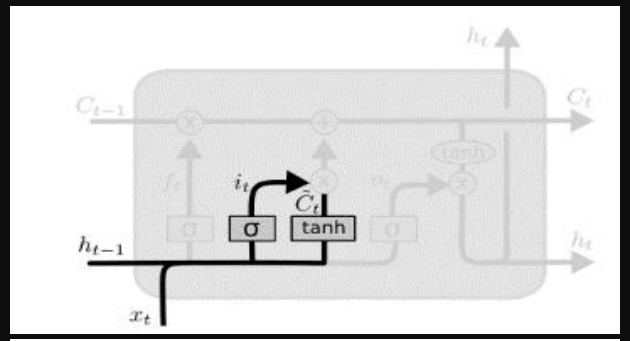


LSTM Cell
f: Forget gate : Where to erase cell
i: Input Gate, whether to write to cell
g: Gate gate(?), how much to write to cell
o : Output gate, How much to reveal cell



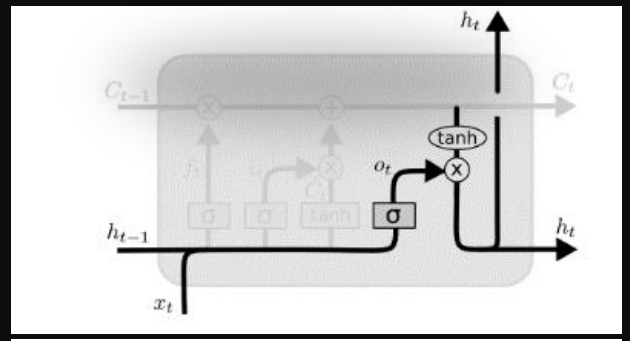
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Forget Gate: After getting the output of **previous state**, **h(t-1)**, Forget gate helps us to take decisions about what must be removed from h(t-1) state and thus keeping only relevant stuff. It is surrounded by a sigmoid function which helps to crush the input between [0,1]



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Input Gate: In the input gate, we decide to add new stuff from the present input to our present cell state scaled by how much we wish to add them. In the above photo, **sigmoid layer decides which values to be updated** and **tanh layer creates a vector for new candidates to added to present cell state**



$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Output Gate: Finally we'll decide what to output from our cell state which will be done by our sigmoid function.
We multiply the input with tanh to crush the values between (-1,1) and then multiply it with the output of sigmoid function so that we only output what we want to.

Data Preparation for Model Building

We created the below 5 columns from our dataset for model building

trimmed_words_short : Processed data from the original Columns “Short description”

trimmed_words_long : Processed data from the original column Description

Assignment group : Original column “Assignment group”

new_group_75 : There are groups which contribute to 75% tickets. Rest all groups are clubbed as Others

new_group_85 : There are groups which contribute to 85% tickets. Rest all groups are clubbed as Others

new_group_95 : There are groups which contribute to 95% tickets. Rest all groups are clubbed as Others

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8379 entries, 0 to 8378
Data columns (total 6 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   trimmed_words_short   8379 non-null   object
 1   trimmed_words_long    8379 non-null   object
 2   Assignment group      8379 non-null   object
 3   new_group_75          8379 non-null   object
 4   new_group_85          8379 non-null   object
 5   new_group_95          8379 non-null   object
```

We are going to run the model with all the possible combinations of trimmed_words_shot/trimmed_words_long and the 4 different assignment groups

Basic Model Building – LSTM Architecture

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
L1_Embedding (Embedding)     (None, 200, 200)          400000
_________________________________________________________________
unified_lstm_8 (UnifiedLSTM) (None, 200, 128)          168448
_________________________________________________________________
L3_Flatten (Flatten)         (None, 25600)             0
_________________________________________________________________
L4_Dense_relu (Dense)        (None, 254)               6502654
_________________________________________________________________
L6_Dense_sigmoid (Dense)     (None, 74)                18870
=================================================================
Total params: 7,089,972
Trainable params: 7,089,972
Non-trainable params: 0
_________________________________________________________________
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_25 (InputLayer)        [(None, 200)]             0
_________________________________________________________________
embedding_23 (Embedding)     (None, 200, 200)          400000
_________________________________________________________________
lstm_22 (LSTM)               (None, 200, 100)          120400
_________________________________________________________________
flatten_22 (Flatten)         (None, 20000)             0
_________________________________________________________________
dropout_22 (Dropout)         (None, 20000)             0
_________________________________________________________________
dense_22 (Dense)             (None, 74)                1480074
=================================================================
Total params: 2,000,474
Trainable params: 2,000,474
Non-trainable params: 0
_________________________________________________________________
```

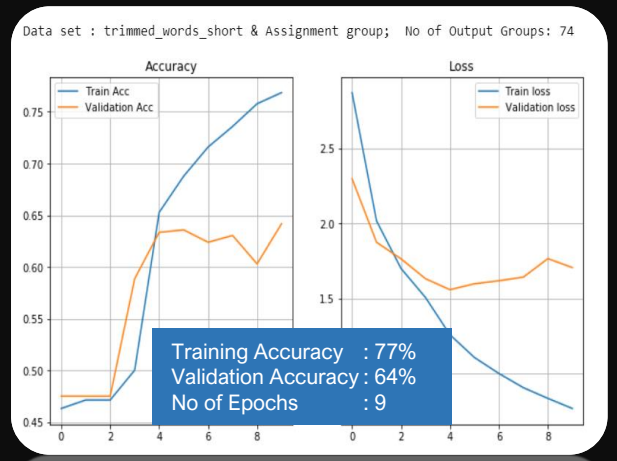
```
maxlen =200
embedding_size = 200
LEARNING_RATE = 1e-3
Units = 100
Drop=0.1
EPOCHS = 1
BATCH_SIZE = 100
#glove=0
numword = 2000
embeddings = {}
embedding_matrix = np.zeros((numword, embedding_size))
t = Tokenizer(num_words=numword)
```

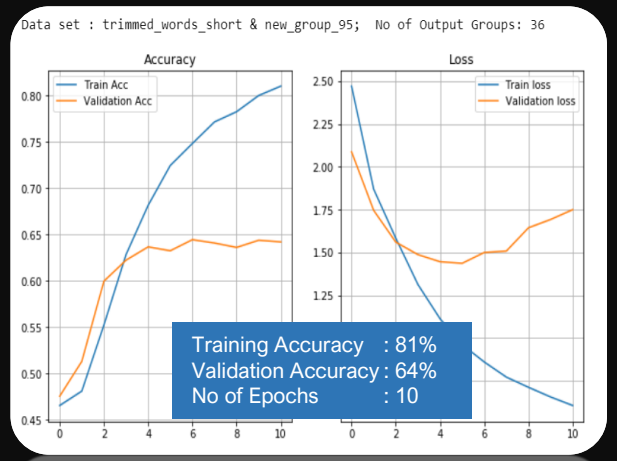
Run the model –

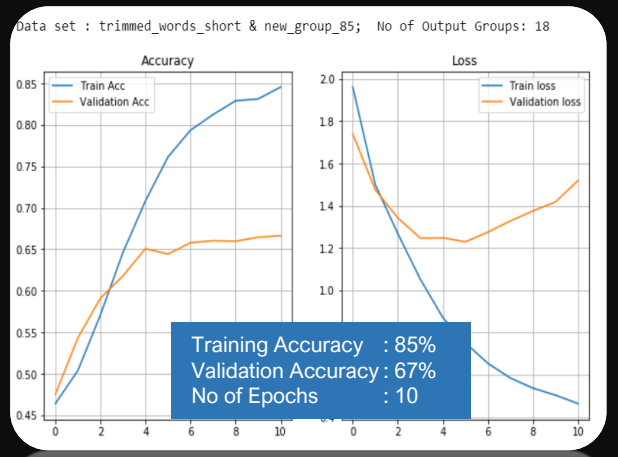
- Without any weight matrix
- With pre-trained GloVe vector

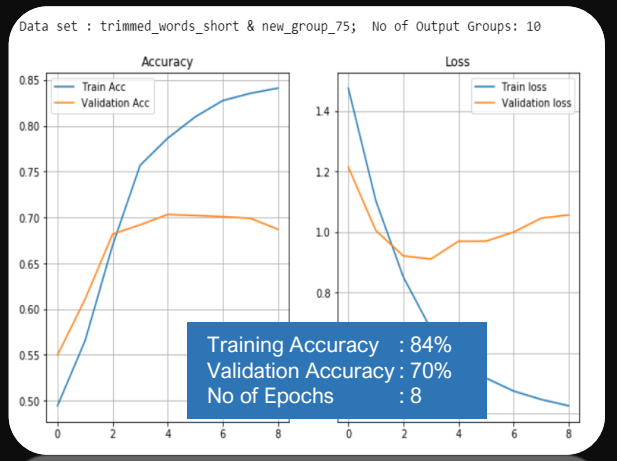
Basic Model Building – Accuracy/Loss plots

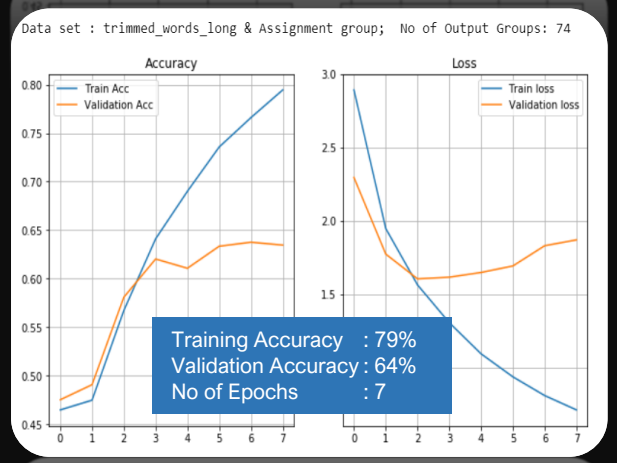
Model built with out any pre-trained weights in the Embedding layer

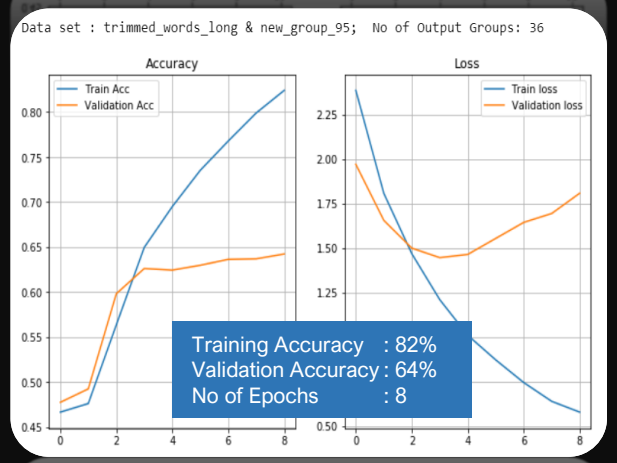


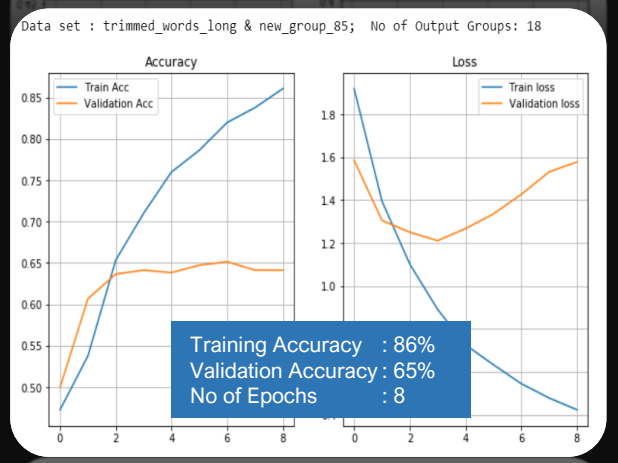


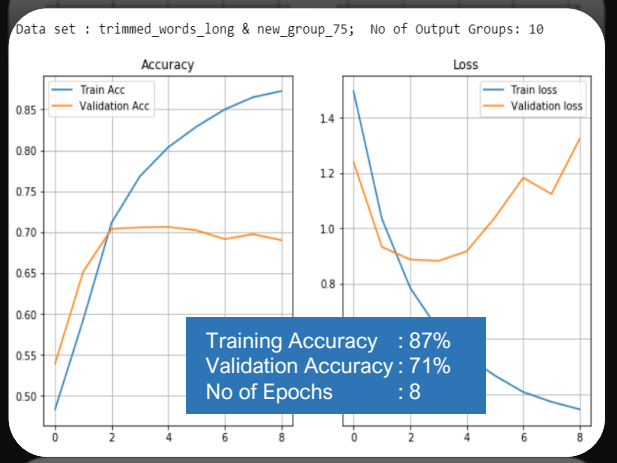






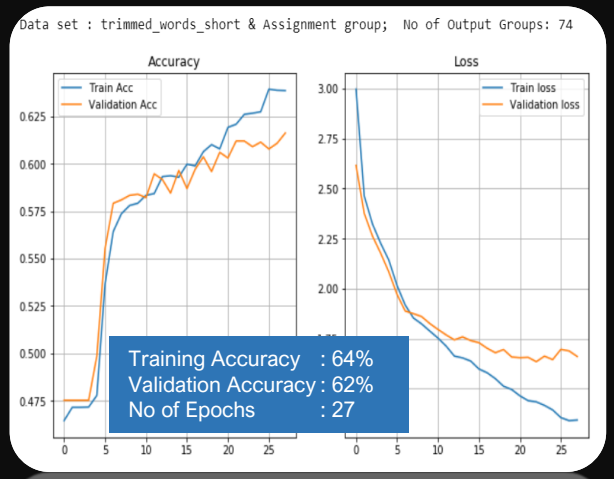


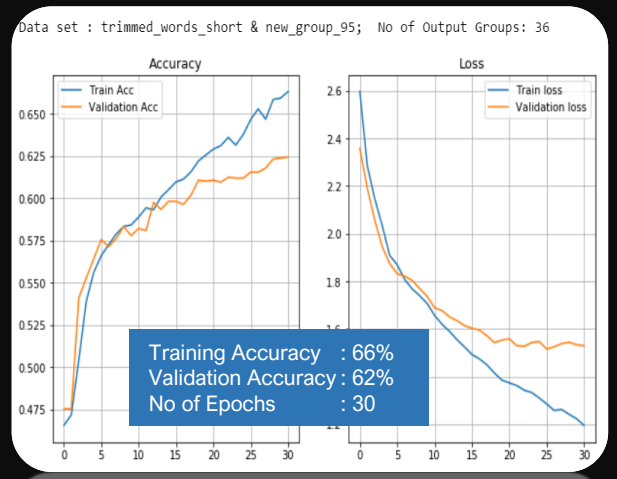


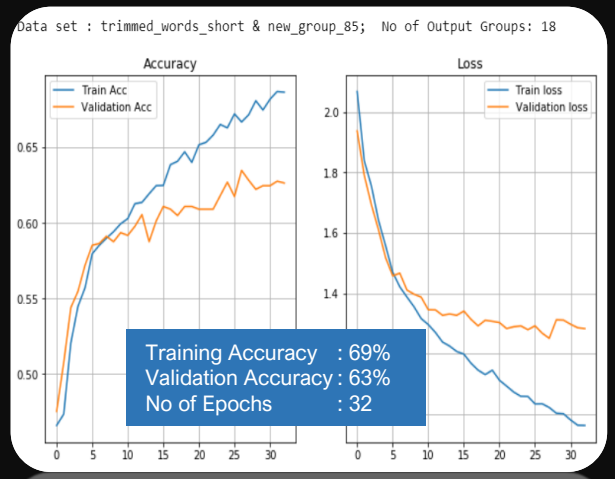


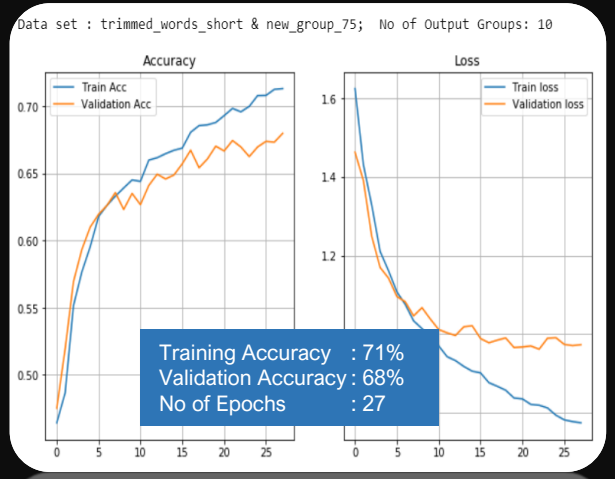
Basic Model Building – Accuracy/Loss plots

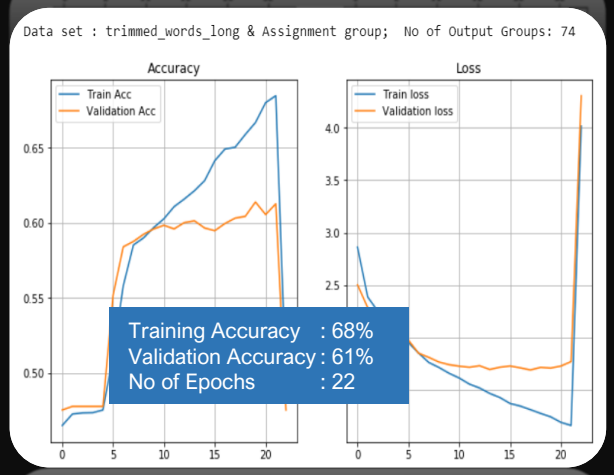
Model built using pre-trained GloVe Vector in the Embedding layer

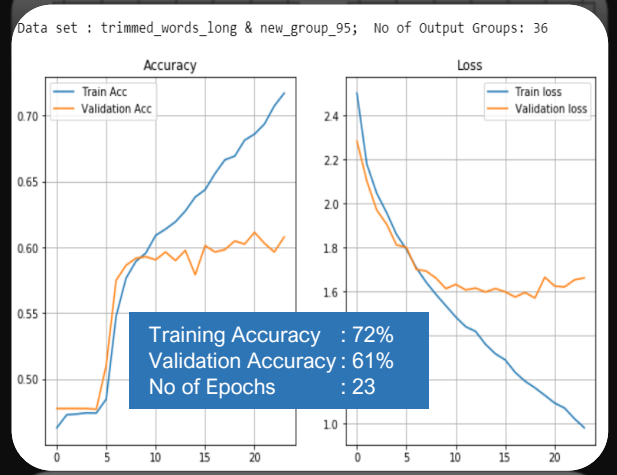


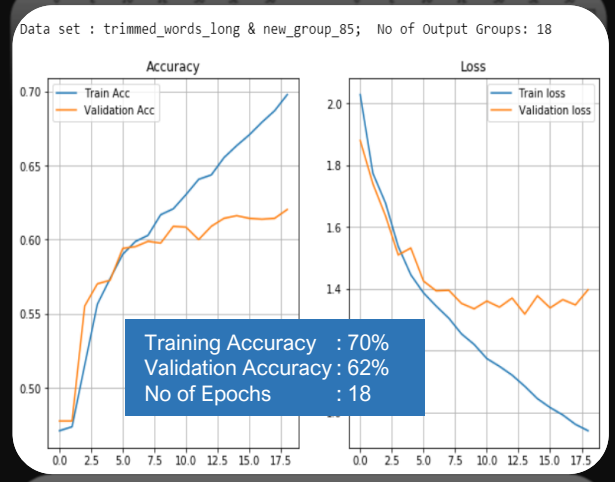


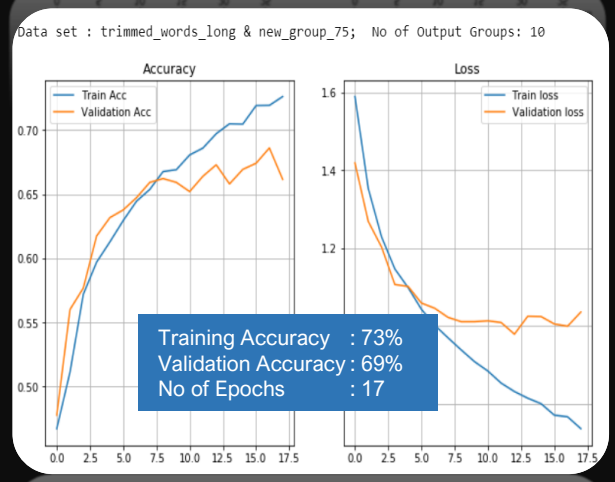












Basic Model Building – Baseline Results

Model results with out using GloVe Vector

	Description Col	Group Col	Dense Group	Training Acc	Validation Acc	Training Loss	Validation Loss
0	trimmed_words_short	Assignment group	74	0.77	0.64	2.87	2.3
3	trimmed_words_short	new_group_95	36	0.81	0.64	2.47	2.09
2	trimmed_words_short	new_group_85	18	0.85	0.67	1.96	1.74
1	trimmed_words_short	new_group_75	10	0.84	0.7	1.48	1.21
4	trimmed_words_long	Assignment group	74	0.79	0.64	2.89	2.3
7	trimmed_words_long	new_group_95	36	0.82	0.64	2.39	1.97
6	trimmed_words_long	new_group_85	18	0.86	0.65	1.92	1.59
5	trimmed_words_long	new_group_75	10	0.87	0.71	1.5	1.32

Model results using GloVe Vector

	Description Col	Group Col	Dense Group	Training Acc	Validation Acc	Training Loss	Validation Loss
0	trimmed_words_short	Assignment group	74	0.64	0.62	3	2.62
3	trimmed_words_short	new_group_95	36	0.66	0.62	2.6	2.36
2	trimmed_words_short	new_group_85	18	0.69	0.63	2.07	1.94
1	trimmed_words_short	new_group_75	10	0.71	0.68	1.63	1.46
4	trimmed_words_long	Assignment group	74	0.68	0.61	4.01	4.3
7	trimmed_words_long	new_group_95	36	0.72	0.61	2.5	2.28
6	trimmed_words_long	new_group_85	18	0.7	0.62	2.03	1.88
5	trimmed_words_long	new_group_75	10	0.73	0.69	1.59	1.42

Observations :

- Using ‘Long Description’ we got better accuracy than the short description. However, the difference between training accuracy and validation is higher in case of long description, understandably due to the sparsity of the data.
- We need to find ways to use information from both the short and long description to improve the model further
- When the model was run without pre-trained weight matrix in the embedding layer, model got overfit very quickly just in 8 to 10 Epochs. Also, we found that there is a wide gap between the training accuracy and the validation accuracy.
- However, when the model was built using GloVe pre-trained matrix, we got lesser accuracy, but generalization loss is lower indicating the model is getting robust.
- We got best accuracy with the target variable “assignment_group_75” which is having only 10 output groups (75% tickets are assigned to 9 groups and rest all the 65 groups were clubbed under others)

How to Improve Model Accuracy Further ?

Try the below model combinations

- Bi-LSTM with CNN Layer
- Modified Encoder-Decoder Network
- Modified Encoder-Decoder Network with Additional Features extracted from text
- Modified Encoder-Decoder Network with Attention layer

Bi-LSTM with CNN layer

The combination of CNN and RNN models proved to be effective, since each model has a specific architecture and its own strengths:

- CNN is known for its ability to extract as many features as possible from the text.
- LSTM/Bi-LSTM keeps the chronological order between words in a document, thus having the ability to ignore unnecessary words using the delete gate.
- The purpose of combining these two models is to create a model that takes advantage of the strengths of CNN and BiLSTM, so that it captures the features extracted using CNN, and uses them as an LSTM input.

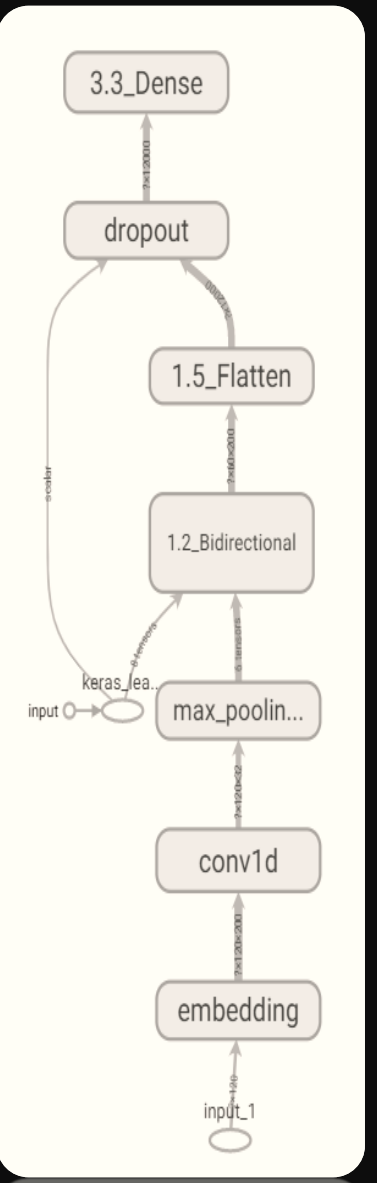
We develop a model that meets this objective, such that the vectors built in the word embeddings are used as convolutional neural network input. Then, the results of all max pooling layers are concatenated to build the BiLSTM input, which applies a BiLSTM layer to filter the information, using its three gates.

The output of this step is the input of the fully connected layer, which links each piece of input information with a piece of output information. Finally, we apply the sigmoid function as an activation function to assign classes

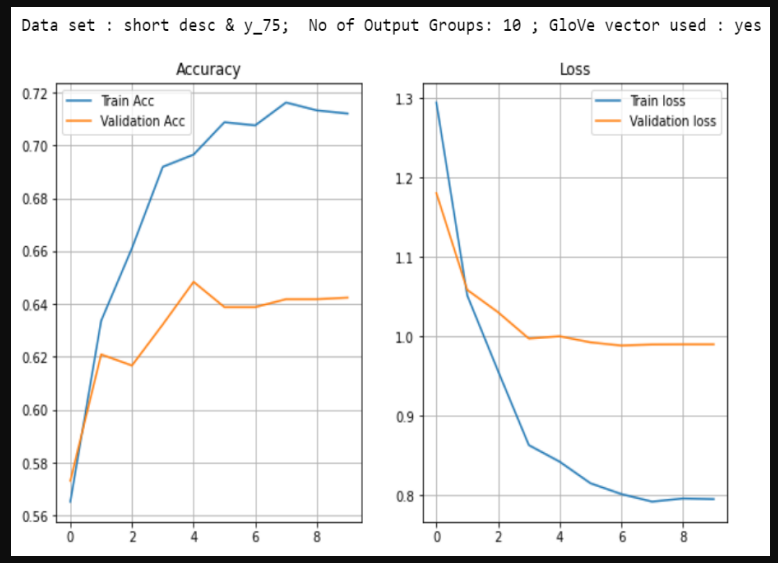
```
Tensor("1.0_Input_Layer_1:0", shape=(?, 100), dtype=int32)
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
1.0_Input_Layer (InputLayer) [(None, 100)]             0
_________________________________________________________________
1.1_Embedding (Embedding)    (None, 100, 200)          2600000
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 100, 32)           19232
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 50, 32)            0
_________________________________________________________________
1.2_Bidirectional (Bidirecti (None, 50, 256)           164864
_________________________________________________________________
1.5_Flatten (Flatten)        (None, 12800)             0
_________________________________________________________________
dropout_1 (Dropout)          (None, 12800)             0
_________________________________________________________________
3.3_Dense (Dense)            (None, 10)                128010
=================================================================
Total params: 2,912,106
Trainable params: 2,912,106
Non-trainable params: 0
```

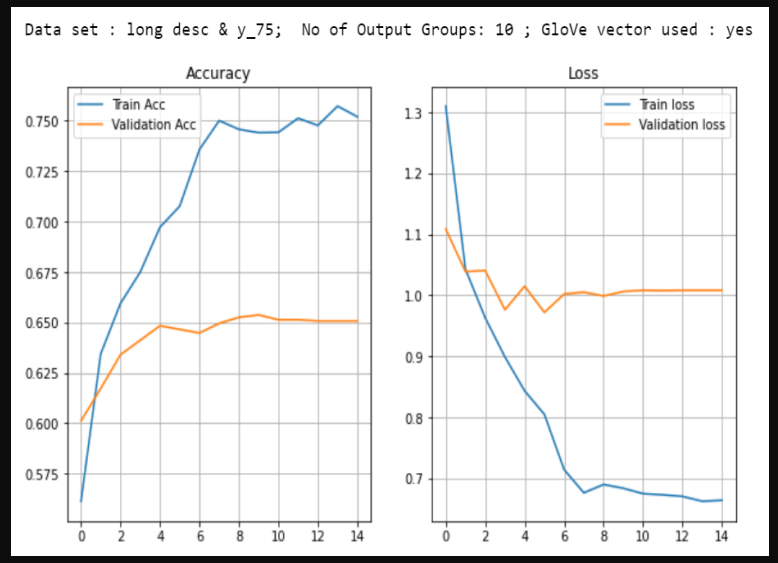
```
# Run the model....
reduce_lr = (ReduceLROnPlateau(monitor='val_acc', factor=0.2,
                               patience=1, min_lr=0.00000001))
early_stop = (EarlyStopping(monitor='val_acc', patience= 3))

history = model.fit(X_train_combo, y_train_75,
                    validation_data=[X_val_combo,y_val_75],
                    epochs=50, batch_size=4,callbacks= ([reduce_lr,early_stop]), verbose = 1)
```

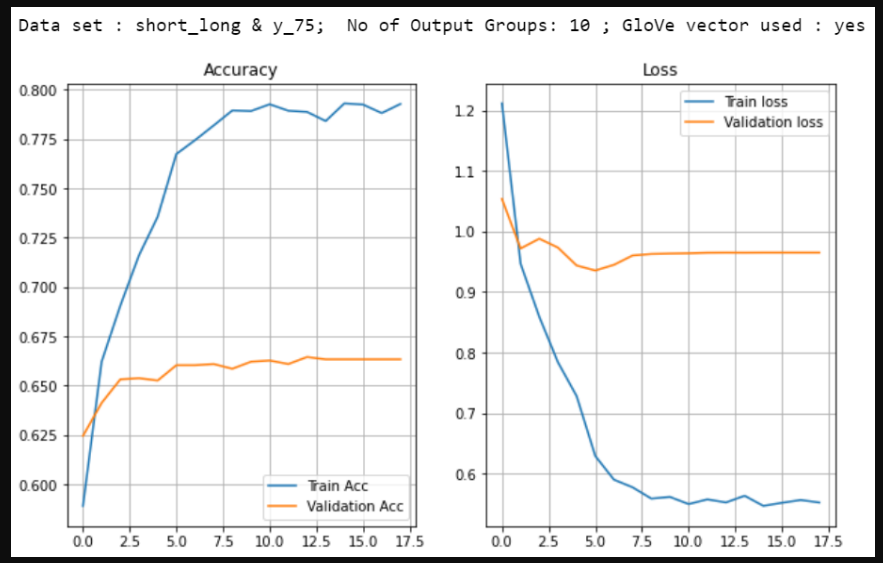
```
Epoch 9/50
6704/6704 [==============================] - 210s 31ms/sample - loss: 0.4105 - acc: 0.8511 - mean_squared_error: 0.0618 - val_loss: 0.8723 - val_acc: 0.7261 - val_mean_squared_error: 0.0717
Epoch 10/50
6704/6704 [==============================] - 200s 30ms/sample - loss: 0.4002 - acc: 0.8570 - mean_squared_error: 0.0630 - val_loss: 0.8789 - val_acc: 0.7255 - val_mean_squared_error: 0.0713
Epoch 11/50
6704/6704 [==============================] - 198s 30ms/sample - loss: 0.4009 - acc: 0.8531 - mean_squared_error: 0.0634 - val_loss: 0.8749 - val_acc: 0.7237 - val_mean_squared_error: 0.0715
Epoch 12/50
6704/6704 [==============================] - 199s 30ms/sample - loss: 0.4010 - acc: 0.8538 - mean_squared_error: 0.0633 - val_loss: 0.8748 - val_acc: 0.7237 - val_mean_squared_error: 0.0716
```



Bi-LSTM with CNN layer







input_x	target_y	unique_target	training_acc	validation_acc	training_loss	validation_loss	glove_vector	attn_layer	conv1d	_of_epochs_	batch_size	learning_rate
short desc	y_75	10	0.847	0.645	0.432	1.365	yes	no	yes	14	16	3.20E-07
long desc	y_75	10	0.865	**0.655**	0.387	1.305	no	no	yes	12	16	6.40E-08
short_long	y_75	10	0.936	0.676	0.167	1.619	no	no	yes	16	16	1.28E-08
short desc	y_75	10	0.712	0.642	0.795	0.990	no	no	yes	10	16	3.20E-07
long desc	y_75	10	0.752	**0.651**	0.664	1.008	yes	no	yes	15	16	6.40E-08
short_long	y_75	10	0.793	0.663	0.553	0.965	yes	no	yes	18	16	1.00E-08

Observations : It was observed that the model was not able to improve the accuracy beyond 65% and running the model further is quickly overfitting. It seems the model is not able to understand the context of the text and hence failing beyond a point to improve the accuracy. Adding attention layer or an Encoder/Decoder network may help to improve the accuracy further

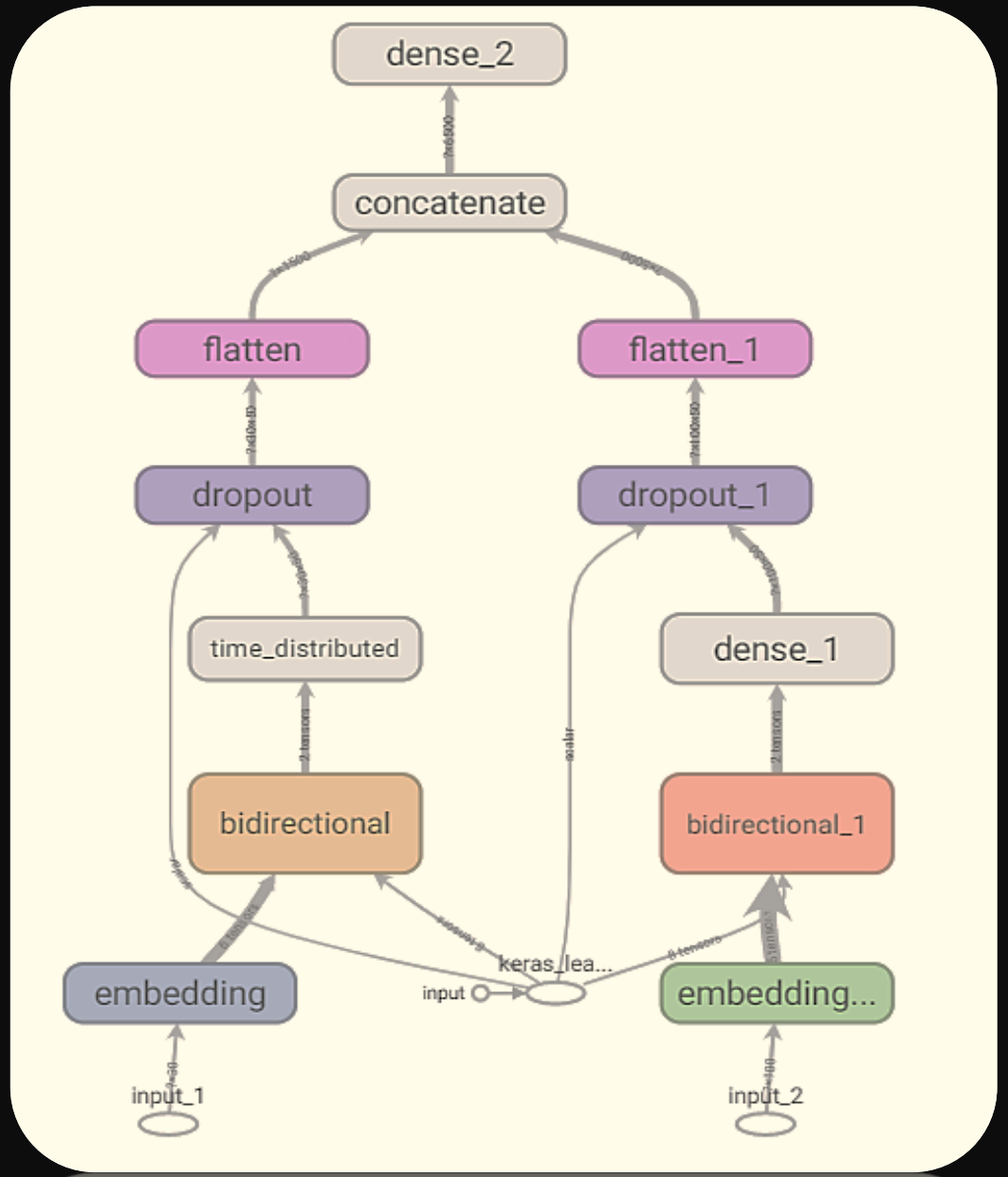
Modified Encoder-Decoder Network

- Used a modified encoder-decoder network to take advantage of the both Short and Long descriptions we have in the dataset.
- Network is constructed by having a Bi-LSTM and Dense layer for Short description and a similar combination for long description and concatenate both the outputs and have another dense layer.
- By doing this we are helping the model to encode the text from both the short and long description and get the flexibility for the model to train using the entire dataset.

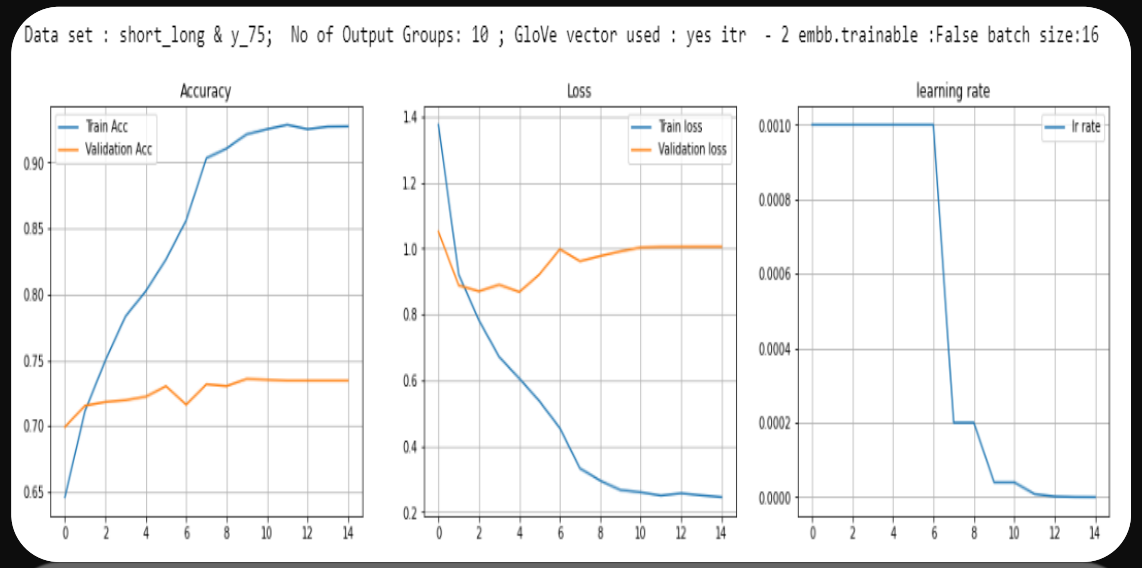
Model: "model"

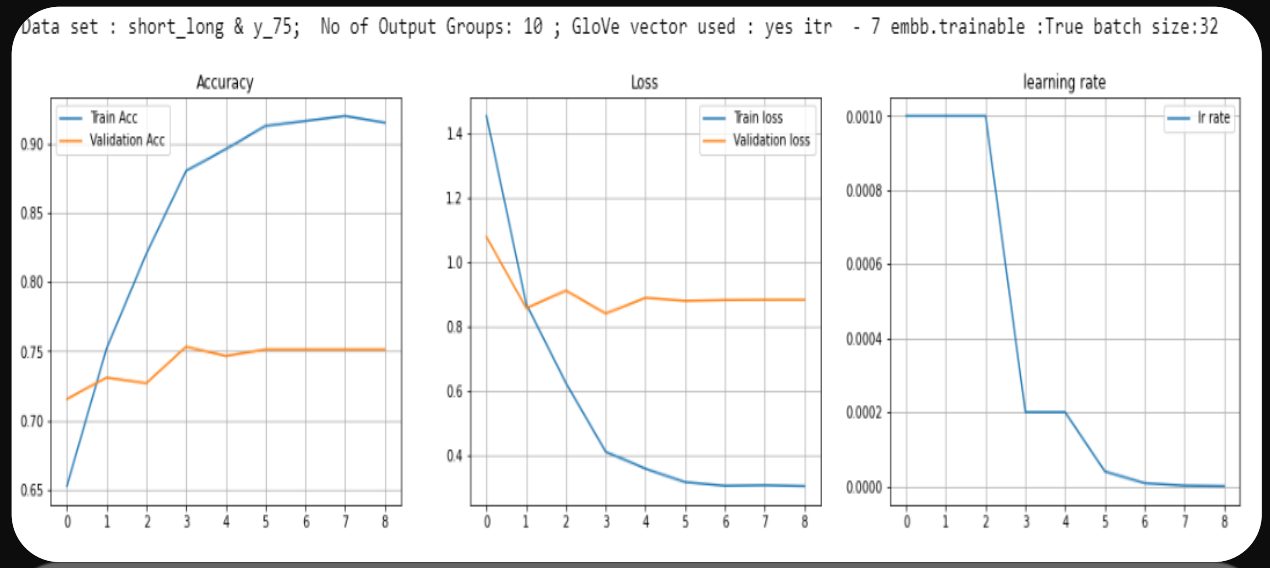
Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 30)]	0	
input_2 (InputLayer)	[(None, 100)]	0	
embedding (Embedding)	(None, 30, 200)	1400000	input_1[0][0]
embedding_1 (Embedding)	(None, 100, 200)	1400000	input_2[0][0]
bidirectional (Bidirectional)	(None, 30, 100)	100400	embedding[0][0]
bidirectional_1 (Bidirectional)	(None, 100, 100)	100400	embedding_1[0][0]
time_distributed (TimeDistribut	(None, 30, 50)	5050	bidirectional[0][0]
dense_1 (Dense)	(None, 100, 50)	5050	bidirectional_1[0][0]
dropout (Dropout)	(None, 30, 50)	0	time_distributed[0][0]
dropout_1 (Dropout)	(None, 100, 50)	0	dense_1[0][0]
flatten (Flatten)	(None, 1500)	0	dropout[0][0]
flatten_1 (Flatten)	(None, 5000)	0	dropout_1[0][0]
concatenate (Concatenate)	(None, 6500)	0	flatten[0][0] flatten_1[0][0]
dense_2 (Dense)	(None, 10)	65010	concatenate[0][0]

Total params: 3,075,910
Trainable params: 275,910
Non-trainable params: 2,800,000



Modified Encoder-Decoder Network





input_x	target_y	unique_targe	training_acc	validation_ac	training_loss	validation_los	glove_vecto	attn_layer	conv1c	no_of_epochs_ru	batch_size	learning_rate	embb_trainable
short_long	y_75	10	0.884	0.724	0.365	0.883	yes	no	yes	12	8	6.40E-08	FALSE
short_long	y_75	10	**0.927**	**0.734**	0.246	1.005	yes	no	yes	15	16	6.40E-08	FALSE
short_long	y_75	10	0.850	0.726	0.478	0.852	yes	no	yes	12	32	3.20E-07	FALSE
short_long	y_75	10	0.782	0.713	0.741	0.899	yes	no	yes	13	100	1.28E-08	FALSE
short_long	y_75	10	0.966	**0.751**	0.132	1.010	yes	no	yes	12	8	1.28E-08	TRUE
short_long	y_75	10	0.960	0.743	0.167	0.953	yes	no	yes	8	16	1.60E-06	TRUE
short_long	y_75	10	**0.915**	**0.751**	0.305	0.883	yes	no	yes	9	32	3.20E-07	TRUE
short_long	y_75	10	0.952	0.742	0.216	1.047	yes	no	yes	11	100	1.60E-06	TRUE

Observations : Model accuracy improved beyond 73%. If we trained the model deeper than 15 epochs, model is getting overfit.

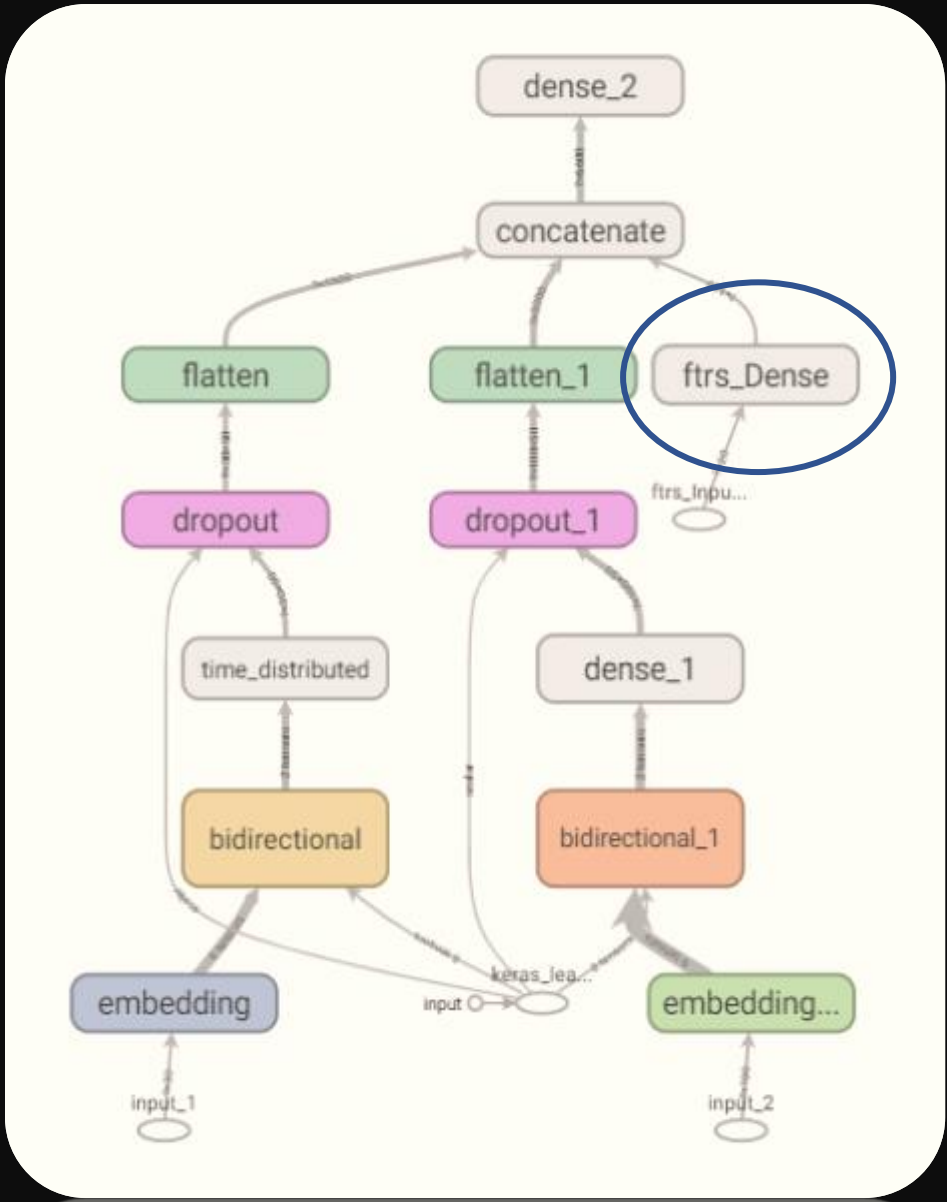
Modified Encoder-Decoder Network With Additional Features

- During pre-processing of the data, we extracted several additional features from the text. In this model we will add those features as an additional input and train the model.
- The Network diagram on the right side shows the additional input layer added to the model (20 features)

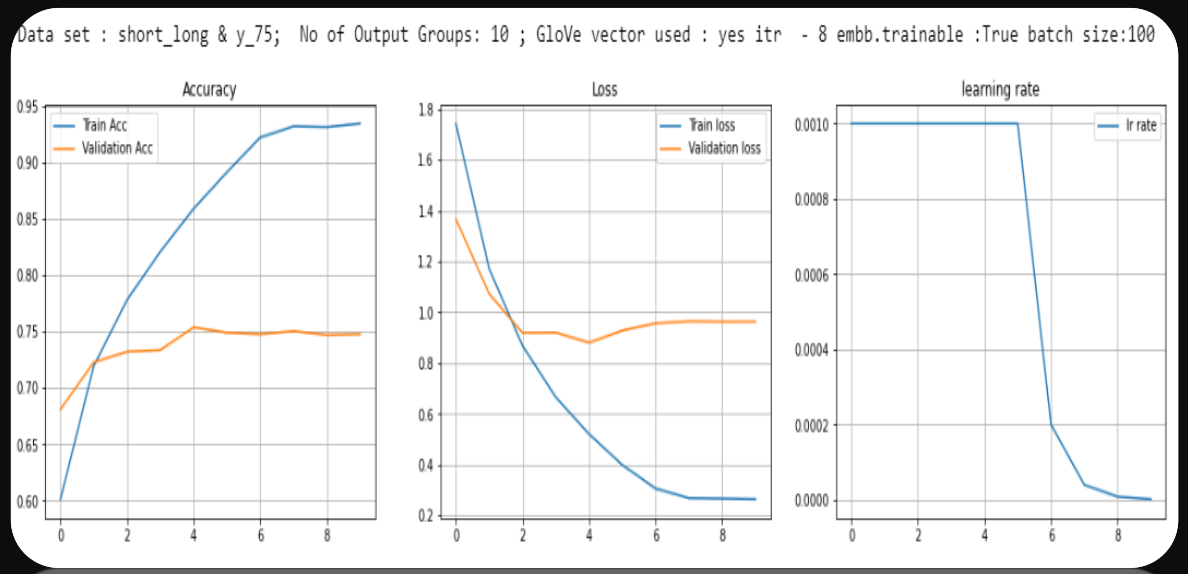
Model: "model_3"

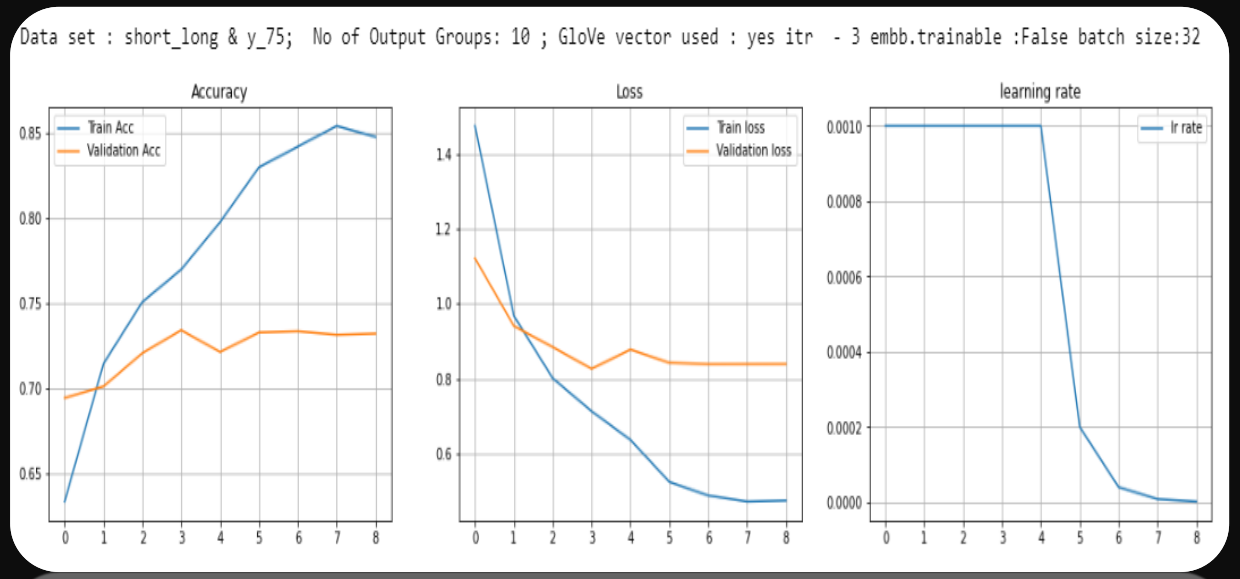
Layer (type)	Output Shape	Param #	Connected to
input_7 (InputLayer)	[(None, 30)]	0	
input_8 (InputLayer)	[(None, 100)]	0	
embedding_6 (Embedding)	(None, 30, 200)	1400000	input_7[0][0]
embedding_7 (Embedding)	(None, 100, 200)	1400000	input_8[0][0]
bidirectional_6 (Bidirectional)	(None, 30, 100)	100400	embedding_6[0][0]
bidirectional_7 (Bidirectional)	(None, 100, 100)	100400	embedding_7[0][0]
time_distributed_3 (TimeDistrib	(None, 30, 50)	5050	bidirectional_6[0][0]
dense_10 (Dense)	(None, 100, 50)	5050	bidirectional_7[0][0]
dropout_6 (Dropout)	(None, 30, 50)	0	time_distributed_3[0][0]
dropout_7 (Dropout)	(None, 100, 50)	0	dense_10[0][0]
ftrs_Input_layer (InputLayer)	[(None, 20)]	0	
flatten_6 (Flatten)	(None, 1500)	0	dropout_6[0][0]
flatten_7 (Flatten)	(None, 5000)	0	dropout_7[0][0]
ftrs_Dense (Dense)	(None, 100)	2100	ftrs_Input_layer[0][0]
concatenate_3 (Concatenate)	(None, 6600)	0	flatten_6[0][0] flatten_7[0][0] ftrs_Dense[0][0]
dense_11 (Dense)	(None, 10)	66010	concatenate_3[0][0]

Total params: 3,079,010
Trainable params: 279,010
Non-trainable params: 2,800,000



Modified Encoder-Decoder Network With Additional Features



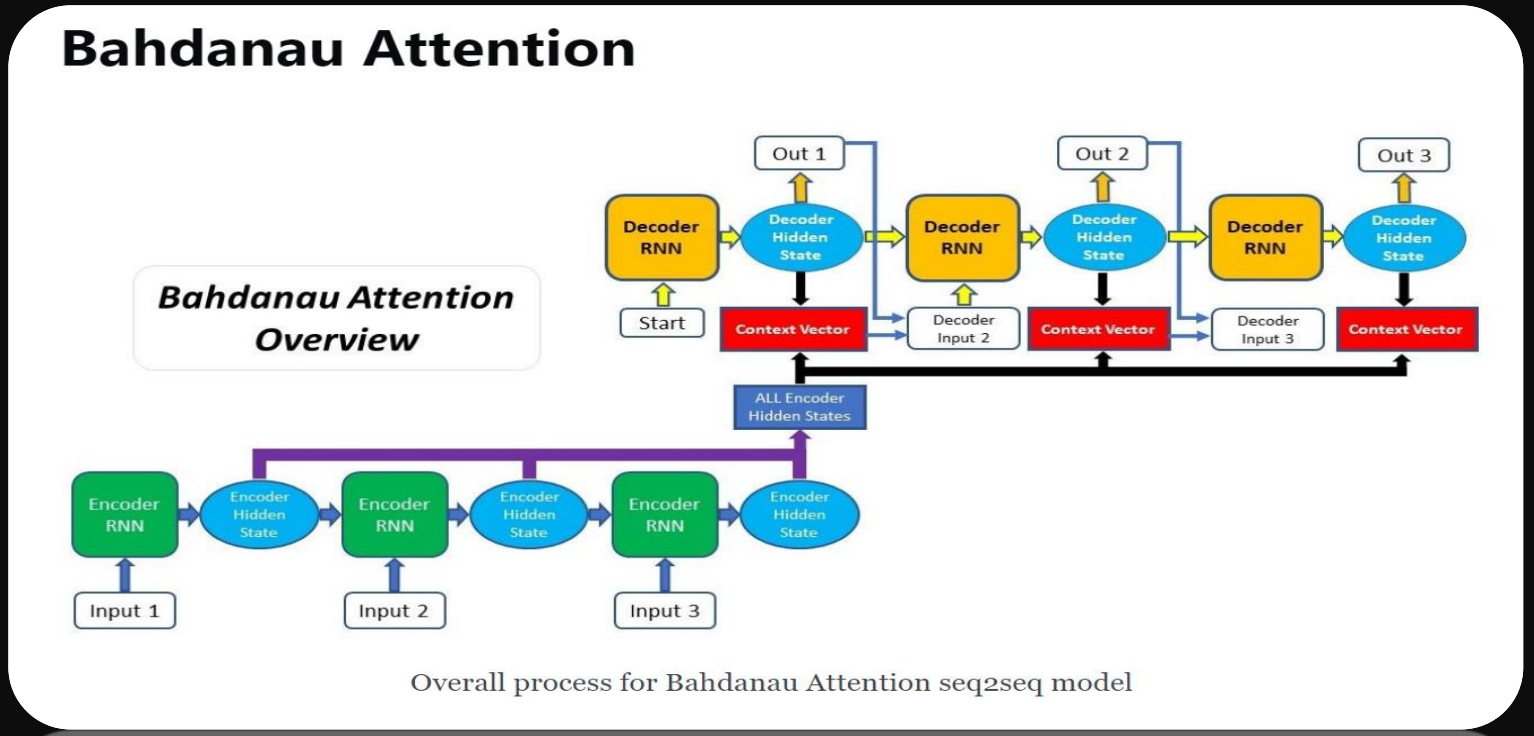


input_x	target_y	unique_target	training_acc	validation_acc	training_loss	validation_loss	glove_vector	attn_layer	conv1d	no_of_epochs_ru	batch_size	learning_rate	embb_trainable
short_long	y_75	10	0.898	0.732	0.325	0.940679	yes	no	yes	11	8	3.20E-07	FALSE
short_long	y_75	10	0.884	0.734	0.360	0.889318	yes	no	yes	17	16	1.00E-08	FALSE
short_long	y_75	10	**0.848**	**0.732**	0.474	0.838796	yes	no	yes	9	32	1.60E-06	FALSE
short_long	y_75	10	0.799	0.727	0.675	0.880993	yes	no	yes	11	100	3.20E-07	FALSE
short_long	y_75	10	0.959	0.755	0.150	0.996531	yes	no	yes	11	8	6.40E-08	TRUE
short_long	y_75	10	0.959	0.742	0.170	0.992903	yes	no	yes	8	16	1.60E-06	TRUE
short_long	y_75	10	0.950	0.744	0.194	1.002487	yes	no	yes	11	32	3.20E-07	TRUE
short_long	y_75	10	**0.935**	**0.748**	0.264	0.96317	yes	no	yes	10	100	1.60E-06	TRUE

Observations : While the validation accuracy is little lesser than our previous model, it is interesting to note that the model is more robust and generalized compared to the previous one.

Addition of attention layer to Encoder-Decoder network

The seq2seq models is normally composed of an encoder-decoder architecture, where the encoder processes the input sequence and encodes/compresses/summarizes the information into a context vector (also called as the “thought vector”) of a fixed length. This representation is expected to be a good summary of the entire input sequence. The decoder is then initialized with this context vector, using which it starts generating the transformed output. ***A critical and apparent disadvantage of this fixed-length context vector design is the incapability of the system to remember longer sequences. Often is has forgotten the earlier parts of the sequence once it has processed the entire the sequence. The attention mechanism was born to resolve this problem.***



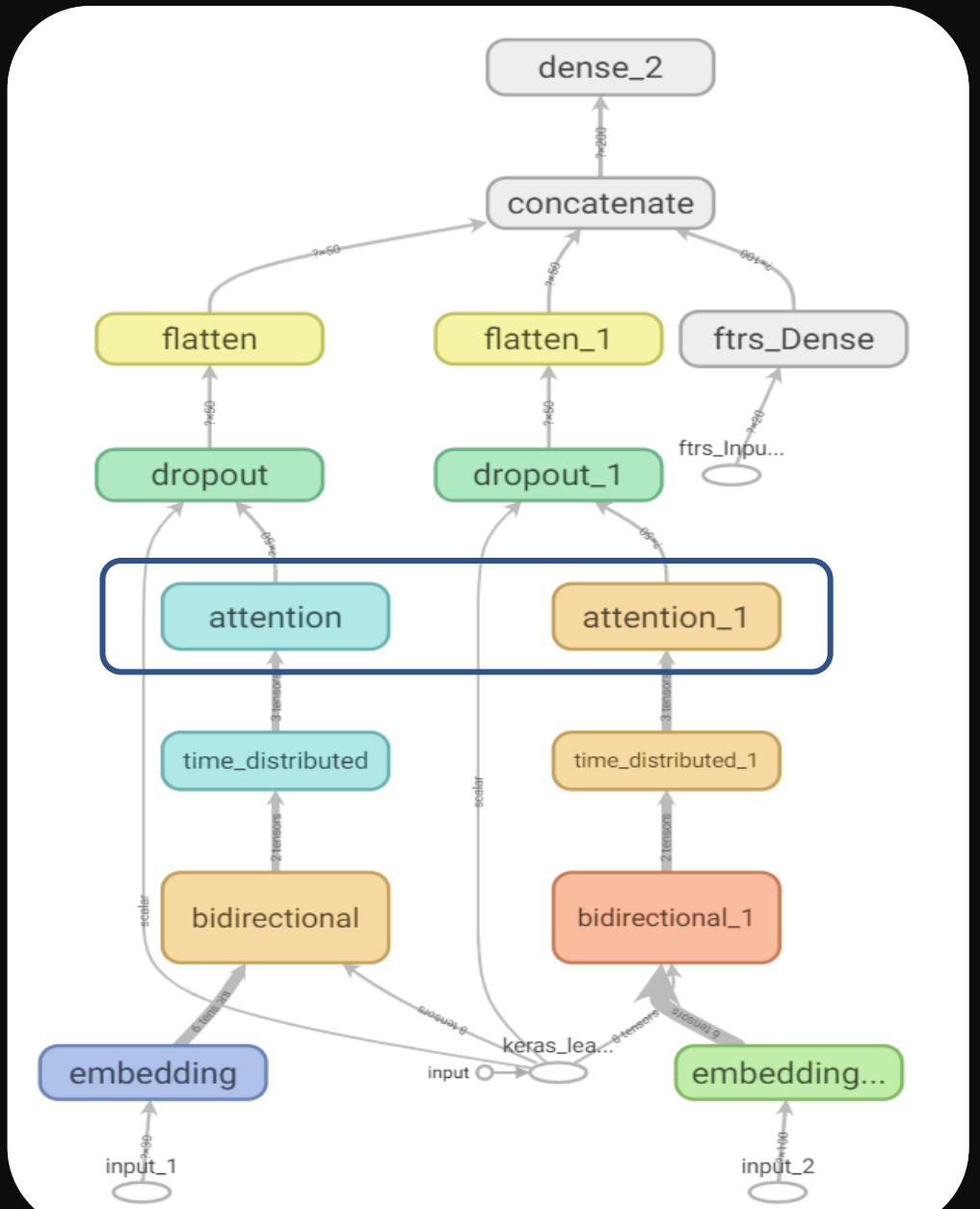
Overall process for Bahdanau Attention seq2seq model

Source: https://blog.floydhub.com/attention-mechanism/

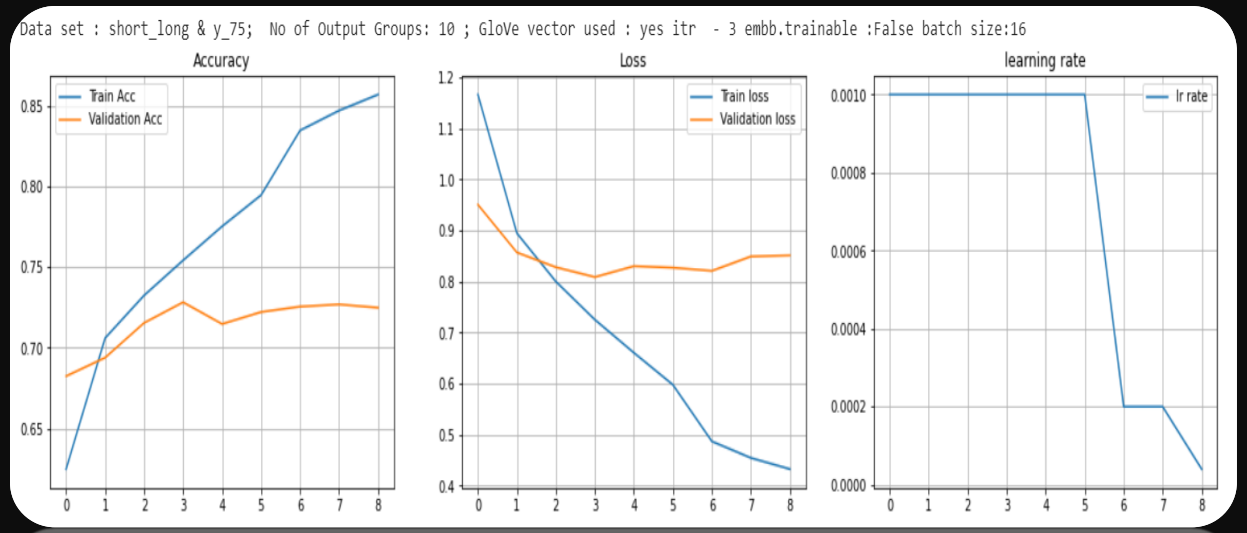
With Attention Layer

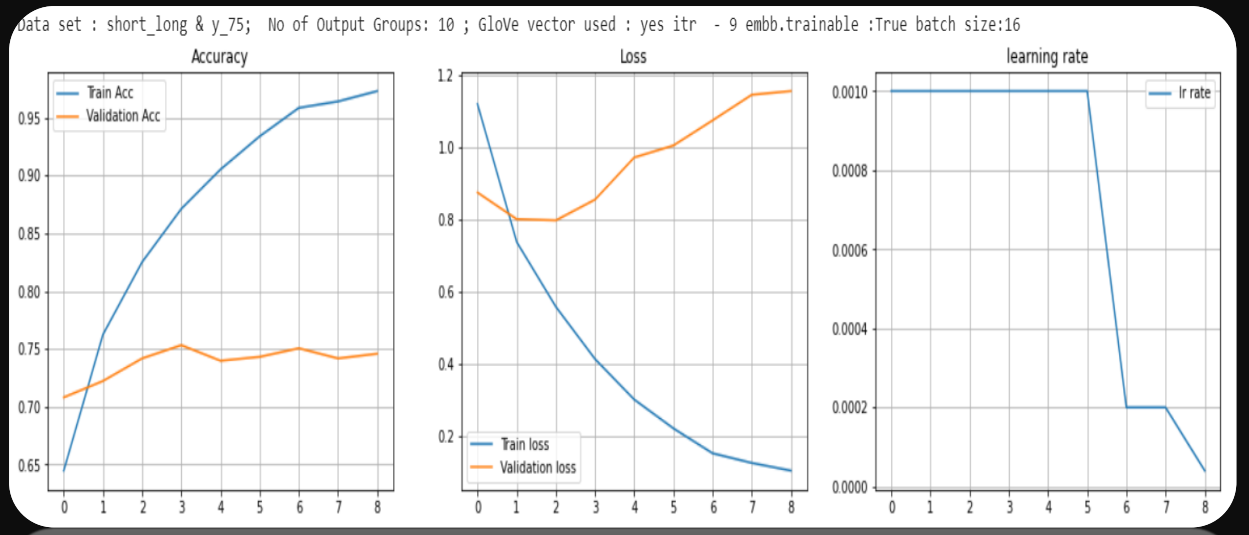
- To the above Encoder-Decoder Network, an attention layer is added to understand the impact

```
Model: "model"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, 30)]         0
__________________________________________________________________________________________________
input_2 (InputLayer)            [(None, 100)]        0
__________________________________________________________________________________________________
embedding (Embedding)           (None, 30, 200)      1400000     input_1[0][0]
__________________________________________________________________________________________________
embedding_1 (Embedding)         (None, 100, 200)     1400000     input_2[0][0]
__________________________________________________________________________________________________
bidirectional (Bidirectional)   (None, 30, 100)      100400      embedding[0][0]
__________________________________________________________________________________________________
bidirectional_1 (Bidirectional) (None, 100, 100)     100400      embedding_1[0][0]
__________________________________________________________________________________________________
time_distributed (TimeDistribut (None, 30, 50)       5050        bidirectional[0][0]
__________________________________________________________________________________________________
time_distributed_1 (TimeDistrib (None, 100, 50)      5050        bidirectional_1[0][0]
__________________________________________________________________________________________________
attention (attention)           (None, 50)           80          time_distributed[0][0]
__________________________________________________________________________________________________
attention_1 (attention)         (None, 50)           150         time_distributed_1[0][0]
__________________________________________________________________________________________________
dropout (Dropout)               (None, 50)           0           attention[0][0]
__________________________________________________________________________________________________
dropout_1 (Dropout)             (None, 50)           0           attention_1[0][0]
__________________________________________________________________________________________________
ftrs_Input_layer (InputLayer)   [(None, 20)]         0
__________________________________________________________________________________________________
flatten (Flatten)               (None, 50)           0           dropout[0][0]
__________________________________________________________________________________________________
flatten_1 (Flatten)             (None, 50)           0           dropout_1[0][0]
__________________________________________________________________________________________________
ftrs_Dense (Dense)              (None, 100)          2100        ftrs_Input_layer[0][0]
__________________________________________________________________________________________________
concatenate (Concatenate)       (None, 200)          0           flatten[0][0]
                                                                 flatten_1[0][0]
                                                                 ftrs_Dense[0][0]
__________________________________________________________________________________________________
dense_2 (Dense)                 (None, 10)           2010        concatenate[0][0]
==================================================================================================
Total params: 3,015,240
Trainable params: 215,240
Non-trainable params: 2,800,000
```



With Attention Layer - Results





input_x	target_y	unique_target	training_acc	validation_acc	training_loss	validation_loss	glove_vector	attn_layer	no_of_epochs_run	batch_size	learning_rate	embb_trainable
short_long	y_75	10	0.93	0.73	0.2114	1.0061	yes	no	10	4	0.000040	FALSE
short_long	y_75	10	0.91	0.73	0.2865	0.9217	yes	no	10	8	0.000040	FALSE
short_long	**y_75**	**10**	**0.86**	**0.72**	**0.4324**	**0.8511**	**yes**	**no**	**9**	**16**	**0.000040**	**FALSE**
short_long	y_75	10	0.88	0.73	0.3790	0.8676	yes	no	14	32	0.000008	FALSE
short_long	y_75	10	0.88	0.72	0.3801	0.9255	yes	no	19	64	0.000008	FALSE
short_long	y_75	10	0.84	0.72	0.4918	0.8670	yes	no	18	100	0.000008	FALSE
short_long	y_75	10	0.98	0.74	0.0755	1.2255	yes	no	8	4	0.000040	TRUE
short_long	y_75	10	0.98	0.75	0.0783	1.2092	yes	no	9	8	0.000040	TRUE
short_long	**y_75**	**10**	**0.97**	**0.75**	**0.1034**	**1.1555**	**yes**	**no**	**9**	**16**	**0.000040**	**TRUE**
short_long	y_75	10	0.96	0.74	0.1395	1.1138	yes	no	12	32	0.000008	TRUE
short_long	y_75	10	0.95	0.74	0.1741	1.0683	yes	no	11	64	0.000040	TRUE
short_long	y_75	10	0.96	0.75	0.1702	1.0818	yes	no	13	100	0.000040	TRUE

Observations : The results are similar to the previous model with Encoder-Decoder network with additional features. We haven't observed significant change in any of the metrics after addition of attention layer. One of the reason could be small amount of dataset.

Try with Elmo Embedding

Elmo Embedding uses context-based word embedding and makes use of bidirectional LSTM to achieve this.

ELMo language model is trained on the 1B Word Benchmark. In addition, the language model really is large-scale with the LSTM layers containing 4096 units and the input embedding transform using 2048 convolutional filters Current Limitation

Elmo embedding was tried on the dataset, however during the build it was noticed that the TensorFlow Hub Modules for Elmo embeddings are currently not available for TensorFlow 2.0. The 1.x code can be invoked by turning off “eager execution” however this limits us from using the embedding as a layer in our current model.

Thus, it was agreed not to proceed with Elmo embedding as it would require us to convert the entire model to 1.x compatible

Final Model - Modified Encoder-Decoder Network

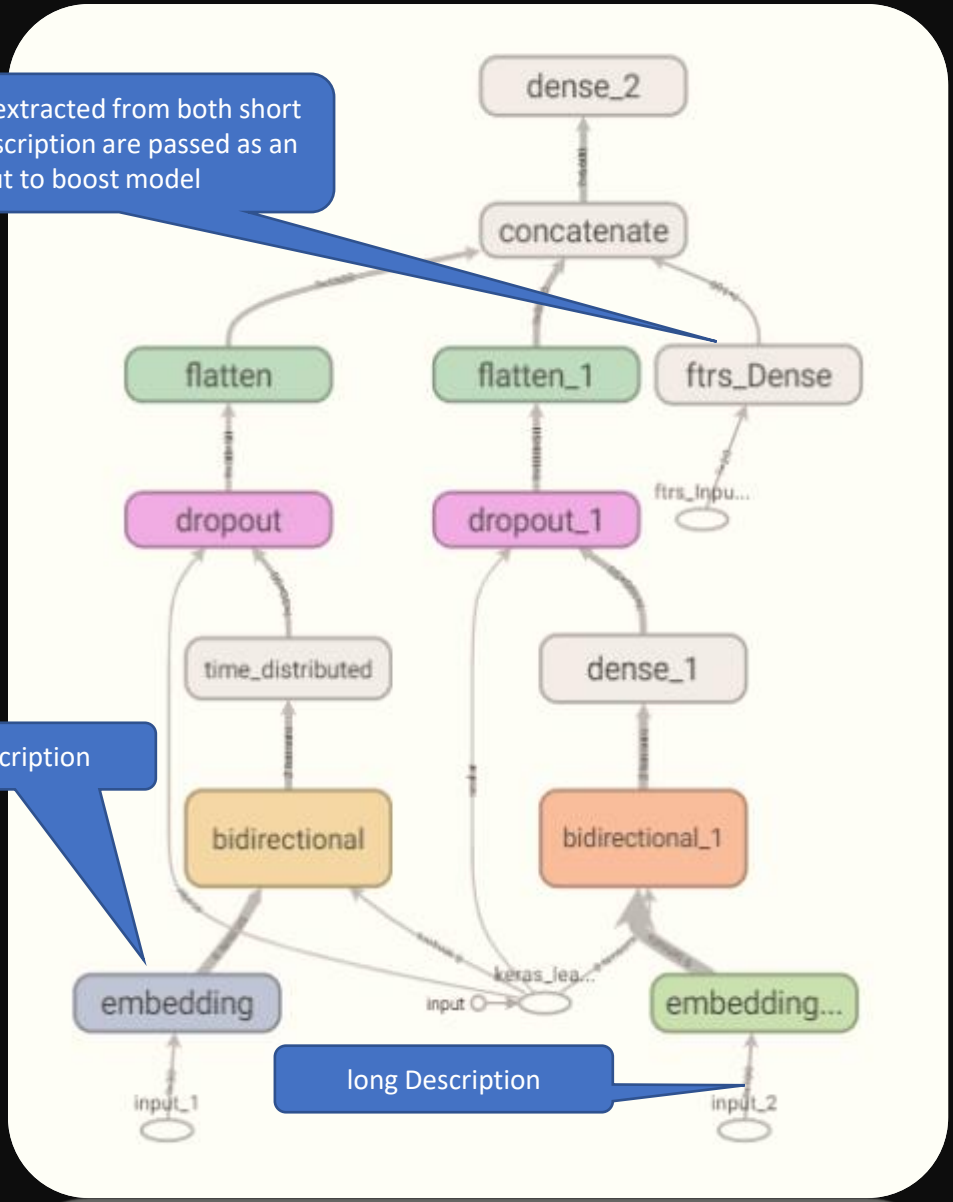
- Based on all the above results, final model selected is Modified Encoder-Decoder network which has 3 input layers
- Short Description, Long Description and Extracted features from the text

```
Model: "model_3"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_7 (InputLayer)            [(None, 30)]         0
__________________________________________________________________________________________________
input_8 (InputLayer)            [(None, 100)]        0
__________________________________________________________________________________________________
embedding_6 (Embedding)         (None, 30, 200)      1400000     input_7[0][0]
__________________________________________________________________________________________________
embedding_7 (Embedding)         (None, 100, 200)     1400000     input_8[0][0]
__________________________________________________________________________________________________
bidirectional_6 (Bidirectional) (None, 30, 100)      100400      embedding_6[0][0]
__________________________________________________________________________________________________
bidirectional_7 (Bidirectional) (None, 100, 100)     100400      embedding_7[0][0]
__________________________________________________________________________________________________
time_distributed_3 (TimeDistrib (None, 30, 50)       5050        bidirectional_6[0][0]
__________________________________________________________________________________________________
dense_10 (Dense)                (None, 100, 50)      5050        bidirectional_7[0][0]
__________________________________________________________________________________________________
dropout_6 (Dropout)             (None, 30, 50)       0           time_distributed_3[0][0]
__________________________________________________________________________________________________
dropout_7 (Dropout)             (None, 100, 50)      0           dense_10[0][0]
__________________________________________________________________________________________________
ftrs_Input_layer (InputLayer)   [(None, 20)]         0
__________________________________________________________________________________________________
flatten_6 (Flatten)             (None, 1500)         0           dropout_6[0][0]
__________________________________________________________________________________________________
flatten_7 (Flatten)             (None, 5000)         0           dropout_7[0][0]
__________________________________________________________________________________________________
ftrs_Dense (Dense)              (None, 100)          2100        ftrs_Input_layer[0][0]
__________________________________________________________________________________________________
concatenate_3 (Concatenate)     (None, 6600)         0           flatten_6[0][0]
                                                                 flatten_7[0][0]
                                                                 ftrs_Dense[0][0]
__________________________________________________________________________________________________
dense_11 (Dense)                (None, 10)           66010       concatenate_3[0][0]
==================================================================================================
Total params: 3,079,010
Trainable params: 279,010
Non-trainable params: 2,800,000
```

20 features extracted from both short and long description are passed as an input to boost model

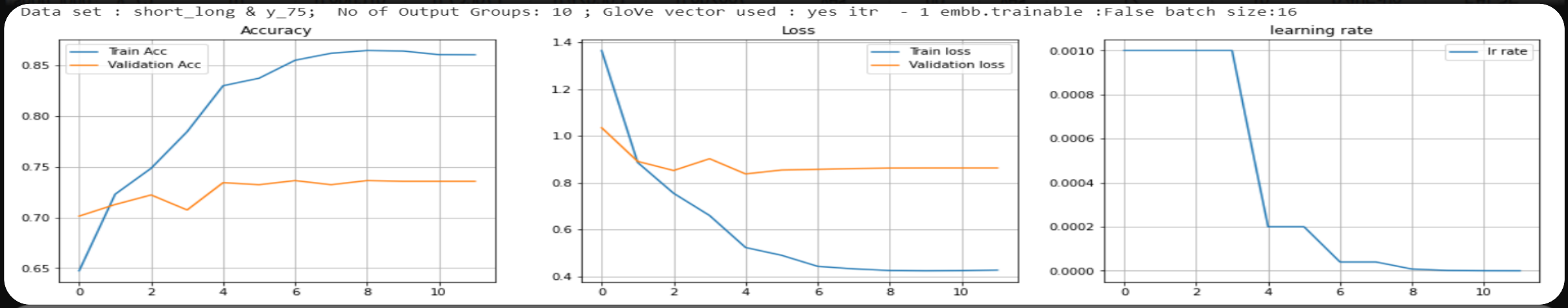
Short Description

long Description



Final Model - Results

input_x	target_y	unique_target	training_acc	validation_acc	training_loss	validation_loss	glove_vector	attn_layer	conv1d	no_of_epochs_run	batch_size	learning_rate	embb_trainable
short_long	y_75	10	0.860516	0.735671	0.426565	0.862887	yes	no	yes	12	16	6.40E-08	FALSE



Confusion Matrix

	GRP_0	GRP_12	GRP_19	GRP_2	GRP_24	GRP_3	GRP_6	GRP_8	GRP_9	GRP_9999
GRP_0	677	2	5	6	5	3	0	0	0	68
GRP_12	8	28	0	1	0	0	0	2	0	18
GRP_19	27	0	6	0	0	4	0	0	0	7
GRP_2	15	3	0	21	0	2	0	0	0	8
GRP_24	13	2	0	0	42	1	0	1	0	2
GRP_3	10	0	3	0	2	12	0	0	0	8
GRP_6	0	0	0	0	0	0	3	0	0	8
GRP_8	2	0	0	0	0	0	0	41	0	6
GRP_9	3	1	0	0	0	0	0	0	5	4
GRP_9999	103	12	3	7	3	3	5	5	1	256

Classification Report

	precision	recall	f1-score	support
GRP_0	0.79	0.88	0.83	766
GRP_12	0.58	0.49	0.53	57
GRP_19	0.35	0.14	0.20	44
GRP_2	0.60	0.43	0.50	49
GRP_24	0.81	0.69	0.74	61
GRP_3	0.48	0.34	0.40	35
GRP_6	0.38	0.27	0.32	11
GRP_8	0.84	0.84	0.84	49
GRP_9	0.83	0.38	0.53	13
GRP_9999	0.66	0.64	0.65	398
accuracy			0.74	1483
macro avg	0.63	0.51	0.55	1483
ighted avg	0.72	0.74	0.72	1483

Final Model – Examples of correct predictions

		Actual	Predicted	
short_desc	long_desc	assignment_grp_	assignment_grp_pre	y
locked	im trying add change item er get error listed knethyen grechduy er	GRP_0	GRP_0	TRUE
engineering tool kpm project numbers right order	hello engineering tool kpm project numbers numerirtcal right orde	GRP_9999	GRP_9999	TRUE
browser issue flash player addins issue	browser issue flash player addins issue	GRP_0	GRP_0	TRUE
employee owned mobility agreement	hello help team attached find agreement need next get email push	GRP_0	GRP_0	TRUE
hostname disk free c warning threshold total size gb	hostname average samples disk free c warning threshold total size	GRP_12	GRP_12	TRUE
engineeringtool working	hello please support dealer engineeringtool installation contact de	GRP_0	GRP_0	TRUE
user issues login pc new password	user issues login pc new password connected user system using te	GRP_0	GRP_0	TRUE
ticket update inplant	ticket update inplant	GRP_0	GRP_0	TRUE
personal certificate error skype issue	personal certificate error skype issue	GRP_0	GRP_0	TRUE
security incidents in possible malware infection traffic sinkh	source ip system name androidfaecee user location unknown sep	GRP_2	GRP_2	TRUE
kndigung action completed	hello kndigung effective approved	GRP_2	GRP_2	TRUE
hostname average samples disk free c warning threshold	hostname average samples disk free c warning threshold total size	GRP_12	GRP_12	TRUE
erp sid account locked	erp sid account locked	GRP_0	GRP_0	TRUE
password reset request	password reset request	GRP_0	GRP_0	TRUE
windows account unlock	windows account unlock	GRP_0	GRP_0	TRUE
query owner shared mailbox	query owner shared mailbox	GRP_0	GRP_0	TRUE
order products online problem	hello ask help cant solve help tomorrow	GRP_0	GRP_0	TRUE
ticket ticketno customer stating label	please ticket ticketno review label customer barcode output print	GRP_9999	GRP_9999	TRUE
password expires	hello password expires couldnt change	GRP_0	GRP_0	TRUE
calls coming rerouting showing missed call	calls coming rerouting showing missed call telephonysoftware ext	GRP_9999	GRP_9999	TRUE
update crm access ticketno	update crm access ticketno	GRP_0	GRP_0	TRUE
software installation	software installation	GRP_0	GRP_0	TRUE
wrong nxd plm	hello drawing saved wrong plm number please delete nxd mit freu	GRP_9999	GRP_9999	TRUE
hpqc password	hello failed login hpqc account got message could please reset pa	GRP_0	GRP_0	TRUE

Final Model – Examples of incorrect predictions

		Actual	Predicted	
short_desc	long_desc	assignment_grp_	assignment_grp_pre	y
connectivity issue company carrier location	best	GRP_9999	GRP_0	FALSE
hello problems running open sales backorder report data dc	hello problems running open sales backorder report data downloa	GRP_9999	GRP_12	FALSE
folder deletion drive	hello could please help retrieving deleted folder drive link folder n	GRP_12	GRP_0	FALSE
security incidents in possible daserf trojan apt activity leeng	source ip source hostname leengineeringtoolzhm destination ip sy	GRP_9999	GRP_2	FALSE
hrtool etime run update ran last night asks adobe flash tried	immediate need	GRP_3	GRP_0	FALSE
launch collaborationplatform internet browser ie launch otl	launch collaborationplatform internet browser ie launch others ar	GRP_3	GRP_0	FALSE
analysis office aao tool	install analysis office aao tool laptop	GRP_19	GRP_0	FALSE
dock station telephonysoftware ip phone	request install dock station ip phone telephonysoftware applicatic	GRP_19	GRP_3	FALSE
cannot print erp documents zzmails functions since today ar	want print erp documents pdf file zzmails function tried since noo	GRP_0	GRP_9999	FALSE
problem outlook client database synchronizing	problem outlook client database synchronizing	GRP_19	GRP_0	FALSE
pos data september	please provide pos data month september	GRP_9	GRP_9999	FALSE
outlook freezing frequently	user called issue outlook freezing frequently	GRP_3	GRP_0	FALSE
coffee spillage keybankrd	coffee spillage keybankrd	GRP_3	GRP_0	FALSE
hostname volume f labeldathostname ad space consumed s	hostname volume f labeldathostname ad space consumed space a	GRP_12	GRP_9999	FALSE
error drucker	received nothing printed error message cid best friendly greet	GRP_9999	GRP_0	FALSE
please release access	received cid best friendly greet	GRP_9999	GRP_12	FALSE
especially email working	server ip reply ping smtp server error communication error contac	GRP_12	GRP_0	FALSE
please give access sid uacyltoe hxgaycze system user pildlad	please give access sid uacyltoe hxgaycze system user pildladjadga	GRP_0	GRP_2	FALSE
monitor working	monitor working	GRP_3	GRP_19	FALSE
wrong calculation erp	rebate calculated correctly inwarehousetools little rebate	GRP_9999	GRP_0	FALSE
pdf printer	want software print documents pdf	GRP_19	GRP_0	FALSE
engineering tool issue	dear sir received new system last week engineering tool downloa	GRP_19	GRP_0	FALSE
unable create print dn cant post detail information see atta	please provide following order number material item number mm	GRP_6	GRP_9999	FALSE
security incidents dsw in traffic sinkhole domain lpawxsf	source ip system name lpawxsf user name na location sep sms sta	GRP_9999	GRP_2	FALSE

Challenges faced in improving the accuracy

Even after trying complex models, it was observed that the model is finding it hard to improve the accuracy beyond 72%. While there is an improvement in the overall accuracy from simple LSTM to BiLSTM, CNN+BiLSTM and Encoder-Decoder network, it was observed that beyond 70% accuracy model is struggling to improve without overfitting. Following attempts were made to improve the model accuracy

- Tried various combinations of **input length** for both Short and Long description
- Tried **combination of both Sort and Long description**
- Increased **num_words** during tokenization to maximum
- Added the **“caller” column** to the model to see if the user info improve accuracy
- Added **additional features extracted** from the data as a separate input layer to improve the accuracy
- Controlled overfitting through
  - Adding **dropout, recurrent dropout** in BiLSTM layers
  - **Dropout** after dense layers
  - Addition of **glove vector** in the embeddings and making the embedding layer **non-trainable**

Even after trying all these combinations and running multiple simulations, we could not improve the validation accuracy beyond a maximum of 75%. After thorough investigation we found that the accuracy can only be improved by providing rich and more data. This is explained in detail in the next slide

Challenges faced in improving the accuracy

- **Data sparsity :** Data is extremely sparce where **55% of the data** (L1/L2 tickets) belongs to 2 categories and **10% data is spread across 53 categories** and 35% of the data belongs to 19 categories.
- **Small Dataset** : For the number of categories in question(74), the data set of 8500 records is **very small**. Deep learning algorithms are data hungry especially when the no of target categories are high.
- **Less no of features in the dataset** : There are very less no of features in the dataset to make accurate predictions. For example, we are not sure even the classification of the category like L1/L2/L3. Additional features like below will can improve the accuracy dramatically
  - Classification of the categories into smaller buckets like L1/L2/L3 with few groups within
  - Additional features of the tickets from the ticketing systems like ticket class, ticket priority..etc....
  - It is not very clear if the ticket is resolved.
  - Reception mode of the ticket like Email, Self generated tickets by agents, users entering the tickets directly into system ..etc....

All the ticketing systems today have ability to capture all these features and many more and they help a lot to enrich the data set and there by improve our model performance.

Conclusion :

Here is the summary of results compared to the business case. If we recall the business case, our objective is

- To assign the tickets to the right group automatically so that the tickets are addressed on time
- L1/L2 teams do not need to spend time on reviewing SOP before assigning a ticket
- Reducing the errors in wrong assignments

Below is a summary of our results. Even with 73% accuracy we achieved with our model, there is significant amount of time saving (178 Hrs.) considering the 8500 tickets given in the dataset.

Total No of Tickets	8500	
No tickets resolved by L1/L2(54%)	4590	
No tickets resolved by L3(56%)	3910	
No of Tickets reviewed for SOP (25%) before assignment	977	
Time taken to review for each ticket(mins)	15	
Total time taken (mins)	14663	244 Hrs.
Auto assignment (AI/ML model) accuracy	73%	
Time saved	178 Hrs.	

Learnings from the project

Data Preparation

This is the most important step in the whole process. We observed that it is impossible to construct a correct model without having a good understanding of the data.

Data Exploration and Profiling

It is important to assess the condition of the data, including looking for trends, outliers, exceptions, incorrect, inconsistent, missing, or skewed information. This is important because the source data will inform all of our model's findings, so it is critical to be sure it does not contain unseen biases. This is the time to catch any issues that could incorrectly skew your model's findings, on the entire data set, and not just on partial or sample data sets.

Formatting data to make it consistent

Ensure your data is formatted in a way that best fits your machine learning model. In our dataset we found that multiple languages are used so we had to convert them to English to make the data consistent.

Improving data quality

It is important to remove unnecessary information from the data. In our case we found lot of junk characters, email id's, html tags, usernames ..etc.... within the text which need to be removed to improve the data quality

Feature engineering

This step involves the art and science of transforming raw data into features that better represent a pattern to the learning algorithms. We extracted **20 more columns** from this small dataset to make sense of the data. This is quite important when we make production grade systems. This not only helps in improving the model accuracy but also helps a lot in validating the model performance in production.

Conclusion - Learnings

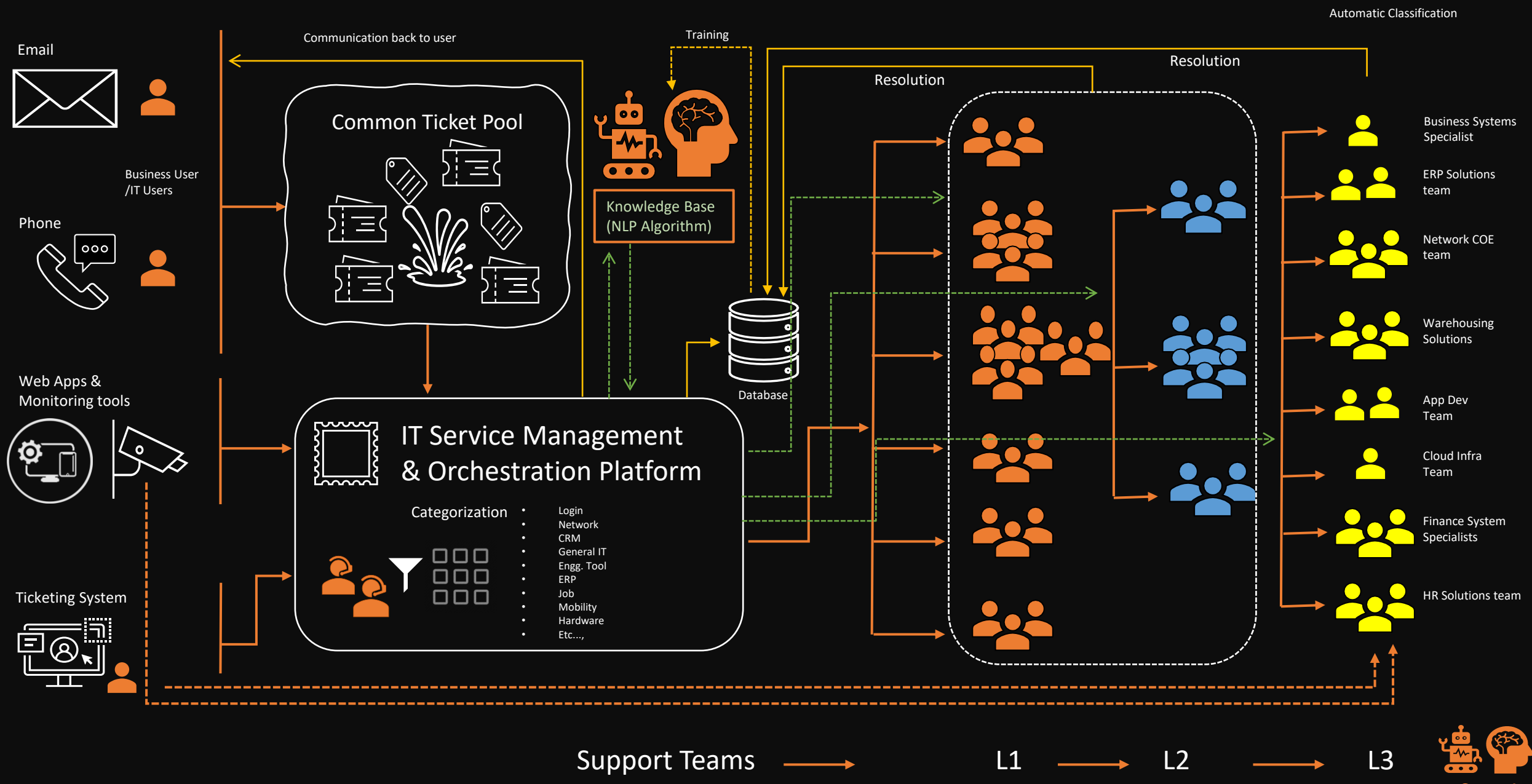
Model Building

Model Architecture: Start with rather a simple architecture to assess the base accuracy to set a benchmark for further improvement. There is no simple or easy way to find the best fit architecture. We have to decide architecture based on the problem and it involves lot of trial and error.

Controlling overfitting: While building the model it is important to control overfitting of the model. High training accuracies looks god but fails in production if not regularized properly.

Test model in production settings, get more insights about what could go wrong and then continue improving our model with continuous integration.

IT Incident Management Framework with AI solution - Proposed



Group-4

Closing notes

Project team(Group-4) want to thank greatlearning’s team for all the help and guidance during the entire length of our course and during this project

This project was not an easy task for the team and as a result there have been some hardships and roadblocks. However, through perseverance and determination we were able to push through them to get to the point where it is now. Though there is a chance to improve the results further, we believe that we were able to get it to a point where it shows that great progress was made. We are truly satisfied with how much we were able to learn and achieve throughout this project.

We wanted to thank Shyam for simply being our mentor throughout our journey and particularly during this project. Shyam was there to mentor and guide us throughout this project and ensure that we had a great learning experience.

Sources/References

How to Develop a Bidirectional LSTM For Sequence Classification in Python with Keras – Machine Learning Mastery

CNN Long Short-Term Memory Networks - Machine Learning Mastery

CONVOLUTIONAL, LONG SHORT-TERM MEMORY, FULLY CONNECTED DEEP NEURAL NETWORKS - Tara N. Sainath, Oriol Vinyals, Andrew Senior, Has¸im Sak Google, Inc., New York, NY, USA {tsainath, vinyals, andrewsenior, hasim}@google.com

Attention Mechanism -

Encoder-Decoder Long Short-Term Memory Networks – Machine Learning Mastery

Understanding Encoder-Decoder Sequence to Sequence Model - Simeon Kostadinov

A ten-minute introduction to sequence-to-sequence learning in Keras – The Keras Blog

https://blog.floydhub.com/attention-mechanism/

https://matthewmcateer.me/blog/getting-started-with-attention-for-classification/

https://opensource.com/article/19/7/get-modular-python-functions

https://www.python-course.eu/python3_modules_and_modular_programming.php#:~:text=Designing%20and%20Writing%20Modules&text=A%20module%20in%20Python%20is,the%20module%20name%20is%20fibonacci.

https://docs.python-guide.org/writing/structure/


Thank
you