

CM3203: One semester Individual project

USING SENTIMENT ANALYSIS TO IMPROVE EMOJI PREDICTIONS IN TWEETS

BY FANI NONCHEVA

Presentation summary

- ▶ Brief introduction
- ▶ Background
- ▶ Approach
- ▶ Data Processing
- ▶ Models and implementation
- ▶ Findings
- ▶ Conclusion
- ▶ Questions panel

Brief introduction

- ▶ The project focuses on the use of deep learning, AKA neural networks to complete its goals.
- ▶ I chose this project influenced by the SemEval task of 2018 that employs different neural network architectures to predict emoji.
- ▶ The scope of this project lies mostly in the realm of natural language processing and focuses on the analysis of twitter data.
- ▶ The task consists of three main points
 - ▶ Creating a base emoji prediction model
 - ▶ Creating a sentiment analysis model
 - ▶ Combining the two and benchmarking to the base model.

Background

- ▶ Studies suggest that for the purposes of natural language processing (NLP), long short term memory (LSTM) neural networks prove to be a good option. Hence why I've chosen them as my primary focus in the project.
- ▶ Bidirectional LSTMs, which essentially encapsulate two LSTMs that analyse the sequential data in both directions (forward and backward), outperform the normal variant of LSTM, as they provide more context.
- ▶ Embeddings aid greatly in determining the correlation between the individual word tokens in given contexts when using LSTM layers.
- ▶ Adding attention (a bias vector that aids with predictions), aids the process significantly, providing more emphasis on specific areas.

Data

- ▶ The data for emoji prediction is the dataset from the study that inspired this project, with annotation for the twenty most popular emojis in winter 2017-2018 in the US.
- ▶ The sentiment analysis data is publicly sourced and available under Stanford's Sentiment 140 project. Provided with not only a training set, but a test one as well.

Approach

- ▶ I have chosen to work with TensorFlow v2.0, specifically the Keras API for this project.
- ▶ The Keras API was chosen, as I was new to data science. Hence, I chose the most user-friendly option available at the start.
- ▶ The data for the two models is processed differently, due to their specific characteristics.
 - ▶ Emoji data is processed prior to running the model
 - ▶ Sentiment analysis is processed during the building of the model.
- ▶ That choice was made as one the emoji model was more complex and required more tweaking and resetting, hence I chose to optimize the time in which it takes to run the application every time.

Data processing

- ▶ Data processing is essentially clearing away the noise from the given data.
- ▶ To accomplish this I employ the following functions:
 - ▶ Lowercasing
 - ▶ Removing extra spaces
 - ▶ Stripping away punctuation and special characters
 - ▶ Removing numerical characters
- ▶ When it comes to tweets more often than not, punctuation, special characters and numerical characters are often used aesthetically, hence not providing crucial information for analysis.

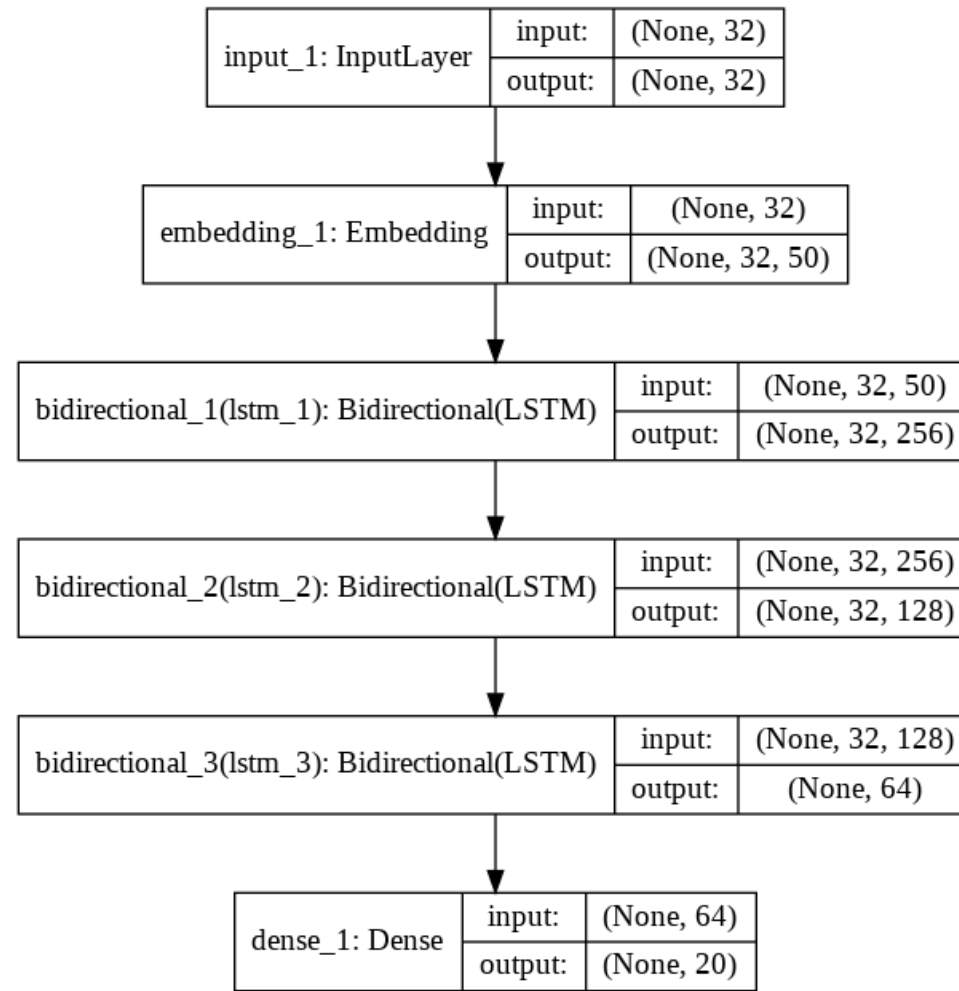
Models and implementation

- ▶ The models are LSTM-based with varying architectures and technical choices depending on their respective use.
- ▶ The first two presented are simple models that complete the first two tasks of basic sentiment analysis and emoji prediction.
- ▶ The remainder of the models are more complex, as they are attempts to converge the two individual models and determine how they would influence each other.

Embeddings

- ▶ Embeddings aid the machine learning model to evaluate how close given tokens are in 'meaning'.
- ▶ I employ Glove embeddings for this project, since training fresh embedding layers gave much more arbitrary results due to the insufficient time and data to surmount that barrier.
- ▶ I use the twenty-seven billion token twitter embeddings provided on the Glove webpage, and referenced in the report.

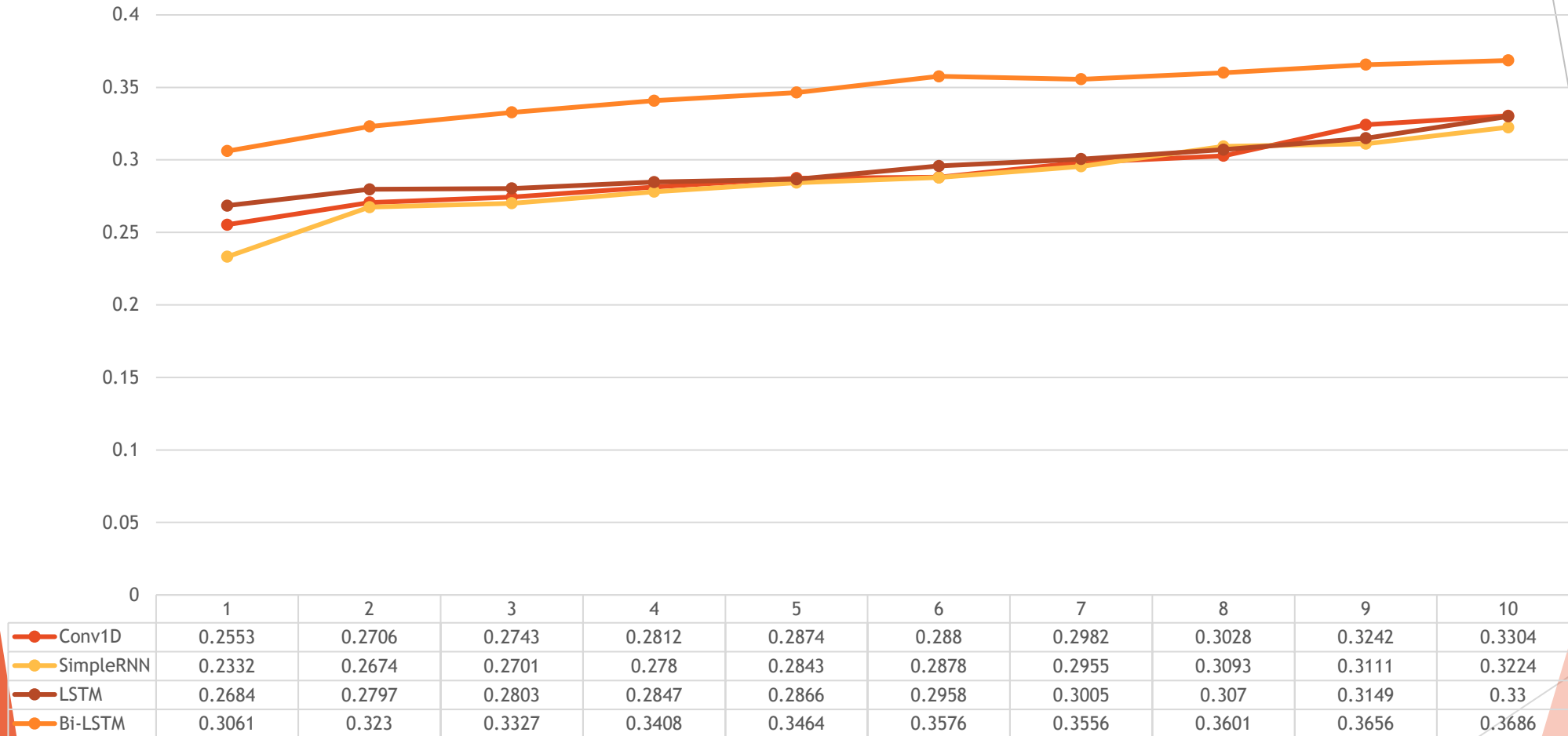
Baseline Emoji prediction



- ▶ This is the baseline I chose to use as benchmark for the final product.
- ▶ Its architecture is reminiscent of the combined structures, by implementing three bidirectional LSTM layers and an embedding layer at the start.
- ▶ It employs the use of categorical cross entropy to calculate loss and accuracy.
- ▶ The optimizer chosen for it is RMSprop as it handles greater sizes of dictionaries well
- ▶ Following up, graphs show it excels in comparison to the Convolutional, RNN and simple LSTM variants of the baseline model.

Characteristics

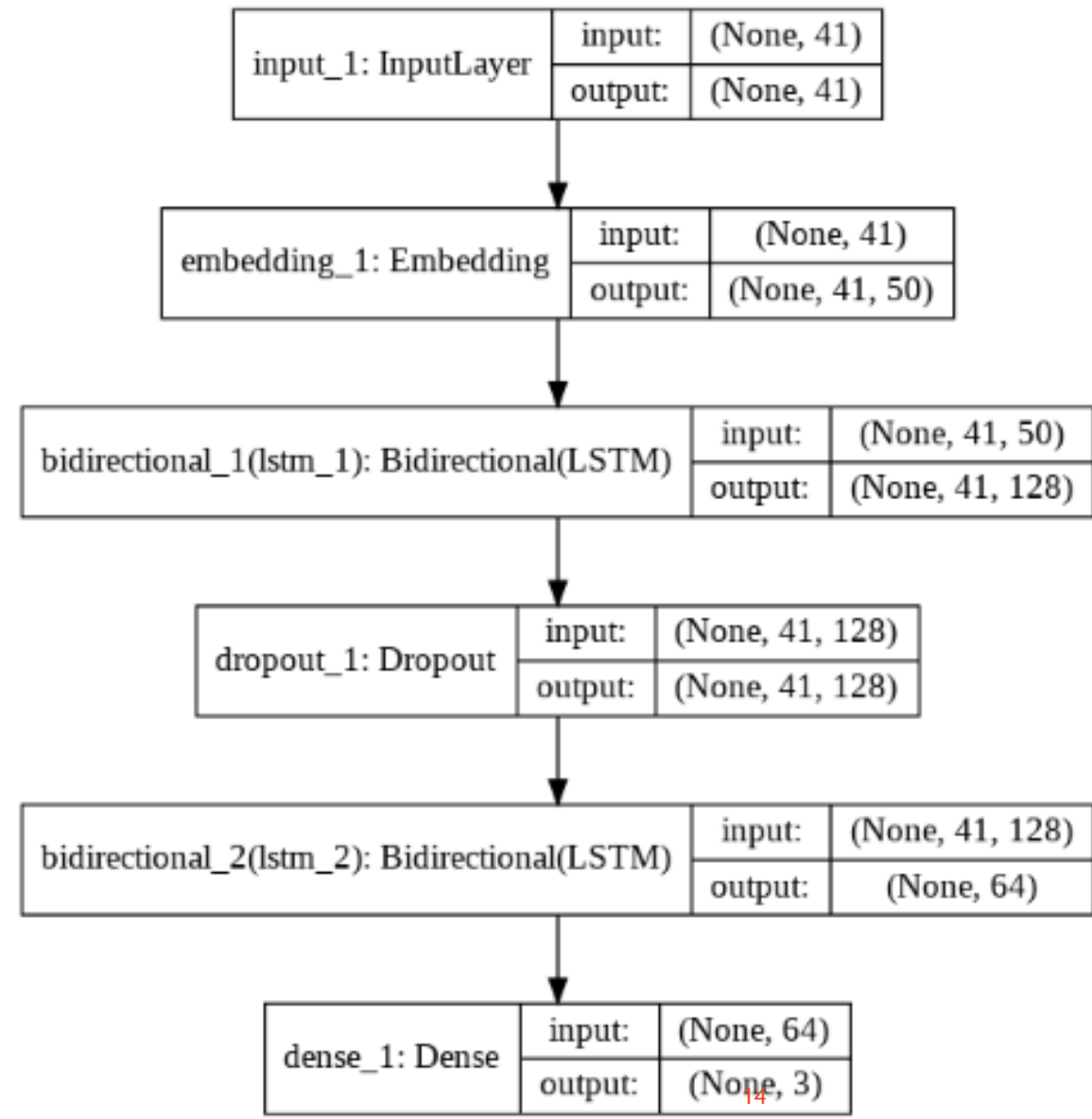
Emoji Prediction Accuracy Architecture Comparison (training)



Emoji Prediction Loss comparison (training)



Sentiment analysis

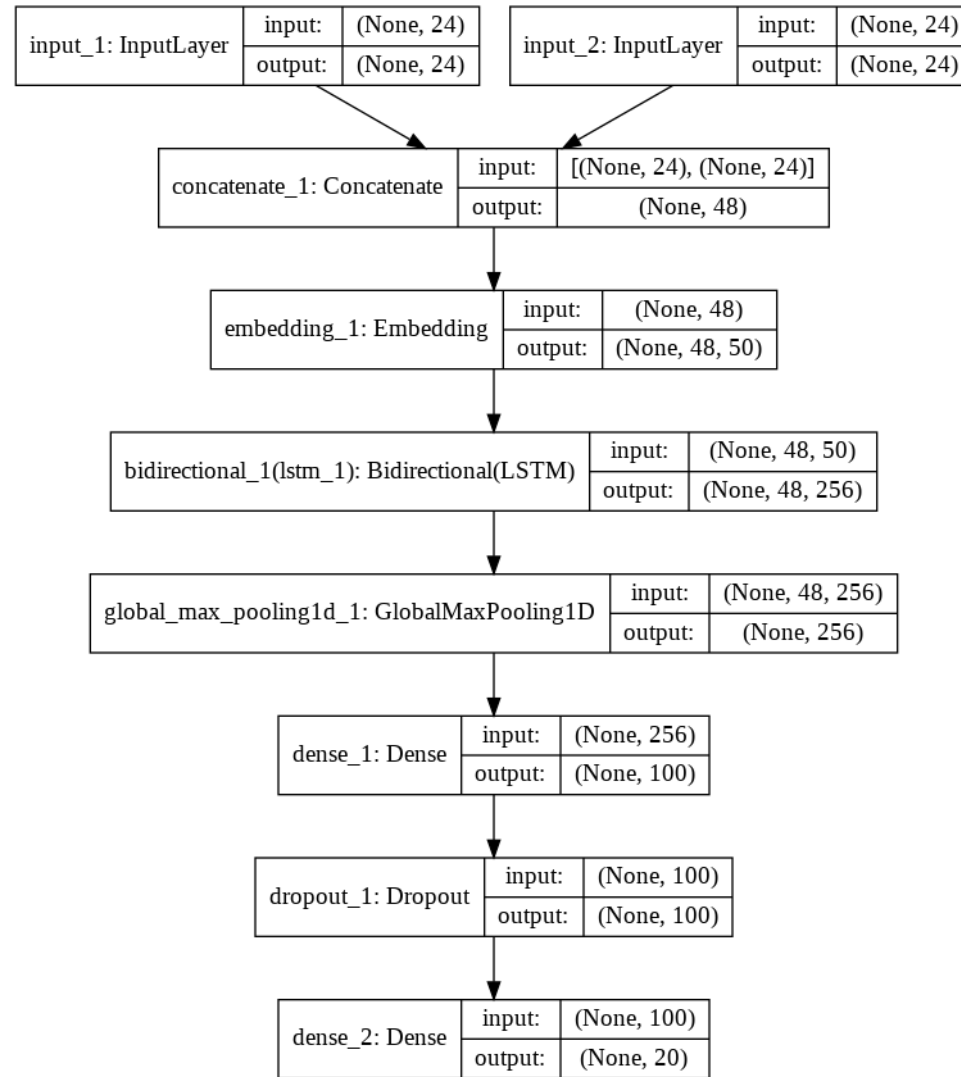


- ▶ This model was derived from its predecessor - an ordinary LSTM model, it outperforms significantly in regards to accuracy and loss.
- ▶ The initial state of this model was a binary classification, between positive and negative sentiment.
- ▶ The final version is more nuanced, with the improvement of teaching the model to discern neutral tweets as well.

Characteristics

Combined models

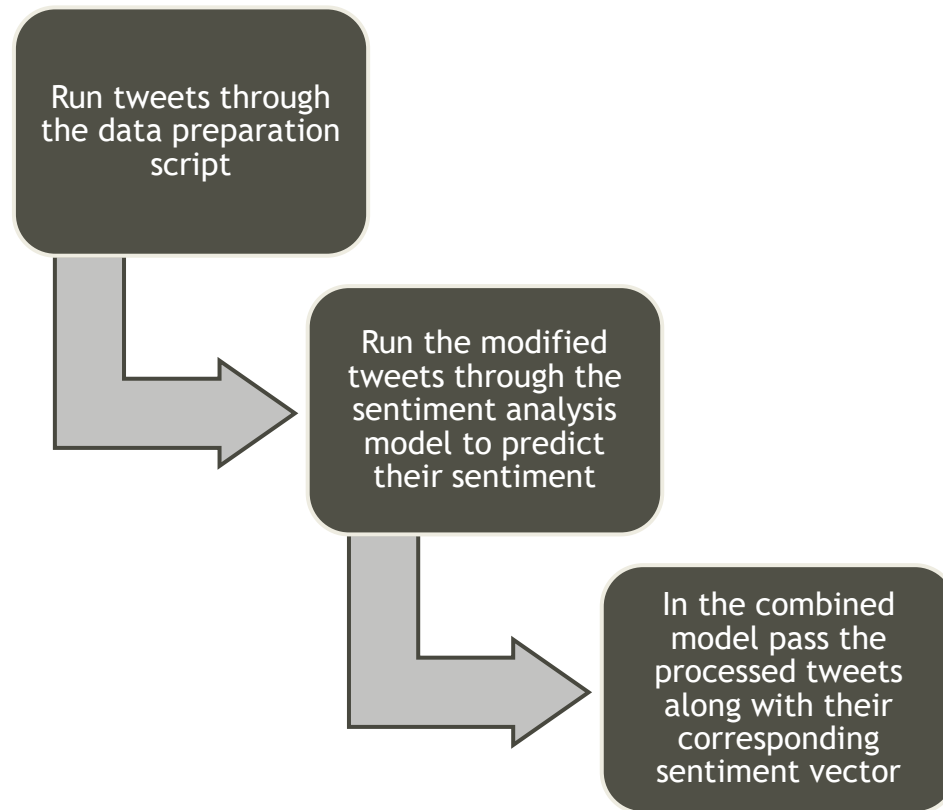
Pipeline Emoji prediction with sentiment analysis vector



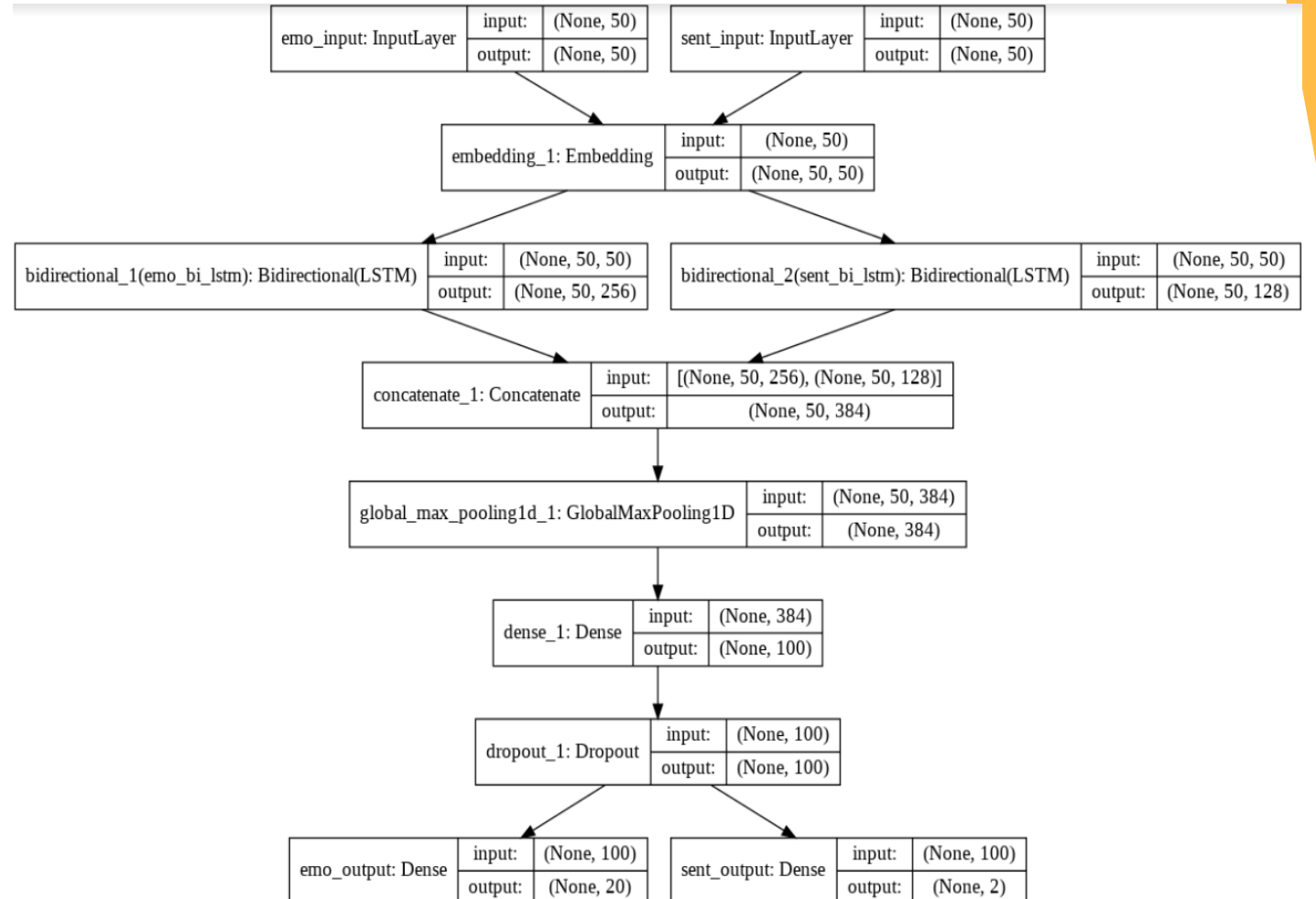
- ▶ This model builds up on the original model by adding an additional input for a vector containing the sentiment of the tweet.
- ▶ Alongside this, the architecture changes by employing a max pooling layer that extracts the Bidirectional LSTM features more adequately.
- ▶ In turn, that provides a more efficient architecture in comparison to the baseline. It proves to be a computationally more optimal solution, as well.
- ▶ The model uses a pipeline to execute its predictions and training, which is showcased in the next slide.

Characteristics

Pipeline overview



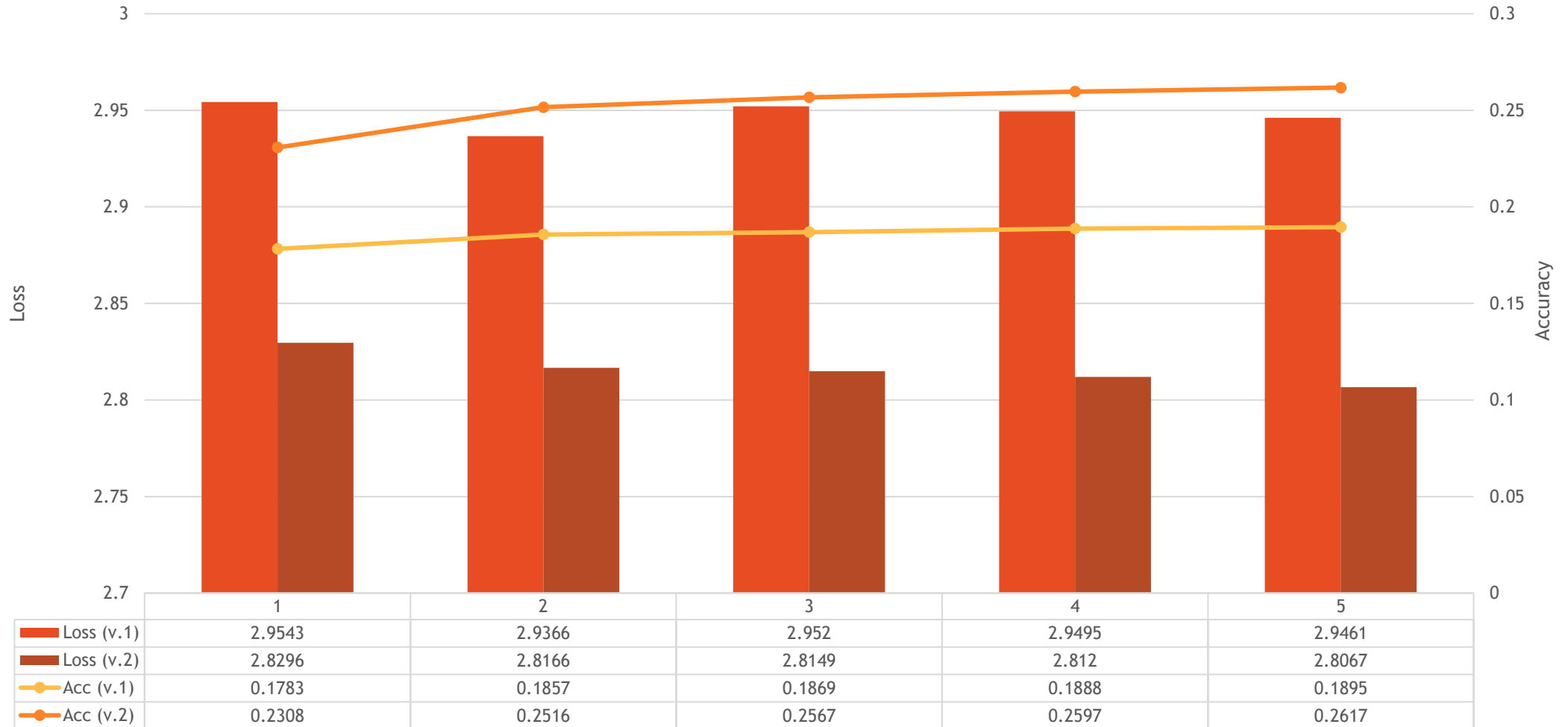
Merged sentiment and emoji prediction model



- ▶ The merged model went through several versions. Only the latest one will be discussed.
- ▶ Similarly to the pipeline combined model it offers a pooling layer. The difference is that this time it is utilized after a concatenation, allowing for a more optimized approach compared to using more LSTMs.

Characteristics

Emoji prediction metrics of merged models



Challenges

- ▶ Steep learning curve in complexity when it comes to machine learning
- ▶ Learning to time-manage in a larger-scale project
 - ▶ Managing time expenditures on individual tasks
 - ▶ Setting realistic estimates.
- ▶ Getting to know the problem specifics and how best to apply the research knowledge to it

Findings

The background of the slide features a series of overlapping, semi-transparent triangles in shades of orange and yellow. These triangles are arranged in a way that creates a sense of depth and movement, with some triangles pointing towards the top right and others towards the bottom left. Thin, dark lines intersect these triangles, adding to the geometric complexity of the design. The overall color palette is warm and vibrant, with the orange and yellow tones dominating the visual space.

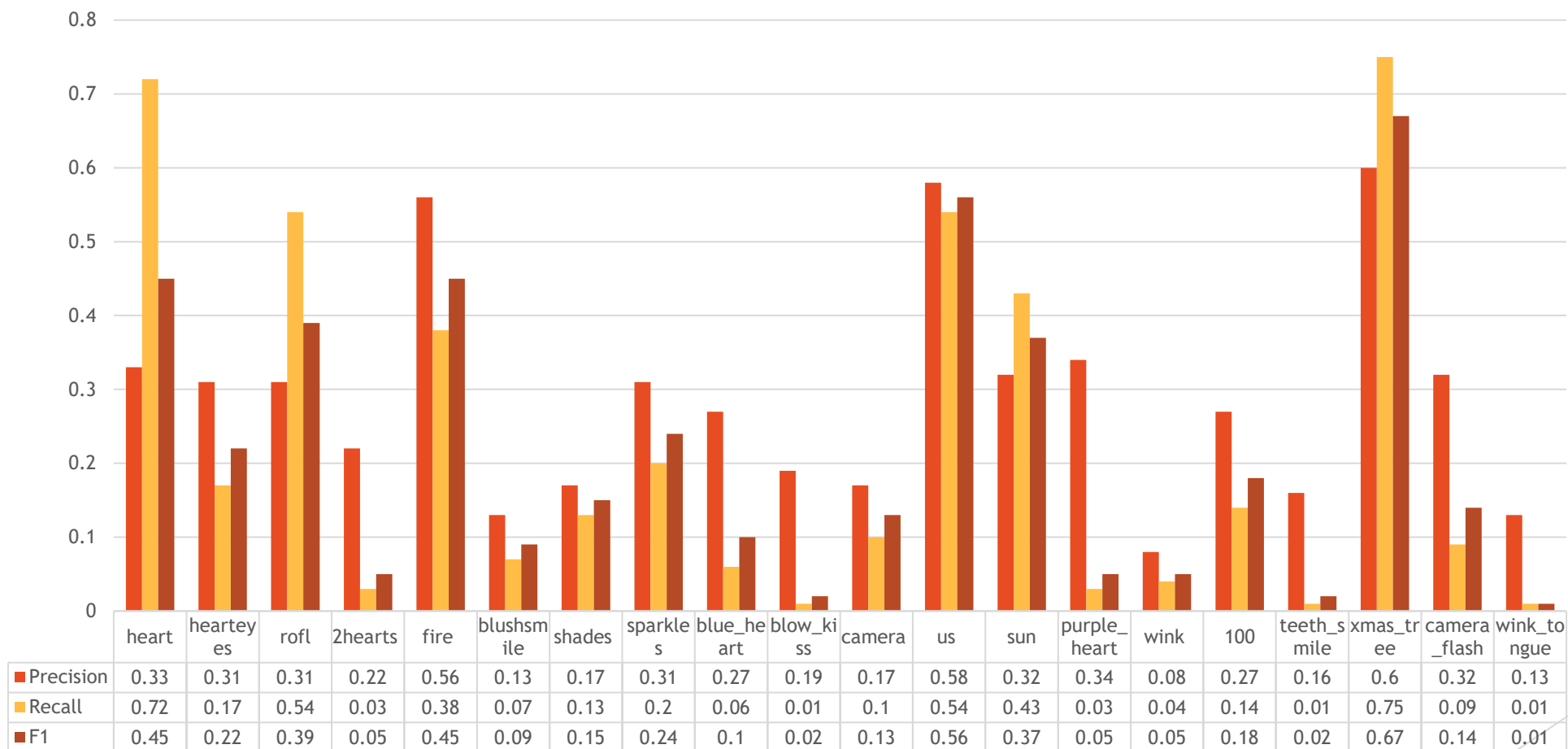
Overall findings

- ▶ The models with two inputs have varying degrees of success.
 - ▶ The model utilizing a sentiment vector outperforms the baseline by a negligible 1% in accuracy.
 - ▶ The model that takes two inputs of sentiment and emoji and adjusts weights for both does not outperform the baseline due to its complexity.
- ▶ The models perform better with medium amounts of noise reduction from the data: overly optimizing the data throws off the LSTM layers and they are unable to adequately assess the sequences.

Macros comparison of baseline and sentiment vector model

- ▶ The following slides give an individual overview for each of the emojis.
- ▶ The statistics cover precision, recall, F1 and accuracy for each individual emoji used in the prediction process.
- ▶ Lastly, there is a comparison table of the results of the emojis overall, as well as a comparison of their macros and parallels to the competition in emoji prediction this project is based on.

Emoji-only in-depth diagram



Emoji plus sentiment in-depth diagram

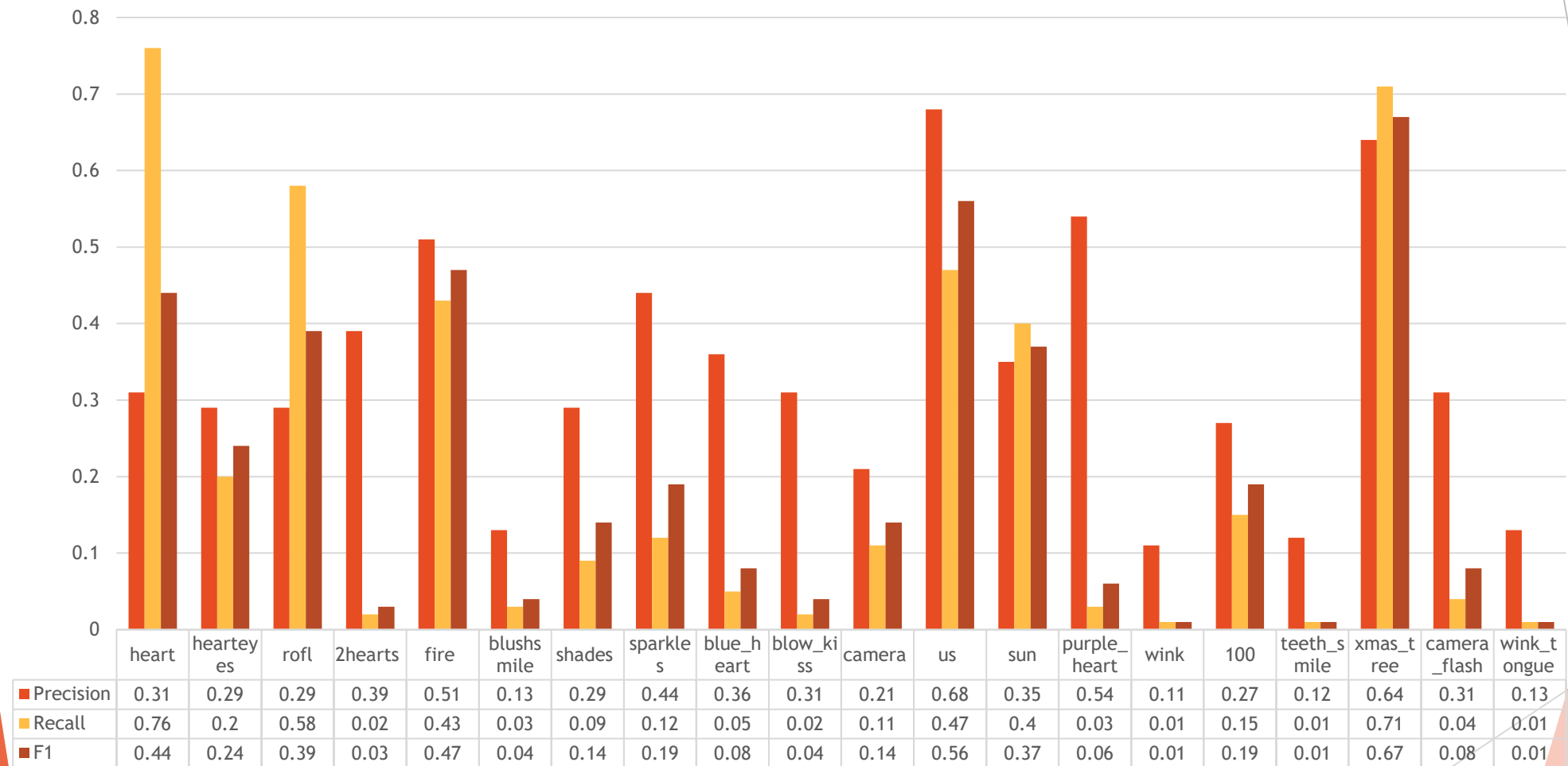
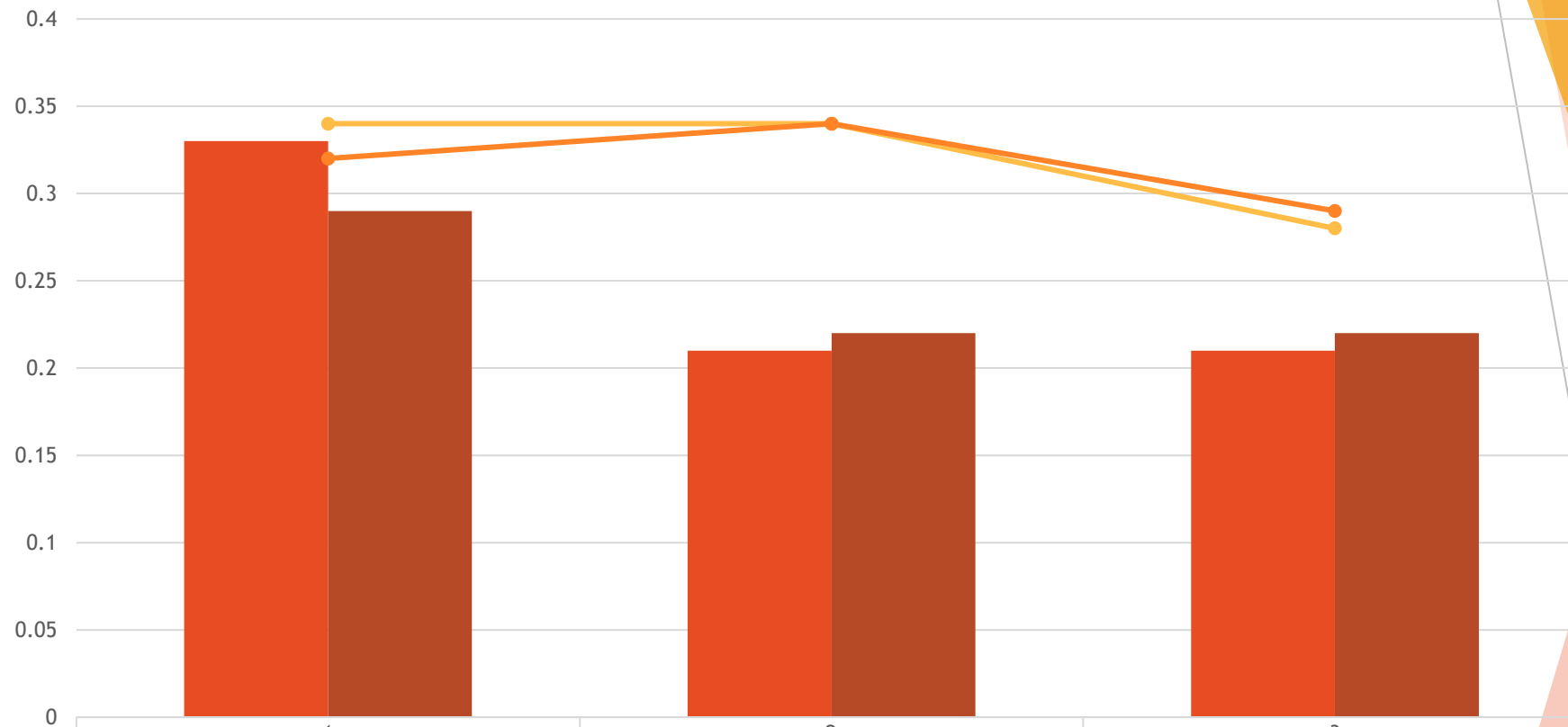


Table side-by-side comparison

Emoji	Precision(base)	Recall(base)	F1(base)	Samples	Precision (sentemo)	Recall (sentemo)	F1(sent emo)
heart	0.33	0.72	0.45	10798	0.31	0.76	0.44
Heart_eyes	0.31	0.17	0.22	4830	0.29	0.2	0.24
rofl	0.31	0.54	0.39	4534	0.29	0.58	0.39
2_hearts	0.22	0.03	0.05	2605	0.39	0.02	0.03
fire	0.56	0.38	0.45	3716	0.51	0.43	0.47
Blush_smile	0.13	0.07	0.09	1613	0.13	0.03	0.04
shades	0.17	0.13	0.15	1996	0.29	0.09	0.14
sparkles	0.31	0.2	0.24	2749	0.44	0.12	0.19
blue_heart	0.27	0.06	0.1	1549	0.36	0.05	0.08
blow_kiss	0.19	0.01	0.02	1175	0.31	0.02	0.04
camera	0.17	0.1	0.13	1432	0.21	0.11	0.14
us	0.58	0.54	0.56	1949	0.68	0.47	0.56
sun	0.32	0.43	0.37	1265	0.35	0.4	0.37
purple_heart	0.34	0.03	0.05	1114	0.54	0.03	0.06
wink	0.08	0.04	0.05	1306	0.11	0.01	0.01
100_emoji	0.27	0.14	0.18	1244	0.27	0.15	0.19
teeth_smile	0.16	0.01	0.02	1153	0.12	0.01	0.01
xmas_tree	0.6	0.75	0.67	1545	0.64	0.71	0.67
camera_flash	0.32	0.09	0.14	2417	0.31	0.04	0.08
wink_tongue	0.13	0.01	0.01	1010	0.13	0.01	0.01
macro avg	0.29	0.22	0.22	50000	0.33	0.21	0.21
weighted avg	0.32	0.34	0.29	50000	0.34	0.34	0.28

Weighted averages and macro average comparison




	1	2	3
Macro avg (emoji+sent)	0.33	0.21	0.21
Macro avg (emoji only)	0.29	0.22	0.22
Weighted avg (emoji+sent)	0.34	0.34	0.28
Weighted avg (emoji only)	0.32	0.34	0.29

Comparison to models presented in SemEval 2018 task 2

Team/Model	F1	Precision	Recall	Accuracy
Baseline(SE2018)	30.98	30.34	33	42.56
Tubingen-Oslo	35.99	36.55	36.22	47.09
EmoNLP	33.67	39.43	33.74	47.46
NTUA-SLP	35.36	34.53	38.00	44.74
UMDuluth-CS8761	31.83	39.80	31.37	45.73
Baseline(current)	29	32	34	33
SentEmo(current)	29	34	34	34

- ▶ While the idea behind the project has merit, further investigation and more advanced approaches must be used to improve the results that yield from this experiment.
- ▶ Future work on this idea would be exploring it by using convolutional neural networks and more complex mechanisms like attention for another LSTM variation.

Conclusion



Feel free to ask
any questions
as of now.

Thank you for
your time. 😊