Machine Learning for NLP, Coursework 2

Craciun, Mihai
24092840

Ibrahim, Omar
24029408

Nichols, Benjamin
24064164

Sewell, Anna
1989283

1 Introduction

As technology has become more widely available, emojis have become a common feature of digital communication. As a consequence, emojis have become a popular topic in various disciplines due to the additional context provided to textual data, for their ability to express emotion (Felbo et al. 2017).

The present study aims to implement two models based on fundamentally different architectures for comparison. This analysis could highlight innovative methods to improve performance at all stages of the pipeline.

2 Related work

2.1 SVM

Hayati and Muis 2019 analysed the effect of emotion incorporation in an emoji prediction experiment using an SVM. Extending Barbieri, Ballesteros, and Saggion 2017, this project began with the same dataset and implemented an SVM along with an additional task in which human participants suggested an emoji appropriate for each tweet. Hayati and Muis 2019 found that emojis which contain emotional information do improve the model's ability to predict an emoji in a tweet. While the human labelling of emojis provided additional information to the project which may have assisted the models in prediction, it may be biased (Godard and Holtzman 2022). Furthermore, Ahanin and Ismail 2022 argues that emojis in tweets act as measures of emotion. Emojis were categorised using multi-label emotion classification, showing that emojis can represent several emotions. This could explain the unexpected use of emojis found in related works.

Barbieri, Camacho-Collados, et al. 2018 explored emoji prediction on tweets in English and Spanish through multiple competing models. The twenty most frequent emojis were included. Models were evaluated using macro-averaged precision, recall, F1, and accuracy. This study suggests word and n-gram embeddings in SVM and neural network architectures provide a reliable methodology for emoji prediction, agreeing with Yang et al. 2023, which combined these architectures. Barbieri, Camacho-Collados, et al. 2018's findings are influential, however as the Spanish dataset is considerably smaller than the English dataset, the findings may not be representative of Spanish emoji use. Additionally, the authors do not explicitly state that cultural differences between language and emoji uses had been accounted for. However, this may be influential as differences in emoji use have been found between and within countries (Kejriwal et al. 2021), presenting an opportunity for future exploration.

As one of the systems submitted to Barbieri, Camacho-Collados, et al. 2018, Çöltekin and Rama 2018 found that linear SVM models produced superior results than the recurrent neural network models in the same tasks. These results highlight the importance of hyperparameter tuning in SVM implementation. For example, the authors suggest that character n-grams are advantageous, as morpheme information is captured, which would not be captured by word n-grams alone.

2.2 TimeLM

The following section discusses transformer based models, specifically BERT, and its variations, RoBERTa and TimeLM.

Barbieri, Camacho-Collados, et al. 2018 proposed a label-wise attention Bi-LSTM model for emoji prediction. They obtained a top-5 accuracy of 42.50%, indicating the importance of transformers in capturing context. Participating in OffensEval 2020 (Zampieri et al. 2020), Kurniawan, Budi, and Ibrohim 2020 used LSTM with pretrained Glove embeddings to detect offensive statements. Moreover, during the SemEval workshop (Barbieri, Camacho-Collados, et al. 2018), Baziotis et al. 2018 utilised Bi-LSTM with attention, achieving a top-1 accuracy of 44.74%. Although their approach is attention-based, the number of emojis used in their data set is relatively small, compared to the present study. That is, their approach using attention in neural network is not as challenging as using attention with 100 different emojis.

The aforementioned models demonstrate the importance of the attention mechanism. This motivates the exploration of transformer models.

To begin, Nusrat et al. 2023 fine-tuned BERT on datasets with 5 and 20 labels. Their approach displays state-of-the-art performance. However, their study did not fine-tune a transformer-based language model on more diverse emojis. On the other hand, the current study extends their chosen number of emojis to 100.

Furthermore, Ma et al. 2020 fine-tuned BERT on eight successive datasets with 20 to 300 classes. They demonstrated outstanding results across top-1 and top-5 accuracy, and F1-score. The authors used the DeepMoji model as a comparison. Unlike the present study, which implements a transformer-free machine learning and a transformer-based language models, they compared two transformers-based language models.

Singh et al. 2022 proposed a single emoji prediction system with BERT, ALBERT, and RoBERTa. They achieved accuracies of 61%, 63%, and 60% respectively.

However, the dataset is limited to tweets from October 2015 to November 2018. Contrastingly, the present study's dataset ranges from 2020 to 2023. The usage of emojis over time may differ between their dataset and that of the current paper.

Tomihira et al. 2020 fine-tuned BERT on a 20-emoji, English and Japanese dataset. They preprocessed by removing URLs, hashtags, retweets, and mentions. In the current study, hashtags were not removed to provide richer contextual cues.

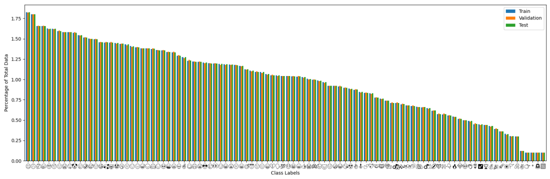
The SuperTweetEval benchmark (Antypas et al. 2023) used the same dataset as this paper, implementing the task across eight different language models. One of them is TimeLM, a RoBERTa model trained on 154 million tweets introduced by CardiffNLP. They concluded that emoji prediction remains challenging. In the benchmark, they used Ray Tune as an optimizer for fine-tuning hyperparameters. However, this approach is resource-intensive. To mitigate this here, the consensus upon the hyperparameters was made after many attempts of randomising and fine-tuning.

Although these studies produced state-of-art results, they did not compare between a domain-specific language model, such as the TimeLM, and traditional machine learning algorithms, such as SVM. This presents a gap in the literature which the present project aims to fill.

3 Exploratory data analysis

3.1 Dataset analysis

The dataset consisted of timestamped tweets ending in a single emoji, which was removed to become the tweet's class label. The emojis used as labels were limited to the 100 most frequent. Training, validation and test splits were provided with ratio 50000/50000/5000.



Listing 1: Plot of emoji class sizes

List.1 demonstrates that there is significant class imbalance, with mean class size 500 and standard deviation 216.4. Moreover, tweets are mixed-case. All URLs are masked with URL, and non-verified user mentions are replaced with @user.

3.2 Date EDA

To investigate the date data, the texts were vectorized using the mean of w2vec vectors from the google-news dataset. PCA reduced the vectors to 2D for visualisation. Training tweets for the top 10 emojis were plotted.

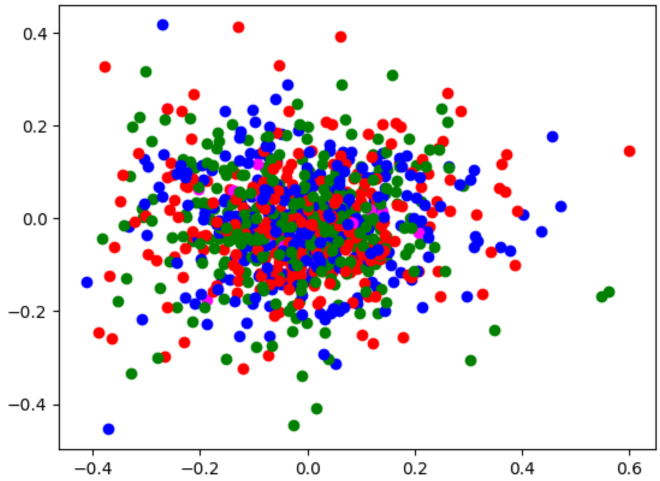
Year and time-of-day were investigated.

Six categories were used for time-of-day: early morning (0-4), morning(5-10), noon(11-14), afternoon(15-17), evening(18-20) and night(21-23). The intuition is that people's moods and thus writing style and emoji use change throughout the day.

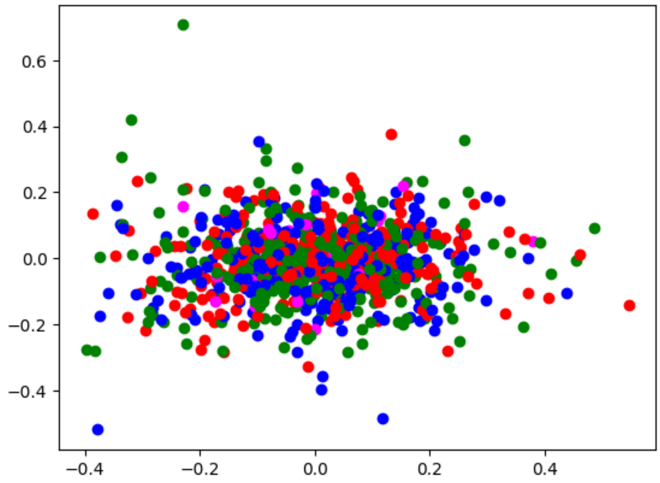
A few examples are shown.

3.2.1 Year

The colour scheme is blue for 2020, green for 2021, red for 2022, magenta for 2023.



Listing 2: Plot of years for 🥴

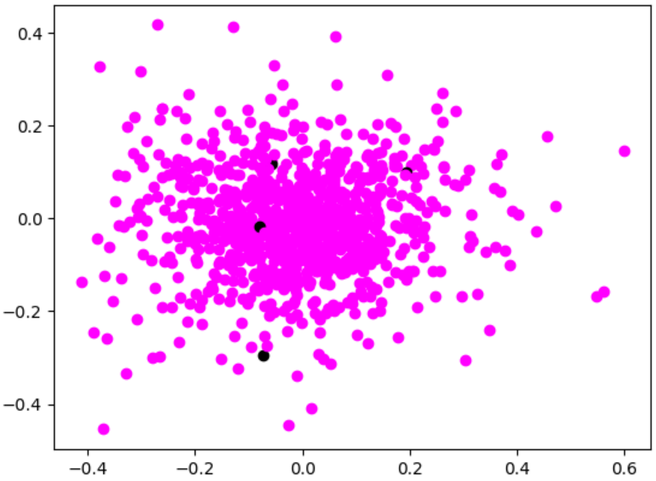


Listing 3: Plot of years for 🙃

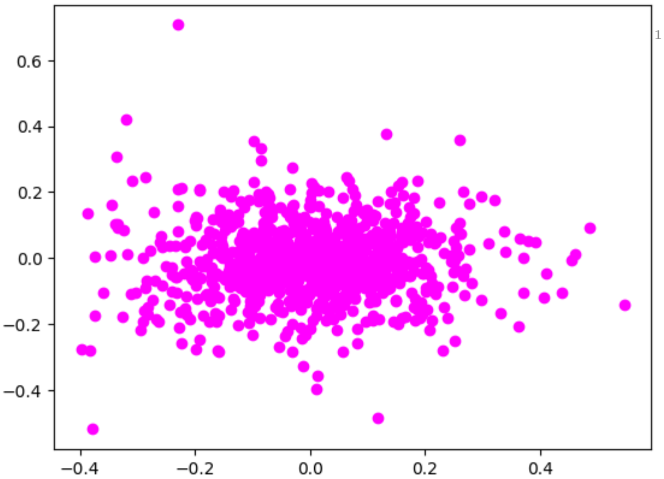
No clear pattern could be discerned for year, hence no significant impact on performance was expected.

3.2.2 Time of day

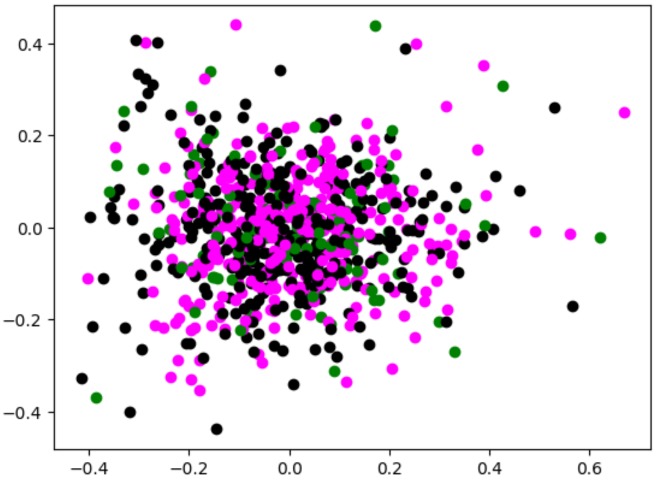
The colour scheme is blue for early morning, cyan for morning, red for noon, green for afternoon, black for evening and magenta for night.



Listing 4: Plot of time-of-day for 🥴



Listing 5: Plot of time-of-day for 🙃

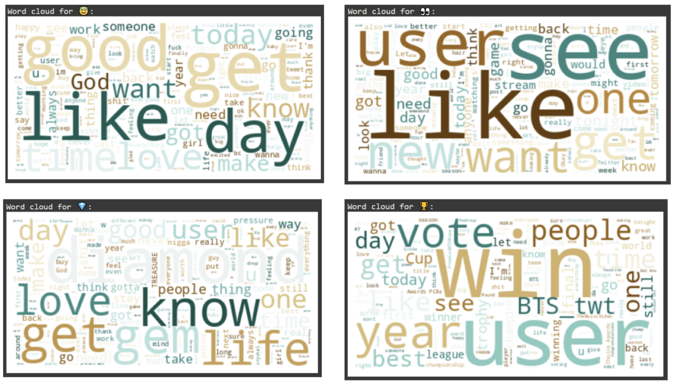


Listing 6: Plot of time-of-day for 🤧

For half of the top 10 emojis, one category dominates, with 3 being exclusively used at night, hence this feature strongly correlates with particular labels.

3.3 Semantics EDA

To explore to what extent the vocabulary used in each tweet reflects the corresponding emoji, word clouds were generated by whitespace-tokenising all tweets for a given emoji, removing stopwords via the NLTK package, and applying the wordcloud package to create the final plots. List.7 shows word clouds for four emojis.


Word cloud for 😇:

Word cloud for 👀:

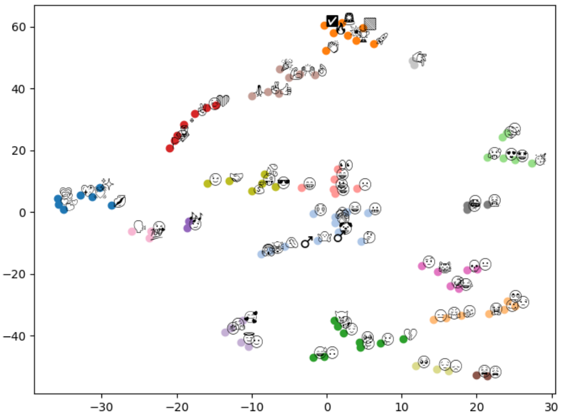
Word cloud for 💎:

Word cloud for 🏆:

Listing 7: Word clouds

The word cloud of each emoji contained vocabulary suggesting its meaning: the cloud for 👀 contains ‘look’, and related concepts: ‘game’, ‘stream’, ‘video’. This motivates incorporating word frequency as a feature. However, word clouds share some common words, even after stopword removal, such as ‘like’, ‘user’, and ‘get’. This justifies using methods such as tf-idf embeddings and feature selection to devalue the impact of these words in the models. Moreover, being difficult to identify any one word indicative for each emoji, methods capturing more information about word context, such as ngrams, may be advantageous.

Beaulieu and Asamoah Owusu 2018 observe that models struggle to distinguish between emojis with similar meanings. The similarity of emoji labels is investigated. Whilst defining ‘emoji similarity’ is its own problem of interest, further inspiration is taken from Barbieri, Ronzano, and Saggion 2016, who created vector representations of emoji through a large text corpus; and Eisner et al. 2016, who created vector representations through summing Word2Vec word embeddings. Combining these approaches, the following was performed: data splits were gathered and whitespace tokenisation applied. Then, a Word2Vec model was trained, with default parameters. The embeddings obtained for all words in each tweet were summed. The resulting representations were averaged for all tweets in a particular emoji class to obtain a representation for that emoji. The dimensionality of the data was reduced via the t-distributed stochastic neighbour embedding (TSNE) method, available from the sklearn library, which aims to preserve the pairwise distances between points during transformation. Finally, k-means clustering was used to group the emoji. After some experimentation 20 clusters were used, as this was judged to create enough clusters to fit the trends of the plotted data, without causing each cluster to become unreasonably small. The results of this are shown in List.8.



Listing 8: Emoji clusters

The clusters obtained tend to correspond with intuitive ideas of emoji similarity. For example, the heart emojis 💕💖💗💞💓 are grouped together.

4 Pre-processing

4.1 Pre-processing methods for the first model

Several feature extraction methods were used. Based on Çöltekin and Rama 2018, character bag of n-grams and bag of n-grams methods were the first to be considered. Tf-idf was then used to provide a comparison between the various frequency-based feature extraction methods. Lastly, a static word embedding method was used, Word2Vec.

These methods were used to pre-process the text. The date was also used to create features. Selectkbest was used for feature selection.

4.1.1 Date pre-processing

The year, month and day were extracted in numeric form. Time of day was included on the basis of promising EDA analysis. Preliminary testing of introducing this feature made quite a jump in performance, with top performing models before this change achieving around 19% top-5 accuracy and subsequent models surpassing 22% top-5 accuracy.

4.1.2 Word2Vec

Word embedding methods were introduced as a computationally modest way of adding more context. This was used with the pre-trained google-news dataset (”word2vec-google-news-300”) for its large size and ease of use.

4.1.3 Character bag of n-grams

Each tweet was split into characters and a CountVectorizer was applied, with lowercasing disabled and ASCII characters as vocabulary. **Bag of n-grams** As referenced previously, this allowed the model to capture information on word contexts. **TfIdf** This method was introduced as a possible alternative to bag of n-grams, as they are both frequency based methods.

As part of preprocessing, word n-grams were formed from the data, with the size of n-grams left as a parameter. Tf-idf weightings were applied to minimise the impact of tokens common across tweets.

4.1.4 Feature selection - SelectkBest

After training, the k best features were selected, for some parameter k. The least informative features were removed as they are less likely to generalise well to unseen test data (Jurafsky and Martin 2024).

4.2 Pre-processing methods for the second model

4.2.1 Introduction

Unlike SVMs, URLs, mentions, and special characters were removed. According to Lee, Jeong, and Park 2022, hashtags enrich meaning by providing contextual cues, thus were not removed.

5 Methodology

5.1 LinearSVM model

5.1.1 Using SVMs for multinomial classification

The classification approaches One-vs-One and One-vs-Rest were considered. One-vs-One is quadratic in number of classes, while One-vs-Rest is linear. Due to performance concerns, One-vs-Rest was used.

5.1.2 SVMs and Kernels – a short introduction

Given a set of inputs X and a set of binary labels Y, and an unseen input x, the task requires predicting a label y. The approach of SVMs is to map the inputs into a multidimensional vector space (most often a Hilbert space, since a dot product is a very convenient way of creating a kernel; this process is called the kernel trick), then create a decision boundary -a hyperplane- that separates the two label categories. To classify an unseen input x, simply observe which side of the hyperplane it is situated on and classify.

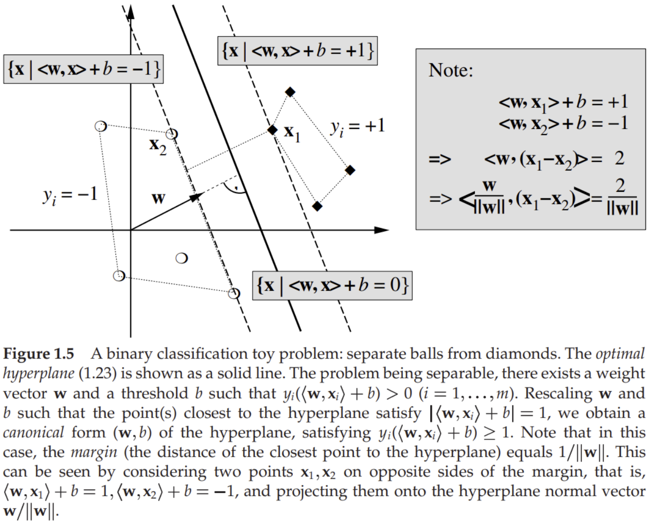


Figure 1.5 A binary classification toy problem: separate balls from diamonds. The *optimal hyperplane* (1.23) is shown as a solid line. The problem being separable, there exists a weight vector $\mathbf{w}$ and a threshold b such that $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) > 0$ ($i = 1, \ldots, m$). Rescaling $\mathbf{w}$ and b such that the point(s) closest to the hyperplane satisfy $|\langle \mathbf{w}, \mathbf{x}_i \rangle + b| = 1$, we obtain a *canonical* form $(\mathbf{w}, b)$ of the hyperplane, satisfying $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$. Note that in this case, the *margin* (the distance of the closest point to the hyperplane) equals $1/\|\mathbf{w}\|$. This can be seen by considering two points $\mathbf{x}_1, \mathbf{x}_2$ on opposite sides of the margin, that is, $\langle \mathbf{w}, \mathbf{x}_1 \rangle + b = 1, \langle \mathbf{w}, \mathbf{x}_2 \rangle + b = -1$, and projecting them onto the hyperplane normal vector $\mathbf{w}/\|\mathbf{w}\|$.

Listing 9: Finding optimal hyperplane (Scholkopf and Smola 2001 p.12)

5.1.3 Kernel types

Linear, RBF, and polynomial kernels were all tested in the present project. Only the linear will be discussed in this report due to the poor performance of the other two. The simplicity, computational affordability and superior accuracy scores found by this study led to implementing the linear kernel.

5.1.4 Pipeline

The ML pipeline consists of pre-processing steps, an SVM classifier, and evaluation methodology. Implementation followed the blueprint of the sklearn pipeline.

5.1.5 Hyperparameter tuning

Due to the number of hyperparameters seen in the following sections, a heuristic based tuning algorithm was used: simulated annealing. The user can tweak the number of runs before an optimum is found.

The hyperparameter search space at the start of the search is explored regardless of whether the move makes sense (moving from a high to a low accuracy is very probable). As the system “cools down”, the less likely this is to happen, and the probability of this event is decided by the Acceptance Function. The algorithm is used when the search space cannot be fully explored due to memory and time constraints. The lowest energy (highest validation set top5 accuracy) state is the goal.

5.1.6 Hyperparameter states

A state class that encapsulates hyperparameter combinations was used to facilitate tuning, which also includes helper functions for finding a neighbour state-a state that has slightly different hyperparameters (e.g. C parameter 0.12 vs 0.14 or n-gram range (1,2) vs (1,3)).

5.1.7 Initial state

The starting state was set to random. Based on several previous projects (Mostafaie, Modarres Khiyabani, and Navimipour 2020), this is standard when using simulated annealing.

5.1.8 Probability array

Given a list of class rankings from a decision_function, a probability distribution can be obtained with softmax.

5.1.9 C parameter

The C parameter controls how tight the fitting of the decision boundary should be. For smaller values of C, slack variables won’t carry as much weight and outliers will be misclassified (Scholkopf and Smola 2001). The opposite is true for higher values of C.

5.1.10 Max_iter

This hyperparameter monitored where the line between diminishing returns and accuracy gains was when considering the number of solver iterations.

5.1.11 N-gram_size

N-gram size ranges from 1 to 4 words in length. The reasoning behind this threshold was two-fold; Moore and Quirk 2009 suggested that n-grams 4 words long are a practical boundary and that model sizes with n-grams longer than 4 words began to suffer significant performance loss. The second reason for this threshold was for simple comparison to the findings of Çöltekin and Rama 2018.

5.1.12 Char-n-gram_size

As one of the hyperparameters used in Çöltekin and Rama 2018, this was kept. The range of values for the n-grams is between (1,1) and (1,6).

5.1.13 Feature Engineering Combinations

Different combinations of feature engineering were used for simulated annealing. The feature engineering methods involved in different combinations were Word2Vec, Bag of n-grams, Bag of character n-grams, and Tfidf vectorizer. Date preprocessing was also present, although this hyperparameter is constant so will not be discussed in this context. To find different combinations, a binary system was implemented. This works by flipping one of the four bits at random, giving a new hyperparameter combination with each flip.

5.1.14 Selectkbest k parameter

The k parameter from selectkbest was used to balance the curse of dimensionality with having a sufficient number of features.

5.1.15 Acceptance function heuristics

The acceptance function used is:

$$e^{\frac{100}{30}\cdot 1.40670535838\cdot 1.15\cdot(Energy_{curr\ state}-Energy_{neigh})\cdot\frac{1}{Temp}}$$

One of the important parts of this task is figuring out a suitable coefficient in the exponent. The coefficient was created through the following steps:

Multiplication by 100 normalised energy differences from [0, 1] to [0, 100].

Division by 30 normalised the difference between the best performing and worst performing model's energies to 1, as the best performing SVM models could not surpass 30% accuracy.

By this point, the exponent is a number ranging from [0, 1] that could theoretically reach its bounds.[1] 1.40670535838 is the coefficient needed to obtain a 6% chance of accepting an always-wrong model (validation-set top-5 accuracy of 0%) coming from a perfect model (validation-set top-5 accuracy of 100%) when the temperature has already halved.

$$e^{Coef\cdot EnergyDifference\cdot\frac{1}{Temperature}}=0.06$$
$$e^{Coef\cdot(-1)\cdot\frac{1}{0.5}}=0.06$$
$$Coef=-\ln 0.06\cdot 0.5$$
$$Coef=1.40670535838$$

Lastly, 1.15 was introduced to obtain slightly lower probabilities of a good-to-terrible model jump, since the number of runs (200) was not large and conservative jumps are preferred to avoid getting stuck in low-accuracy states.

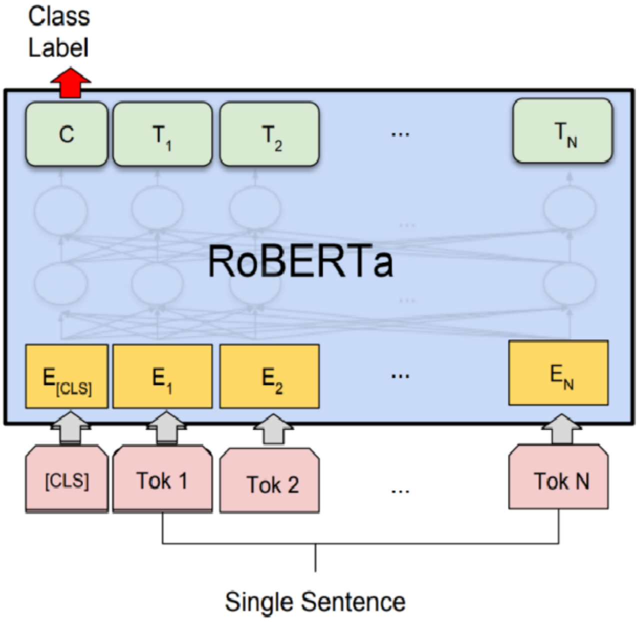
5.2 TimeLM model

5.2.1 Introduction

This model is pre-trained on tweets from January 2018 to December 2022. That is, the dataset used to fine-tune the model also encompasses tweets gathered from the same time frame.

5.2.2 Model Architecture

TimeLM, a variant of the RoBERTa architecture, is pre-trained on 154 million tweets. This aims to better capture the idiosyncrasies of informal language found in the tweets (Khusuma, Maharani, and Gani 2023).



Listing 10: The architecture of RoBERTa (Khusuma, Maharani, and Gani 2023)

5.2.3 AutoTokenizer

The TimeLM AutoTokenizer was used to pre-process the data. During tokenisation special tokens such as [CLS] and [SEP] were added, denoting the sentence's start and end, respectively. Tokens were converted into "input IDs" that the model understands.

5.2.4 Handling Class Imbalance

Oversampling techniques were abandoned in this study due to their poor performance in the dataset. Therefore, class weights were calculated. The model compensated by adjusting the loss calculation, prioritizing infrequent emojis during learning.

5.2.5 Hyperparameters

After manual trials, the following hyperparameters were settled upon.

Learning Rate Set to a low value of $5e-6$, avoiding aggressive learning. This value helped the model capture the linguistic nuances.

Epochs The model was trained for 15 epochs due to the large number of labels, thereby training well without overfitting.

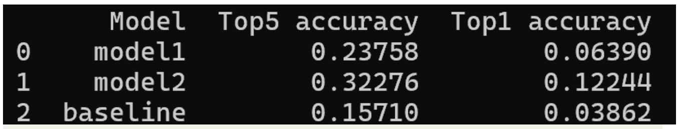
Optimiser The optimiser used was AdamW, which supported effective generalisation by adapting the learning rates based on the gradients while integrating weight decay into the optimisation process.

Training Batches Linguistically, a low batch size was chosen to adjust the model's parameters more frequently, better capturing the implications of the emojis.

6 Evaluation

Antypas et al. 2023 and Lee, Jeong, and Park 2022 are followed in reporting top-5 accuracy. Top-1 accuracy is also mentioned where relevant. Top-5 is well-suited for the task, as users often select from a set of suggestions rather than using one prediction.

The models were compared against a baseline model with Multinomial Bayes and Tfidf with size 2 n-grams.



```
      Model  Top5 accuracy  Top1 accuracy
0    model1        0.23758        0.06390
1    model2        0.32276        0.12244
2  baseline        0.15710        0.03862
```

Listing 11: Performance metrics for the two models and the baseline Naive Bayes model

6.1 LinearSVM

LinearSVM performed noticeably better than the baseline model.

Among the 200 runs of the algorithm, the following combination performed the best:

```
C parameter: 0.30000000000000004
n-gram parameter: (1,1)
char n-gram parameter: (1,4)
feature engineering used: bag_of_char_ngrams; bag_of_ngrams; date pre-processing
selectkbest k parameter: 500
max_iter parameter: 16000
```

Due to the nature of a heuristic search algorithm like Simulated Annealing, the top accuracy is not a guaranteed global optimum. It is rather a case of using limited computing resources (a cap on the number of runs, 200) as efficiently as possible. For comparison, a grid search would explore $50 \cdot 4 \cdot 6 \cdot 4 \cdot 12 \cdot 50 = 2880000$ possible combinations of hyperparameters.

6.2 TimeLM

TimeLM performed better than both baseline and LinearSVM.

On the test dataset, the model achieved a Top-1 Accuracy of 12.24% and a Top-5 Accuracy of 32.28%, indicating the task remains challenging. Its performance fell slightly short of the SuperTweetEval benchmark, which achieved a result of 35.46%. Future research efforts shall aim to match or surpass this benchmark.

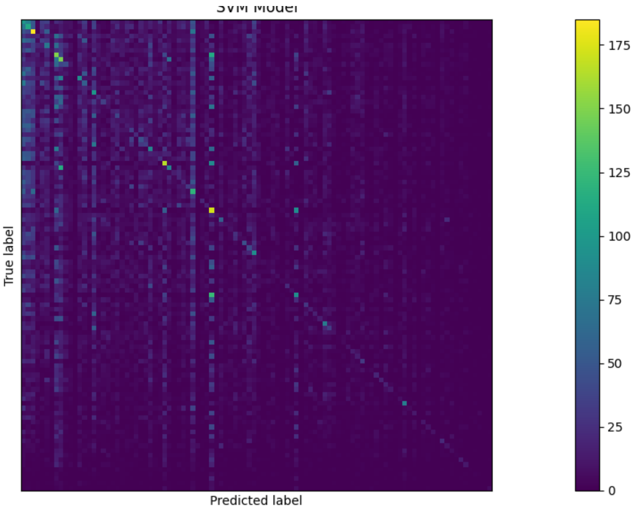
7 Error Analysis

As the baselines presented in Antypas et al. 2023 indicated, the emoji-prediction task was challenging for both architectures present here. The following section investigates how much this was due to the choice of models, or was an innate feature of the given data.

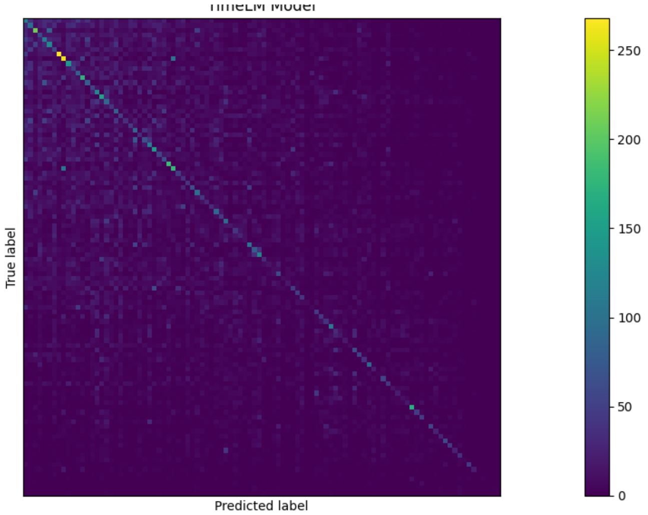
The TimeLM model inherited limitations from its RoBERTa architecture. Optimal choice of hyperparameters was a prominent influence on model performance (Sy et al. 2024). However, Semary et al. 2023 note that RoBERTa's complexity means it is computationally expensive, requiring long times to fine-tune it, so a complete hyperparameter search was not performed, potentially resulting in sub-optimal performance.

Moreover, one known limitation of SVM models is their behaviour on unbalanced datasets: Batuwita and Palade 2013 suggests that in this case SVMs made predictions biased in favour of the more dominant classes.

Through plotting confusion matrices with axes reordered from most- to least-frequent emojis, one observes weak clouds of misclassifications in the top-left-hand corners. So, emojis were more likely to be misclassified as the dominant classes.



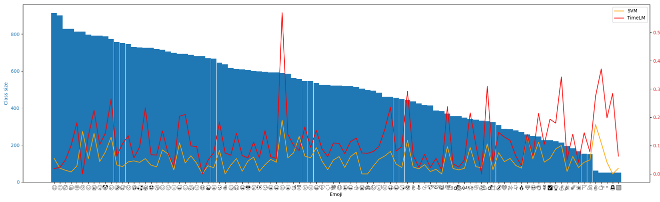
Listing 12: Confusion matrix SVM



Listing 13: Confusion matrix TimeLM

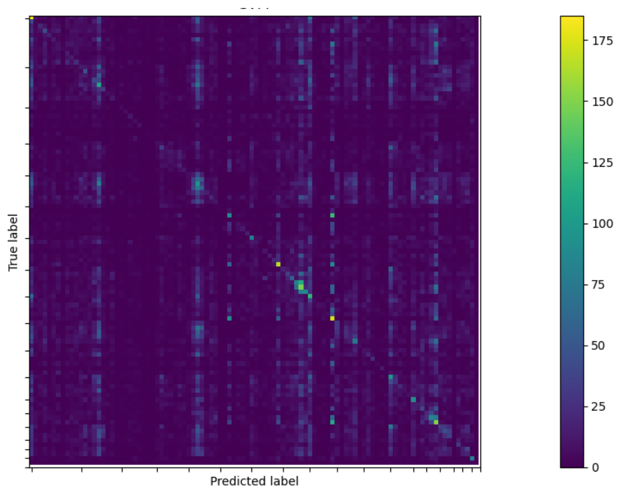
To explore this idea further, a bar chart of class size was plotted, and graphs of the individual f1-score for each class overlaid. One might intuitively expect the more dominant classes perform better, but List.14 demonstrates this was not the case. No clear correlation between class size and performance was observed. Benkendorf et al. 2023 supports this, suggesting that unbalanced datasets contributed to poor model performance. The emoji 🚨 and 👉, despite having a class size of 50, both outperformed the most common emoji 🥴🙃🤔🤧🙄, which all had a class size over 800. Thus, there seems to be some other characteristic of the data much more significant than class-imbalance in explaining the difference in model performance on different emoji.

This could be explained by the difference in the ways in which these emojis were consistently used. For example, 🚨 was frequently used to denote urgency in the present dataset, while face emojis are arguably more polysemous (Shardlow, Gerber, and Nawaz 2022). Moreover, Scheffler and Nenchev 2024 showed that some face emojis can be used in various and unexpected contexts, creating ambiguity around their meanings.

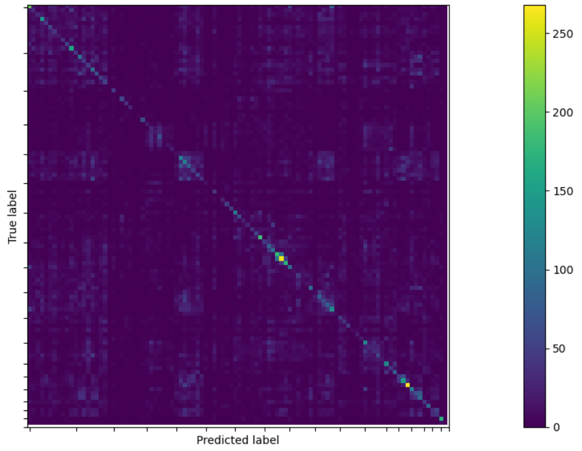


Listing 14: Class-size performance

Another possible characteristic was the similarity of certain emojis. Recall from earlier the clustering of emojis by semantic similarity. Redrawing the confusion matrices with emojis in the same cluster grouped together on the axes, a weak pattern of boxes around the diagonal appeared, suggesting that misclassified tweets were often misclassified as emojis in the same cluster as the gold label.



Listing 15: Cluster confusion matrix SVM



Listing 16: Cluster confusion matrix TimeLM

Some emojis appeared to be outliers in their cluster – for example, the emoji ☹️ in a cluster of otherwise positive emojis – but that their vector representations were the sum of Word2Vec word embeddings suggests these outliers occur in contexts similar to other emojis in their cluster. Hence, some emojis have multiple meanings, and each tweet could have multiple emoji depending on the original writer's intended meaning. This relates to Ahanin and Ismail 2022, who found that multiple emojis may apply to a single tweet depending on the tweet's overall sentiment. Table 17 demonstrates the variety of sentiments of tweets labelled 😢.

Text	Top-5 Predictions (SVM Model)
YAll. I go back to work next Friday.......	😁 😒 🙃 😅 🥴
SUperbbb	🥰 🤔 😋 👀 🥴
i fkn hate finals week. the stress i've been under is unreal	😅 😔 🤣 🤬 😒
LOOOL this movie night has finished me	🥰 🥴 🤯 😒 😍
Thankful for my person	🙏 □ 💯 😒 💕
It's so hot my AC can't keep up w it	🙃 😫 😌 🥵 💔

Listing 17: Polysemy

For each tweet, the model failed to predict the gold label, but predicted several emojis which match its sentiment, or could match the text under a different intended meaning.

	SVM		TimeLM	
	Near Matches / Emoji	Misses / Emoji	Near Matches / Emoji	Misses / Emoji
😒	**117.17**	29.78	**175.50**	25.76
🤔	**99.50**	32.84	**94.00**	34.49
😊	**76.00**	24.92	**104.67**	23.84
🔥	3.33	**14.04**	11.67	**12.60**
🤦	**42.00**	15.33	**45.00**	14.49

Listing 18: Near miss counts

To explore whether all emojis reflected this, the following was performed: in the top-5 setting, for each gold label, the number of times each model misclassified a tweet as an emoji in the same cluster (a ‘near match’), or outside of the cluster (a ‘miss’), were counted. These counts were divided by the number of emojis inside and outside of the cluster, respectively. Table 18 shows that emojis in the same cluster as the gold label generally comprised a greater proportion of misclassifications than emojis outside of the cluster. Hence, the models tended to misclassify emojis as other similar emojis.

Therefore, parsing between semantically similar emoji was the error most frequently made by both models. Differentiating between emojis typically used to express similar emotions is a task which humans may struggle with (Shardlow, Gerber, and Nawaz 2022). Therefore, these findings seem to be in keeping with expected human behaviour.

8 Discussion of hyperparameter tuning

8.1 Using the date as a feature

Adding the date data caused the best performing model’s top-5 accuracy to increase from 19% to 23.758%. Including just time-of-day improved accuracy by 3%. This finding strongly points towards a need to explore novel ways of feature creation.

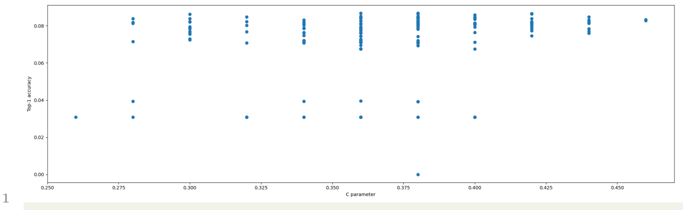
8.2 Hyperparameter importance for LinearSVM

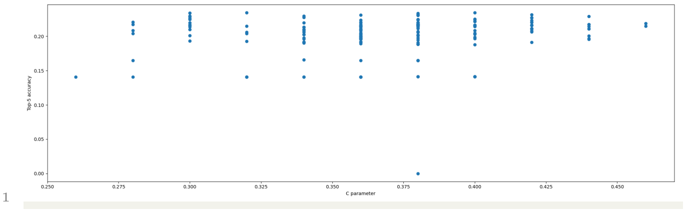
The importance of the six hyperparameters used in the first model was also tested.

Possible hyperparameter values were plotted on the x-axis, with corresponding accuracy values on the y-axis, for both top-5 and top-1 accuracy. For the raw data, see ”results_200_1.txt”.

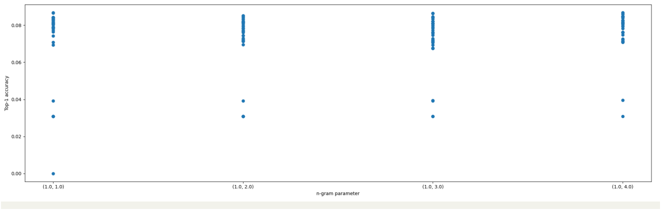
As seen below, C, n-gram, char-n-gram and feature_engineering_combinations didn’t produce large variations. However, max_iter showed a preference for values above 8000, and selectkbest k showed a preference for low-moderate values, with the best performing model needing only 500 features.

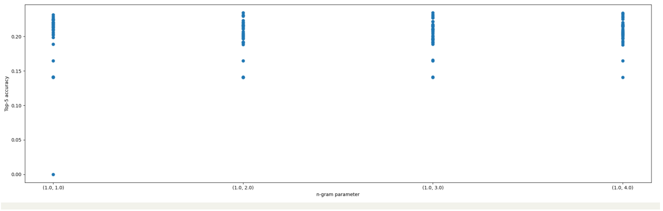
8.2.1 C parameter



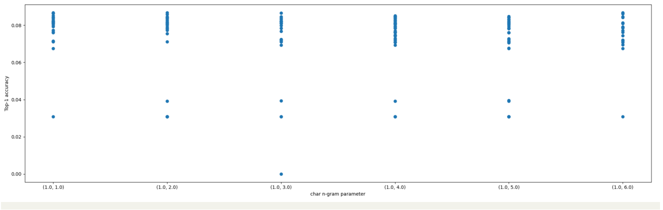


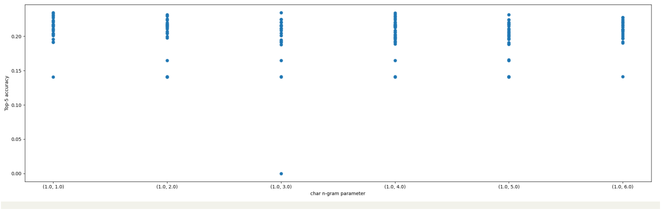
8.2.2 n-gram parameter



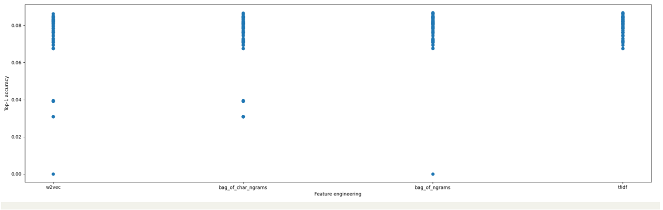


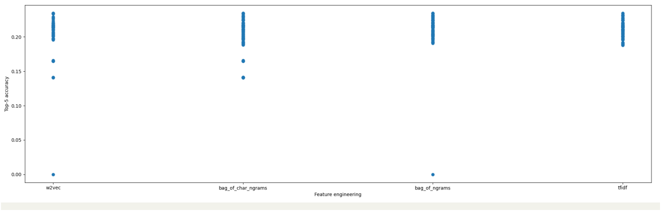
8.2.3 char-n-gram parameter



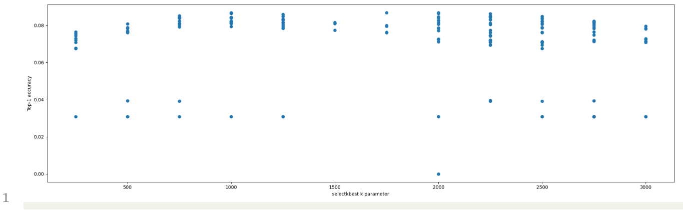


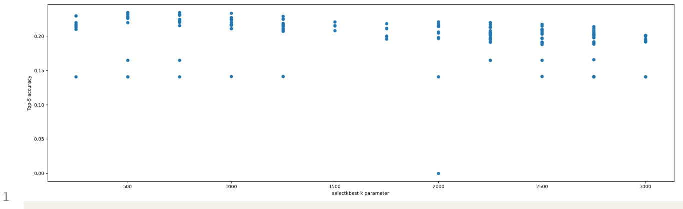
8.2.4 Feature engineering combinations



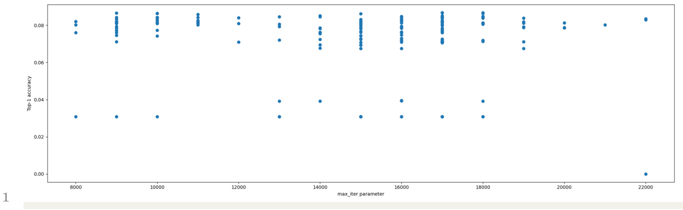


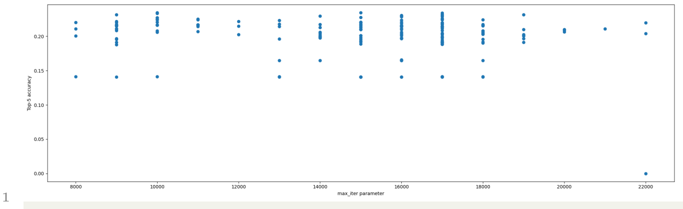
8.2.5 Selectkbest k parameter





8.2.6 max_iter parameter





8.3 The result of tuning the first model

As seen in Çöltekin and Rama 2018 and corroborated by the findings of this study, a combination of bag of n-grams and character n-grams is ideal for feature creation for this task.

9 Future developments

One interesting area that remained unexplored in this study is the usage of proper hyperparameter tuning for the TimeLM model, as well as figuring out a way to introduce date pre-processing into the TimeLM pipeline.

Another area worth exploring is using less computationally expensive Rbf and Poly kernels, by using SGDClassifier's approximation of these kernels.

Even though many feature extraction methods were used in the first model, more varied methods such as Part-of-speech tagging could be used.

Given the informal setting of the corpus, further pre-processing such as stripping repeated characters may be applicable.

10 Considerations of ethics

Even though data was anonymised, Twitter users might not expect their tweets to be collected.

11 Conclusion

The present project aimed to determine which of the two machine learning models created performed better in the task of predicting the emoji accompanying a given tweet. Overall, the current study found that the TimeLM model performed better than the SVM in both top-1 and top-5 accuracy metrics.

However, the SVM architecture showed promising results, particularly after training the model on data including dates and time of day. The optimal model produced by model 1 was that in which bag of character n-grams and bag of word n-grams were implemented as part of the simulated annealing algorithm, which corroborates the findings of Çöltekin and Rama 2018.

Additionally, the better performance of model 2 showcased the importance of the attention mechanism. These findings have also highlighted new approaches to the topic of emoji prediction by introducing emoji clustering methods.

References

Ahanin, Zahra and Maizatul Akmar Ismail (2022). "A multi-label emoji classification method using balanced pointwise mutual information-based feature selection". In: *Computer Speech Language* 73, p. 101330. ISSN: 0885-2308. DOI: `https://doi.org/10.1016/j.csl.2021.101330`. URL: `https://www.sciencedirect.com/science/article/pii/S0885230821001236`.

Antypas, Dimosthenis et al. (Dec. 2023). "SuperTweetEval: A Challenging, Unified and Heterogeneous Benchmark for Social Media NLP Research". In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, pp. 12590–12607. DOI: `10.18653/v1/2023.findings-emnlp.838`. URL: `https://aclanthology.org/2023.findings-emnlp.838/`.

Barbieri, Francesco, Miguel Ballesteros, and Horacio Saggion (Apr. 2017). "Are Emojis Predictable?" In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Ed. by Mirella Lapata, Phil Blunsom, and Alexander Koller. Valencia, Spain: Association for Computational Linguistics, pp. 105–111. URL: `https://aclanthology.org/E17-2017/`.

Barbieri, Francesco, Jose Camacho-Collados, et al. (June 2018). "SemEval 2018 Task 2: Multilingual Emoji Prediction". In: *Proceedings of the 12th International Workshop on Semantic Evaluation*. Ed. by Marianna Apidianaki et al. New Orleans, Louisiana: Association for Computational Linguistics, pp. 24–33. DOI: `10.`

18653/v1/S18-1003. URL: https://aclanthology.org/S18-1003/.

Barbieri, Francesco, Francesco Ronzano, and Horacio Saggion (May 2016). “What does this Emoji Mean? A Vector Space Skip-Gram Model for Twitter Emojis”. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC‘16)*. Ed. by Nicoletta Calzolari et al. Portorož, Slovenia: European Language Resources Association (ELRA), pp. 3967–3972. URL: https://aclanthology.org/L16-1626/.

Batuwita, Rukshan and Vasile Palade (2013). “Class Imbalance Learning Methods for Support Vector Machines”. In: *Imbalanced Learning*. John Wiley amp; Sons, Ltd. Chap. 5, pp. 83–99. ISBN: 9781118646106. DOI: https://doi.org/10.1002/9781118646106.ch5. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118646106.ch5. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118646106.ch5.

Baziotis, Christos et al. (2018). *NTUA-SLP at SemEval-2018 Task 1: Predicting Affective Content in Tweets with Deep Attentive RNNs and Transfer Learning*. arXiv: 1804.06658 [cs.CL]. URL: https://arxiv.org/abs/1804.06658.

Beaulieu, Jonathan and Dennis Asamoah Owusu (June 2018). “UMDuluth-CS8761 at SemEval-2018 Task 2: Emojis: Too many Choices?” In: *Proceedings of the 12th International Workshop on Semantic Evaluation*. Ed. by Marianna Apidianaki et al. New Orleans, Louisiana: Association for Computational Linguistics, pp. 400–404. DOI: 10.18653/v1/S18-1061. URL: https://aclanthology.org/S18-1061/.

Benkendorf, Donald J. et al. (2023). “Correcting for the effects of class imbalance improves the performance of machine-learning based species distribution models”. In: *Ecological Modelling* 483, p. 110414. ISSN: 0304-3800. DOI: https://doi.org/10.1016/j.ecolmodel.2023.110414. URL: https://www.sciencedirect.com/science/article/pii/S030438002300145X.

Çöltekin, Çagri and Taraka Rama (2018). “Tübingen-Oslo at SemEval-2018 Task 2: SVMs perform better than RNNs in Emoji Prediction”. In: *International Workshop on Semantic Evaluation*. URL: https://api.semanticscholar.org/CorpusID:43958367.

Eisner, Ben et al. (Nov. 2016). “emoji2vec: Learning Emoji Representations from their Description”. In: *Proceedings of the Fourth International Workshop on Natural Language Processing for Social Media*. Ed. by Lun-Wei Ku, Jane Yung-jen Hsu, and Cheng-Te Li. Austin, TX, USA: Association for Computational Linguistics, pp. 48–54. DOI: 10.18653/v1/W16-6208. URL: https://aclanthology.org/W16-6208/.

Felbo, Bjarke et al. (Sept. 2017). “Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm”. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Ed. by Martha Palmer, Rebecca Hwa, and Sebastian Riedel. Copenhagen, Denmark: Association for Computational Linguistics, pp. 1615–1625. DOI: 10.18653/v1/D17-1169. URL: https://aclanthology.org/D17-1169/.

Godard, Rebecca and Susan Holtzman (2022). “The Multidimensional Lexicon of Emojis: A New Tool to Assess the Emotional Content of Emojis”. In: *Frontiers in Psychology* 13. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2022.921388. URL: https://www.frontiersin.org/journals/psychology/articles/10.3389/f%20psyg.2022.921388.

Hayati, Shirley Anugrah and Aldrian Obaja Muis (June 2019). “Analyzing Incorporation of Emotion in Emoji Prediction”. In: *Proceedings of the Tenth Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. Ed. by Alexandra Balahur et al. Minneapolis, USA: Association for Computational Linguistics, pp. 91–99. DOI: 10.18653/v1/W19-1311. URL: https://aclanthology.org/W19-1311/.

Jurafsky, Daniel and James H. Martin (2024). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. 3rd. Online manuscript released August 20, 2024. URL: https://web.stanford.edu/~jurafsky/slp3/.

Kejriwal, Mayank et al. (2021). “An empirical study of emoji usage on Twitter in linguistic and national contexts”. In: *Online Social Networks and Media* 24, p. 100149. ISSN: 2468-6964. DOI: https://doi.org/10.1016/j.osnem.2021.100149. URL: https://www.sciencedirect.com/science/article/pii/S2468696421000318.

Khusuma, Rianda, Warih Maharani, and Prati Gani (Feb. 2023). “Personality Detection On Twitter User With RoBERTa”. In: *JURNAL MEDIA INFORMATIKA BUDIDARMA* 7, p. 542. DOI: 10.30865/mib.v7i1.5598.

Kurniawan, Sandy, Indra Budi, and Muhammad Okky Ibrohim (Dec. 2020). “IR3218-UI at SemEval-2020 Task 12: Emoji Effects on Offensive Language IdentifiCation”. In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Ed. by Aurelie Herbelot et al. Barcelona (online): International Committee for Computational Linguistics, pp. 1998–2005. DOI: 10.18653/v1/2020.semeval-1.263. URL: https://aclanthology.org/2020.semeval-1.263/.

Lee, SangEun, Dahye Jeong, and Eunil Park (2022). “MultiEmo: Multi-task framework for emoji prediction”. In: *Knowledge-Based Systems* 242, p. 108437. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2022.108437. URL: https://www.sciencedirect.com/science/article/pii/S0950705122001794.

Loureiro, Daniel et al. (2022). “TimeLMs: Diachronic Language Models from Twitter”. In: *CoRR* abs/2202.03829. arXiv: 2202.03829. URL: https://arxiv.org/abs/2202.03829.

Ma, Weicheng et al. (2020). *Emoji Prediction: Extensions and Benchmarking*. arXiv: 2007.07389 [cs.CL]. URL: https://arxiv.org/abs/2007.07389.

Moore, Robert C. and Chris Quirk (Aug. 2009). “Improved Smoothing for N-gram Language Models Based on Ordinary Counts”. In: *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*. Ed. by Keh-Yih Su et al. Suntec, Singapore: Association for Computational Linguistics, pp. 349–352. URL: https://aclanthology.org/P09-2088/.

Mostafaie, Taha, Farzin Modarres Khiyabani, and Nima Jafari Navimipour (2020). “A systematic study on meta-heuristic approaches for solving the graph coloring problem”. In: *Computers Operations Research* 120, p. 104850. ISSN: 0305-0548. DOI: https://doi.org/10.1016/j.cor.2019.104850. URL: https://www.sciencedirect.com/science/article/pii/S0305054819302928.

Nusrat, Muhammad Osama et al. (2023). *Emoji Prediction in Tweets using BERT*. arXiv: 2307.02054 [cs.CL]. URL: https://arxiv.org/abs/2307.02054.

Scheffler, Tatjana and Ivan Nenchev (2024). “Affective, semantic, frequency, and descriptive norms for 107 face emojis”. In: *Behavior Research Methods* 56.8, pp. 8159–8180.

Scholkopf, Bernhard and Alexander J. Smola (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press. ISBN: 0262194759.

Semary, Noura A. et al. (2023). “Improving sentiment classification using a RoBERTa-based hybrid model”. In: *Frontiers in Human Neuroscience* 17. ISSN: 1662-5161. DOI: 10.3389/fnhum.2023.1292010. URL: https://www.frontiersin.org/journals/human-neuroscience/articles/10.3389/fnhum.2023.1292010.

Shardlow, Matthew, Luciano Gerber, and Raheel Nawaz (2022). “One emoji, many meanings: A corpus for the prediction and disambiguation of emoji sense”. In: *Expert Syst. Appl.* 198, p. 116862. URL: https://doi.org/10.1016/j.eswa.2022.116862.

Singh, Gopendra et al. (Jan. 2022). “Unity in Diversity: Multilabel Emoji Identification in Tweets”. In: *IEEE Transactions on Computational Social Systems* PP, pp. 1–10. DOI: 10.1109/TCSS.2022.3162865.

Sy, Christian Y. et al. (2024). “Beyond BERT: Exploring the Efficacy of RoBERTa and ALBERT in Supervised Multiclass Text Classification”. In: *International Journal of Advanced Computer Science and Applications* 15.3. DOI: 10.14569/IJACSA.2024.0150323. URL: http://dx.doi.org/10.14569/IJACSA.2024.0150323.

Tomihira, Toshiki et al. (2020). “Multilingual emoji prediction using BERT for sentiment analysis”. In: *Int. J. Web Inf. Syst.* 16, pp. 265–280. URL: https://api.semanticscholar.org/CorpusID:224947877.

Yang, Qiaoling et al. (2023). *SAMN: A Sample Attention Memory Network Combining SVM and NN in One Architecture*. arXiv: 2309.13930 [cs.LG]. URL: https://arxiv.org/abs/2309.13930.

Zampieri, Marcos et al. (Dec. 2020). “SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020)”. In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Ed. by Aurelie Herbelot et al. Barcelona (online): International Committee for Computational Linguistics, pp. 1425–1447. DOI: 10.18653/v1/2020.semeval-1.188. URL: https://aclanthology.org/2020.semeval-1.188/.