University of Sheffield

Contextual Emotion Detection in Text

Nina Kasimova

Supervisor: Nikolaos Aletras

A report submitted in fulfilment of the requirements
for the degree of BSc in Artificial Intelligence and Computer Science

in the

Department of Computer Science

May 10, 2023

Declaration

All sentences or passages quoted in this report from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this project and the degree examination as a whole.

Name: Nina Kasimova

Signature: Nina Kasimova

Date: 09.05.2023

Abstract

This report explores the task of emotion detection from text in the absence of any other clues such as voice changes or face expressions using natural language processing techniques. It has a wide range of applications in communication with people in written format. The method is based on the assumption that the language people use is affected by their current emotional state. Thus it is essential to understand the basics of psychology behind the emotion theories to develop an effective model. The main aspects covered include the choice of classification methods followed by applying them to build a set of emotion detection models using machine learning. Finally, the results produced by various techniques are compared and discussed to conclude which one is more effective for the provided dataset that includes four emotion classes. The results show that the most effective approach is to employ deep learning, specifically LSTM models, however the advantages of different algorithm are provided as well.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Emotion detection is a part of natural language processing attempting to detect and identify emotions expressed in a text. It has a wide range of applications in any area that requires human computer interaction by helping to understand or predict how people will behave in different situations. Examples include customer service, technical support, healthcare, mental health counselling, robotics and chatbots.
Emotions play a huge role in communication between people. This is how we respond to social clues or infer additional information from other people's behaviour. Subtle changes in the choice of words can affect our mood and opinions. All these factors together contribute to the way we perceive the experience.

Since it's such an important issue, further development of emotion awareness in machines could significantly improve the quality of communication. Companies use it to efficiently analyse reviews and automatically identify weak or strong points about their product, thus improving customer relationship management. In healthcare, the output of a chatbot can be adjusted according to the emotional state of the person that was deducted from their previous responses, making them feel more comfortable and offering better help.

The process of emotion detection is similar to the way a human would think about it. A model will use a set of features, such as words and relationships between them, that are typically linked to a particular emotion category. It is then possible to predict if the speaker is experiencing that particular emotion by analysing the presence and frequency of these words in a sentence. Since other clues such as face expressions or voice tone are not available, this method is based on the premise that people will use different language patterns when affected by an emotion and a model will be able to infer their current emotional state based on that change.

The accuracy of the current state of the art emotion detection systems reaches above 95%. How well it performs depends on several factors: quality and size of the training dataset, efficiency of the feature extraction method applied and the classification approach. Therefore, many researchers aim to find the combination of these steps that results in a higher performance.

The theoretical limitations of emotion detection are similar to what people experience,

for example identifying just one emotion from an ambiguous statement, failing to understand sarcasm or a metaphor. Furthermore, people express their emotions differently depending on their background, culture or personal circumstances . This problem becomes especially noticeable when using a complex emotions model with more classes and less obvious boundaries. In order to build an efficient emotion detection system, it is essential to be aware of the psychology behind it to choose a suitable emotion model, eliminate errors and tune the model to fit a particular task.

Chapter 2

Background

2.1 Limitations of identifying emotions

The main difficulties during this task arise from the complexity of emotions and their ambiguity. Different people express emotions in different ways making them highly subjective. Expressions vary between cultures, ages, backgrounds and personal circumstances (Bazanelle 2004). Similar phrases can appear in a sarcastic or metaphorical way or be used to show a number of emotions depending on the context, for example, the phrases "They were laughing" and "They were laughing at someone" will have opposite implications, although the word that carries the most semantic meaning in both sentences is the same.
In its simplest form, emotion detection from text is reduced to finding relations between the words or symbols and the emotional state of the author that prompted them to choose those words (Kao et. al., 2014).

2.2 Psychological models of emotions

A broad definition applicable to the way emotions are viewed in sentiment analysis is "states that reflect evaluative judgments (appraisal) of the environment, the self and other social agents, in light of the organisms goals and beliefs, which motivate and coordinate adaptive behavior" (Hudlicka, 2011). There is no standard system of classifying emotions due to the multiple levels of complexity of human minds. However, a number of models accepted in psychology are widely applied in sentiment analysis.

Emotions in psychology can be divided into basic and complex, "i.e. emotions that are hard to classify under a single term such as guilt, pride, shame, etc." (Tabari, Zadrozny, 2018). In the context of sentiment analysis, the majority of past works refer to basic emotions: sadness, happiness, fear, anger, surprise and disgust.

The two main types of theories of emotion classification are categorical and dimensional. They differ by how they define emotions, their approach to categorising them and the number of states defined. The complexity varies from using four basic emotions to classifying them into highly specific states.

2.2.1 Discreet model

According to the discrete model, also called categorical or basic emotions theory, all emotional experience can then be defined in terms of various mixtures or blends of a finite set of primary or basic emotions (Allen et al, 1988), where each "state is marked by its neurobiological activity and expression pattern like distinct facial expressions" (Izard, 1977). Paul Ekman and Carol Izard argued that similar expressions of at least six emotions, otherwise known as core emotions, are recognised across different countries and cultures. A popular example is Ekman's model of six basic emotions: fear, anger, joy, sadness, disgust and surprise (Ekman, 1992).

This definition makes the discrete model easier to use in the emotion detection task since it is possible to classify the sentiment into separate classes making theories like Ekman's or Plutchik's the most common in NLP. Although Plutchik's theory combines basic emotions and dimensions by illustrating pairs of emotions against each other, such as joy versus sadness.

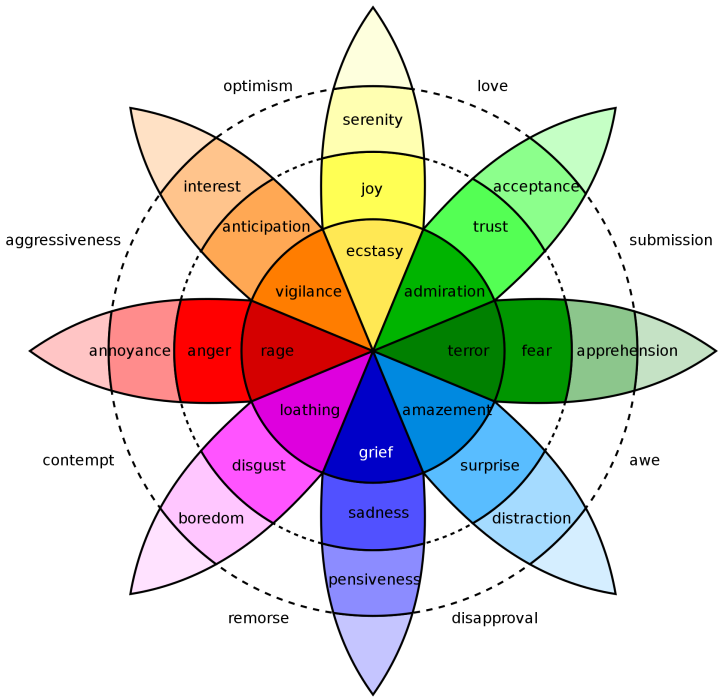

Figure 2.1: Wheel of emotions (Robert Plutchik, 1980)

2.2.2 Dimensional model

The other type of models is based on the premise that emotions are composed of varying degrees of two or more dimensions, for example, pleasantness and intensity in Russell's circumplex model (Barret, Russel, 1998). " Each emotion can be understood as a linear

combination of these two dimensions" (Russell, 1980). The dimensions are valence and arousal or their synonyms. Valence degree is the polarity of a sentiment, i.e. how positive or negative the emotion is, for example joy vs sadness. Arousal refers to the intensity or physiological stimulation of the experience. Examples of emotions on opposite sides of the spectrum would be fear and contentment. In dimensional model emotions are not independent of each other, two emotions that are both far away from the centre are less likely to be experienced at the same time, as shown in figure 2 below.
The main difference that comes with using this emotion theory in NLP is that it changes the structure of a dataset. Each sample needs to be labelled with a continuous score along the two dimensions instead of an emotion category.

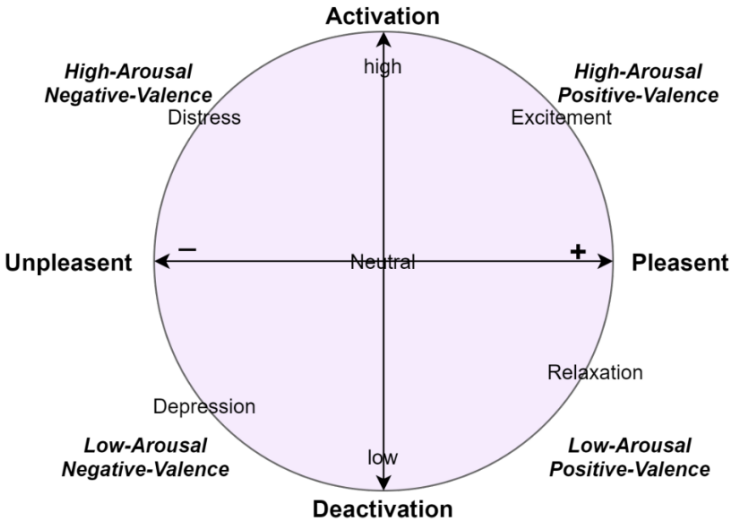

Figure 2.2: Circumplex model of emotions (Zad et al., 2021)

2.3 The process of sentiment analysis

Various techniques of sentiment analysis are broadly divided into lexicon based, machine learning and deep learning approaches. The steps involved in the latter two are described in Figure 2.3. Each stage might be different depending on the nature of the problem and its final purpose, desired solution and available datasets.

2.3.1 Lexicon based approach

Lexicons are collections of nouns, verbs, adjectives and adverbs annotated with an emotion they evoke and assigned a value. The mean of the values is then used to compute the sentiment of a sentence (Pansy and Rupali, 2021).
High coverage and high precision emotions lexicons are difficult to create due to the costs and

time limits since it requires human annotators.Examples of such term-emotion lexicon those created by Mohammad and Turney (2013) called EmoLex that was larger than WordNet Affect (Strapparava and Valitutti, 2004) and included most frequent English unigrams and bigrams annotated according to Plutchik's model.
One of the flaws of this method is that it doesn't take the context into account, so the sentiment of the word will be the same in all the sentences. However, even with a good corpus, low recall of a model becomes a problem, although it produces high precision results. (Chatterjee et. al, 2019) Implicit emotions are more difficult to identify with a lexicon if there is no emotional word in the sentence. Expressions that are used in a metaphorical or sarcastic way will be missed or misclassified by a lexicon.

2.3.2 Machine learning approach

Currently best performing systems for emotion classification are using supervised machine learning (Seyeditabari, Armin, Tabari, 2018). The systems are created by training a model on a large dataset of text samples and their corresponding labels where each one is describing an emotion category. The model then learns patterns associated with an emotion class and makes predictions for unseen data based on computed probabilities. Figure 2.3 briefly summarises the process. Algorithms such as Naive Bayes and SVM typically achieve higher accuracy than lexicon based approaches. Hasan et al (2014) achieved results close to 90% using KNN, SVM, Naive Bayes and Decision Tree and categorising the data into four classes of the circumplex model. However it largely depends on the quality of the dataset used for training the model (Mumtaz, Ahuja,2009). An annotated dataset can be expensive and time consuming to obtain. It is typically done by creating a set of possible emotion categories depending on the model of choice and then human judges assign the most appropriate class to each sample.

There is a way to create a dataset automatically using unsupervised learning, however the labels will be less accurate compared to human annotators (Varghese, 2013). A hybrid approach was implemented by (Chatterjee, 2019) where an initial small set of dialogues was annotated by human judges. Then a candidate set was created by producing word embeddings of the original dialogues and finding similar samples in the entire pool. The potential samples are shown to human judges to confirm if it is labelled correctly. This approach reduced the number of judges required to create a labelled dataset. An example of a widely used dataset is the SemEval (Semantic Evaluation) (Strapparava and Mihalcea, 2007) containing new headlines annotated with six Ekman's emotions.

In recent years research has shown that neural networks produce more accurate results as opposed to traditional machine learning techniques. Models such as convolutional or recurrent neural networks are able to capture complex relationships between words meaning they will infer information from context as well. The difference in performance becomes noticeable in texts where the emotion is ambiguous or subtle. Their only disadvantage is that these models tend to be computationally expensive and require more training data.

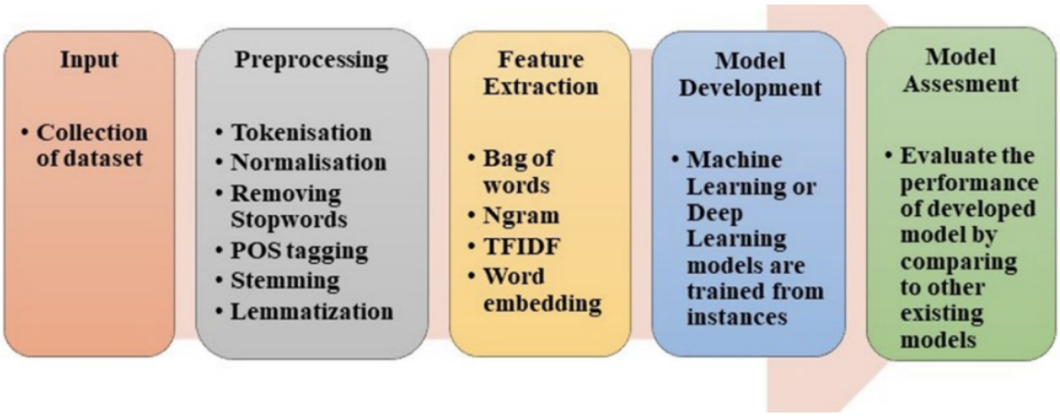

Figure 2.3: Steps of sentiment analysis using machine learning (Nandwani, Verma,2021)

2.3.3 Preprocessing

Tokenization

This step breaks down a collection of text documents into a list of words and symbols called tokens. For example, "a collection of documents" will become ['a', 'collection', 'of ', 'documents'].

Normalisation

Normalisation is done to make a text more uniform and consistent. It involves converting all the words into either lower or upper case so that the words 'happy' and 'Happy' are viewed to be the same by a word embedding model. In the context of text messaging, it can also include substituting abbreviations like 'u' with 'you'.

Stemming

The stemming is the process of bringing a word to its base form or root by removing any suffixes, such as '-ing', '-or' and '-ed'.

Stop words removal

Stop words are common words used in every text that do not carry any semantic meaning and are therefore not useful for s. Examples are articles, prepositions, auxiliary verbs etc. As a part of a dimensionality reduction strategy, they can be safely removed from the training data. There are a number of libraries that provide a standard list of stopwords that apply for any task, however it can be adjusted to fit a specific purpose.

2.3.4 Word Embedding

The results of a certain model will depend on the effectiveness of the features selected. This is broadly referred to as feature extraction, or word embedding in case of natural language processing. Since the algorithms do not actually understand plain words, there is a series of steps that needs to be taken before classification to convert the text into a different format that is comprehensible by a machine learning algorithm. However, the features used by the machine learning algorithm don't have to be just a different representation of text. Balabantaray et al. (2012) created a model using SVM classifier with 11 features including unigrams, Word-net Affect lexicon, parts of speech, emoticons and more. The emotions were classified according to Ekman's model with an accuracy of 73.24

Bag of Words

One of the simplest techniques to do that is 'Bag of Words' (BOW) where a unique vocabulary is created for a text corpus by tokenizing it into individual words. Then each sample is represented by a vector of word frequencies in it. The intuition behind it is that most meaning of a text is contained within the words rather than their order and sentences close in meaning will have similar vectors representing them.

This approach is fairly straightforward to implement however BoW ignores the order of words in a sentence as well as grammatical structure. It also treats all the words equally and uses only frequency without considering how much information each word carries which leads to a number of drawbacks such as failing to capture the order and meaning of words in a sentence as well as technical issues of dealing with a sparse matrix. The issue of ignoring the meaning of sentences is addressed by TF-IDF described in the section. This method takes into account the frequency of words in a sample as well as in the whole set of documents thus providing a score that reflects the importance of a particular word in the dataset.

Tf-IDF

An approach similar to BOW is term frequency-inverse document frequency (TF-IDF). It is based on the idea that a rare term will carry more information about the topic or the sentiment of a sample (Liu et al. 2019).
The calculation for each term t combines two variables: term frequency and inverse document frequency. Term frequency is the number of times a word appears in a document over the total number of words in that document tf(t, d). In order to counteract the fact that the term might just be a common word in general, for example, articles or prepositions, inverse document frequency (IDF) is added to the equation. IDF is computed by taking a log of the total number of documents |D|divided by the number of documents containing the term, df(t).

$$idf(t, D) = log(\frac{|D|}{df(t)}) \tag{2.1}$$

$$tf.idf(t, d, D) = tf(t, d) \cdot idf(t, D) \tag{2.2}$$

Word2Vec

Some of the more complicated methods include pre-trained models such as latent semantic analysis, Word2Vec (Mikolov et al., 2013), GloVe (JeffreyPennington et al., 2014), Google's Bert. They work by using statistics about a words' co-occurrences and hence their closeness to other words, eg with Word2Vec, "the result of a vector calculation vec("Madrid") - vec("Spain") + vec("France") is closer to vec("Paris") than to any other word vector" (Mikolov et al., 2013). There are two techniques in Word2Vec: CBOW (continuous bag of words) and Skip-gram, both of which are two-layer neural networks.

The Skip-gram model tries to predict the probability of surrounding words given one word, or its context within a specific 'window size'. The input layer is a one-hot encoded vector of size N, where 1 represents the word of interest and 0's for the rest of the vocabulary. In the output layer, each of N neurons is a probability of how likely the corresponding term is to appear next to the original word in a text. Hence, it forms a vector that represents the context or semantic meaning for each term in the vocabulary in relation to the specific training text. This model is useful for semantic analysis because if two words are found to have similar vectors, it means they are close in their meaning.
CBOW works in the opposite way. Given surrounding words this model will predict the word in the middle.
An advantage of using Word2Vec for the word embedding is that it reduces the number of features needed for learning models compared to TF-IDF or BOW and more importantly, it captures the semantic meaning of the terms rather than relying on their frequency.

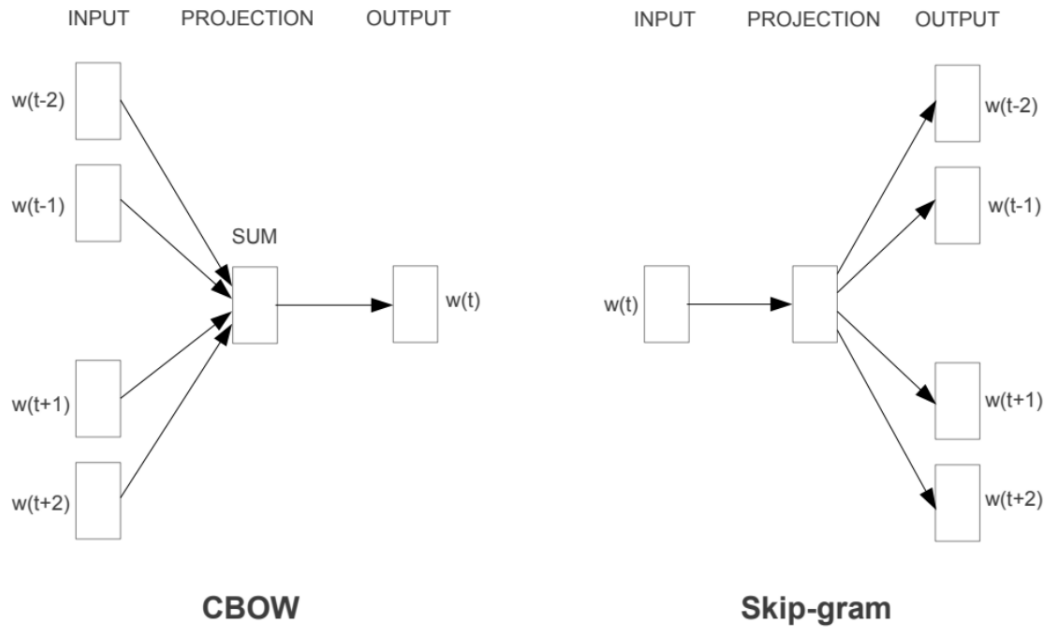

Figure 2.4: Skip gram (Mikolov et al, 2013)

2.4 Review of machine learning approach

In relation to contextual emotion detection in text specifically as opposed to semantic analysis, The work "Review of Contextual Emotion Detection in Text paper" by (Chatterjee et al., 2019) is a good example of related works and popular solutions because it describes multiple top models picked from 311 submissions all using machine learning.
The task given to teams contained a training dataset consisting of 30160 dialogues that have undergone initial preprocessing. The data was filtered to remove any offensive or sensitive content and personally identifiable information using keywords and human judges. Each sample had three utterances and was classified as one of the categories: 'happy', 'angry', 'sad' or 'other.
The baseline model has minimal preprocessing strategies including removing extra punctuation and spaces and making all the text lower case. The word embeddings were produced using Glove embeddings which are passed to an LSTM layer and then mapped to a 4 dimensional vector that represents class probabilities. This model achieved a starting point of 0.5861 f1 score.

Key observations of the models designed by the best performing teams:

- Using a neural network architecture for higher accuracy
- Transfer learning using BERT, ELMo or GloVe
- Attention mechanisms
- Combination of techniques to improve predictions

For word embeddings, the most common method was transfer learning using BERT, ELMo, and GloVe being the most popular. BERT (Bidirectional Encoder Representations from Transformers) is a neural network architecture. It performs well in this task because it takes into account the order of words in a sentence and more importantly creates a vector based on the context of a word or its contextual embedding. Since it's bidirectional the model will consider the text on both sides of the word. The input to the model has to be a whole sentence so that if there are two identical words with different meanings, BERT will create two separate vectors unlike Word2Vec. Although BERT is a pre-trained model, it can be fine-tuned to fit a specific task by adding a classification layer in this case.

Most approaches used a combination of techniques to improve predictions via addressing specific problems such as class distribution differences or separating emotion classes from others. One of the teams combined a semantic and emotional meaning of words via GloVe, ELMo and Deepmoji and later applied contextual LSTM to describe the whole sentence, thus using both semantic and context information of the sample. In another example a team was able to better distinguish between emotion classes and 'other' by combining BERT and USE models. An attention mechanism was applied by the majority of teams. This is a technique introduced by Bahdanau et al (2015) to improve the performance of deep learning algorithms. Parts of the input are assigned a certain weight instead of the whole sequence being equal.

This allows the model to capture the context by focusing on relevant information and leads to more accurate predictions.

Chapter 3

Model Design

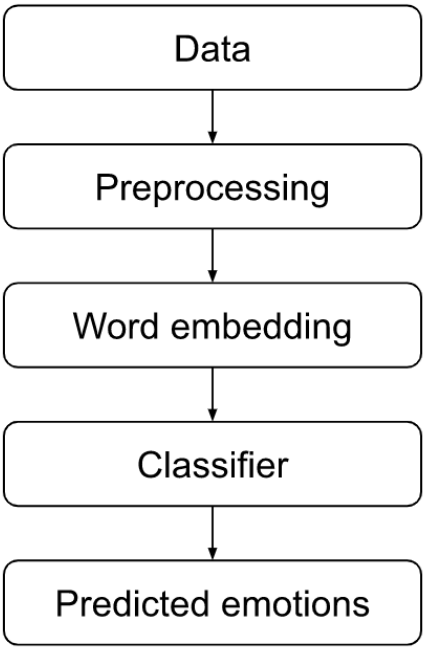

Figure 3.1: Model design overview

3.1 Dataset

The training data linked in the references is a publicly available dataset that consists of 30k records of labelled casual text messages. Each one of them includes three turns of short utterances and a label describing one of the conveyed emotions: “happy”, “sad”, ”angry” or “others”. Half of the dataset is labelled as class “others” containing instances that either did not contain any clear emotion or were ambiguous. Figure 3.2 is an example of several

samples:

Turn 1	Turn 2	Turn 3	Label
ok i will	THANK YOU	wlcm	others
Ntng….	anyways good morning…how's life at your end	Haha…… this is evening	happy
You are fool	So I've been told.	You are dumb	angry

Figure 3.2: Samples of the data

The main challenge when trying to classify this text is that it uses a lot of 'texting language' with a lot of slang words and abbreviations such as "u r" instead of "you are", "tmw" for "tomorrow" etc. This creates a problem for the word embedding algorithm since those words will be treated as different from each other. The data also contains a large number of emojis that are a helpful cue to disambiguate the sentence. However, it creates a limitation on the choice of word embeddings that do not recognise them. The impact of emojis is visible when comparing the results of TF-IDF and Word2vec methods. The first utterance in the following dialogue in Figure 3.3 is an example of how an emotion is conveyed solely through the laughing emoji.

Turn 1	Turn 2	Turn 3	Label
😂😂😂 it happens sometimes	it happens to me all the time	haha.... leave it	Happy

Figure 3.3: Effect of emojis on the emotion category of a text

Some of the dialogues are quite ambiguous and might have been mislabelled as well as shown in Figure 3.4. For example, a sentence which does not really have any emotions in it is labelled as happy but because it does not contain any words associated with this emotion it was predicted to be in the 'others' category by a logistic regression model. The opposite scenario occurs in several cases as well.

Turn 1	Turn 2	Turn 3	Label
Send me ur voice clip	I will send. I can try singing in Lata Didi's voice.	Chalega? Wow	Happy

Figure 3.4: An example of an ambiguous dialogue

Another common issue is an overlap between 'sad' and 'angry' because certain words can indicate either of those or even both. In the example in figure 3.5 the words 'sorry please' are most likely to appear in a 'sad' category and there is no clear indication of anger, however this line is labelled as 'angry'. Confusion matrices show the highest misclassification rate between these two classes as well.

Turn 1	Turn 2	Turn 3	Label
Sorry please	you're sending me all kinds of mixed messages	Do not act like you don't know me	angry

Figure 3.5: An example of mixed emotions

3.1.1 Classes distribution

One of the factors that affect a model's performance and the choice of evaluation methods is how evenly distributed the data classes are. This dataset consists of 30,160 instances with the "others" category being 50% of it as shown in figure 3.6. The imbalance is addressed in the model design section. There are two options for each classifier: with only "happy", "sad", "angry" and the second one is using the full dataset to compare how "others" category affects the performance of each model. The rest of the classes are almost even or at least to a point where it does not have a significant impact on the performance.

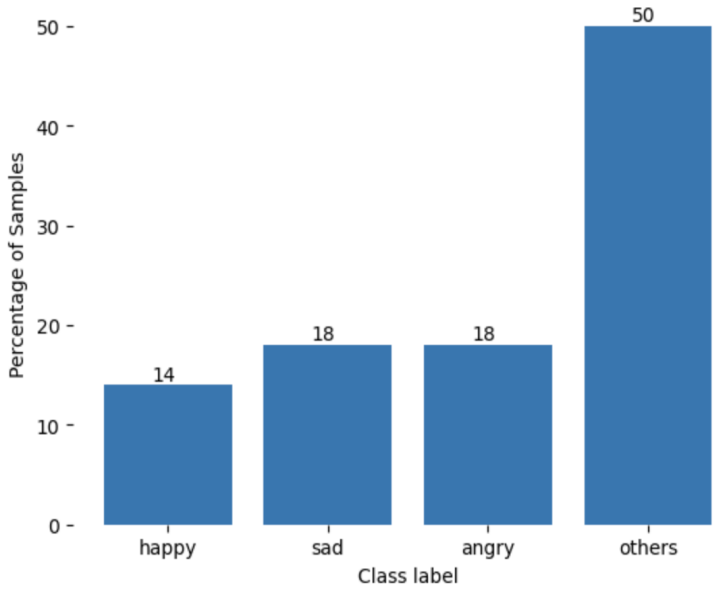

Figure 3.6: column chart to show the distribution of classes in the dataset

Chapter 4

Implementation

The language of choice for these experiments was python due its simple syntax, readability and a large variety of libraries for data analysis.

Scikit-learn

Although originally the first models were written from scratch, scikit-learn allows for a faster and more efficient implementation of linear ML classification models as well as evaluation methods. It makes it less likely for the results to be affected by a mistake in the implementation, makes the code more readable and easier to adjust.

Pandas and Matplotlib

Pandas library is commonly used for data manipulation. The raw text data is stored in the form of tables making it straightforward to read, filter and find certain sentences according to the emotion labels, for example. Matplotlib is a library used to create various plots and helps visualise the data or results received from classifiers.

Keras

One of the models implemented in the experiments is LSTM which is a neural network that can be built using Keras. It is a high-level application programming interface (API) that helps build and train learning models, tune hyperparameters, and provides easy to understand error messages, all of which resulting in cleaner code and ability to experiment with different setups.

4.0.1 Preprocessing

The goal of this step is to prepare the data to be used in a machine learning model. It includes cleaning the text to improve its quality, finding an efficient way to reduce dimensionality without losing any useful features, restructuring it for easier analysis and vectorising text data to get a numerical representation. The provided data required minimum preprocessing. The dialogues are taken from online conversations and thus they already contain the minimum

information. Words are commonly shortened so performing stemming or lemmatization decreases precision. A way to further clean text data is using regex to remove any non language symbols like punctuation that are usually not necessary for identifying semantics. However, in this case, punctuation symbols are used to write emojis, which appear a lot in messages and are a good indicator of the implied emotion, e.g :) is often seen in sentences labelled "happy".

Stopwords removal is generally considered a good way to reduce the amount of features by removing words that carry little to no meaning on their own and therefore, do not contribute to the overall semantic meaning of the sentence. Some words like "no" or "but" are also filtered out and this becomes a problem because they completely change what the author is trying to say, leading to the loss of information or incorrect interpretation. According to experimental results, removing stop words by using the list from sklearn library actually slightly reduces the accuracy of the models and manually creating a list has no significant impact on either the speed or performance. Finally, the main word embedding methods I used were TF-IDF and a skip-gram Word2Vec the results of which are described in the next chapter.
Two options of Word2Vec were implemented: one model was trained on the provided data and the other one was a downloaded ready model pre-trained by Google on a news dataset. The performance of the latter one was noticeably better so for the sake of comparing word embedding methods, the pretrained model was used.

4.0.2 Classifiers

For the classifier I chose the models that are most commonly used for this task in other works and show higher performance.

Naive Bayes

Naive Bayes is a supervised machine learning algorithm that is straightforward and fast to implement. It is based on Bayes theorem that finds the most likely emotion class given a sentence when prior information is available.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{4.1}$$

This formula's result is the probability of a hypothesis A given B is true or probability of a sentence belonging to an emotion class: P(class|sample), where sample is a set of features. P(B|A) is the likelihood of observing a set of words from the given sample in class A. It is the frequency of each word found in text labelled as sentiment A divided by the number of features with the same label. Since words are assumed to be independent of each other, the final value is going to be the product of the likelihoods of each word in the sample.
P(A) is the prior probability of a class that is calculated based on the distribution of labels in the dataset.
P(B) is the probability of observing a set of words regardless of their label.

Using all this data the posterior probability P(A|B) is computed for each class and the class with the highest probability will be the predicted label.
The assumption of Naive Bayes that the features (words in a sample) are independent of each other is not true in the context of sentiment analysis and can have a negative impact on accuracy.
Naive Bayes can be affected by an imbalanced dataset and skew towards the majority class, however the dataset is evenly divided between "sad", "happy" and "angry" categories.

Logistic regression

The input features are combined using weights and biases and are then passed to the sigmoid function that maps them between values 0 and 1 that represent the predicted probability. A threshold value is used to determine what class a sample belongs to, for example, if the probability is more than 0.5, it is the positive class and otherwise, the instance is classified as belonging to the negative class. The parameters of the model are learnt by attempting to minimise the difference between predicted class labels and the true labels.

Logistic regression is most commonly used for binary classification problems. One way of modifying the algorithm to fit multiple classes is splitting the multinomial problem into several binary classification problems and using the standard variant of logistic regression. Another method involves changing the loss function and the number of output values to match the amount of classes.

LSTM

The LSTM (long short-term memory) model is widely used in natural language processing because it is designed to work well with sequential data. Its main benefit is the ability to identify the important information that needs to be stored in the memory and unnecessary information that can be removed. The architecture consists of multiple memory cells connected to each other in a sequence. The information flow in and out of the cell is controlled by three gates: input, forget and output. Input gate controls what information should be stored in memory. It uses current input and previous states to update the cell state. Forget gate determines what information should be kept and what should be discarded from the memory. Finally, the output gate decides what the next state is going to be.
Due to its structure LSTM is able to retain information from previous parts of a sentence for a long time and use it with current input to make more accurate predictions. This feature makes LSTM better than traditional machine learning techniques at identifying relationships between words and capturing the context which is important for understanding the emotional content of a sentence.

Despite their effectiveness, LSTM models have several downsides. They are more computationally expensive to train, especially when using large datasets and several hidden layers. The accuracy of LSTM depends on hyperparameters which are also time and resource consuming to tune well. Their lack of interpretability makes it difficult to understand how the model

arrived at the predictions, which can be essential in emotion detection for identifying what words are important and affect the predictions the most. Having that information could help adjust the data and tune the model to improve its performance as well as increasing confidence in the accuracy of results if any business decisions rely on the predictions of that model.

Chapter 5

Results and Discussion

5.1 Evaluation Techniques

Evaluation metrics are used to assess the performance of a classifier. The choice depends on the purpose of a task, input data, class distribution and other variables. Most common values used in past works on text classification are accuracy, f1 score, precision and recall. The confusion matrix is slightly different for multiple classes opposed to binary classification. Figure 5.1 helps to visualise how the evaluation metrics are derived and why they matter for measuring the performance.

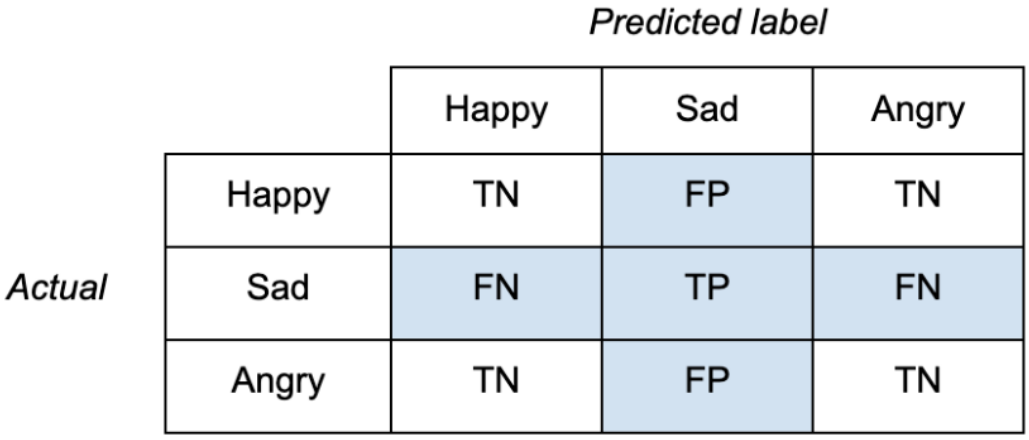

Figure 5.1: Confusion matrix for any one class in a multinomial classification, TP - true positives, FN - false negatives, FP - false positives, TN - true negative

Accuracy is the proportion of correctly classified samples out of all predictions made.

$$accuracy = \frac{TP + TN}{allpredictions} \tag{5.1}$$

Recall is the number of correctly classified instances belonging to a class divided by the

true number of instances in that class. It is useful to assess the model's ability to identify a certain emotion in a sentence.

$$recall = \frac{TP}{TP + FN} \tag{5.2}$$

Precision is the proportion of correctly predicted labels of a class out of all predicted labels of that class. It is useful in problems where it is important to reduce the number of false positives.

$$precision = \frac{TP}{TP + FP} \tag{5.3}$$

Finally, F1 score is a harmonic mean of recall and precision.

$$accuracy = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{5.4}$$

Macro F1 score is calculated by averaging the values of all classes and equally weighing them. On the other hand, weighted f1 score takes into account the number of samples in each class and can be used on imbalanced datasets. This can be seen when training a model on the original dataset without excluding the "others" label. For example, for a logistic regression model without "others" macro and weighted f1 scores are both 86%, while including "others" results in macro being 75% and weighted 78%.

F1 score as a performance measure is generally slightly better than accuracy to account for a different number of samples in each class. There are less samples labelled "happy" in the dataset as shown on Plot X in the model design section, f1 scores and accuracy are the same for all models meaning this imbalance has little effect on the performance when training models on only three classes: "happy", "sad", "angry".
When using the full dataset with "others" class in it, however, it creates a visible difference between accuracy and f1 score because the high proportion of correctly predicted instances of "others" class increases the overall accuracy.

Classifier	Accuracy	F1 score
Naive Bayes	0.72	0.65
Logistic regression	0.78	0.75
LSTM	0.85	0.84

Table 5.1: Compare accuracy and f1 scores when including "others" label

5.2 Discussion of results

The best result of 91% f1 score is produced by the LSTM model, using TF-IDF embedding and not removing stopwords from data.

Note

Uneven datasets tend to have an impact on the performance of classifying algorithms and it can be different depending on what technique is used. To address this problem there are two options of the dataset: the full one with 4 classes and one with only "happy", "sad" and "angry" classes that does not take "others" category into account. This issue is addressed further in the last part of this section. The tables summarising the results of preprocessing options and word embedding methods show the score received by the best conditions possible, i.e using the dataset with only three classes.

5.2.1 Preprocessing

The first and only preprocessing step in the model is tokenization. The result of it is replacing sentences with a list of individual words that are further used to create word embeddings. Stemming or lemmatization was unnecessary since most utterances contain little text that is already abbreviated and shortened. Introducing this step did not affect results in any way apart from decreasing the processing speed.

The point of removing stopwords is to reduce the amount of features used for training the model and increase its speed as well. It might improve the accuracy by decreasing the amount of noise and leaving only the important features. This process removes words that are not supposed to have any effect on the meaning of a text such as "the", "or", "a" etc. However, in the case of this model it has a slightly negative effect on the f1 score of every classifier as shown in table 7 below.

Classifier	No stopwords	Including stopwords
Naive Bayes	0.80	0.82
Logistic regression	0.83	0.86
LSTM	0.89	0.91

Table 5.2: F1 scores of classifiers to compare stopword removal step

5.2.2 Word embeddings choice

The highest f1 scores across all classifiers were received when using TF-IDF word embedding. Word2Vec tends to perform better on text classification since it also captures the context of a word. However in this case the dataset has many casual messages that contain a lot of emojis which are a good indicator of the intended emotion, and not much meaningful information otherwise. Word2Vec is trained on words only and would require a similar model designed for emojis to work on this data.

Classifier	TF-IDF	Word2Vec
Naive Bayes	0.82	0.64
Logistic regression	0.86	0.80

Table 5.3: F1 scores to compare performances of word embedding techniques

5.2.3 Classifiers comparison

The table shows the best results received by all the classifiers when using the combination of the highest performing word embedding and preprocessing methods and training on the whole dataset, including “others” class and then on the same dataset but without taking classes labelled ”others” into account. These results demonstrate the effects of an imbalanced dataset on the performance of each classifier.

Classifier	Dataset with ”others”	Dataset without ”others”
Naive Bayes	0.82	0.65
Logistic regression	0.86	0.75
LSTM	0.91	0.83

Table 5.4: Highest f1 scores achieved by the classifiers for the two options of the dataset

5.2.4 Recall in classifiers

As described in the evaluation metrics section recall is the proportion of true positives out of all instances of a certain emotion class in the dataset. High recall reflects a model's ability to correctly detect the most amount of samples containing a particular emotion from the given test set. This is important because emotions can be ambiguous, subtle or expressed in different ways making it difficult to detect them. Improving recall helps identify more emotions and receive a better insight into an opinion.

Although recall is factored in the f1 score, it should be compared separately, especially if it is important for the model to correctly identify all the emotion classes equally. For example, if the purpose of a task is to detect any negative emotion in user reviews for a product and the model suffers from low recall for “angry” or “sad” categories, it would have lower chances of picking up on slight frustration or disappointment expressed by a customer. This issue could potentially lead to missing out on important information about what could be improved for the benefit of the company. However, if the recall for other classes is much higher, f1 score and accuracy values still average out to be acceptable.

Recall is affected by a number of factors such as complexity of an emotion class, the amount of training data, class distribution in the dataset and the algorithms used for classification. Including the “others” label creates an imbalanced dataset. The impact of it is noticeable with traditional machine learning models because they are more likely to be biassed towards the majority class. Logistic regression is able to adjust the decision boundary for an uneven class distribution so it is less sensitive to an uneven class distribution. While Naive Bayes is

way more affected by the imbalance. Figure 5.2 shows the recall values for labels in Naive Bayes and logistic regression that are used to calculate their f1 scores.

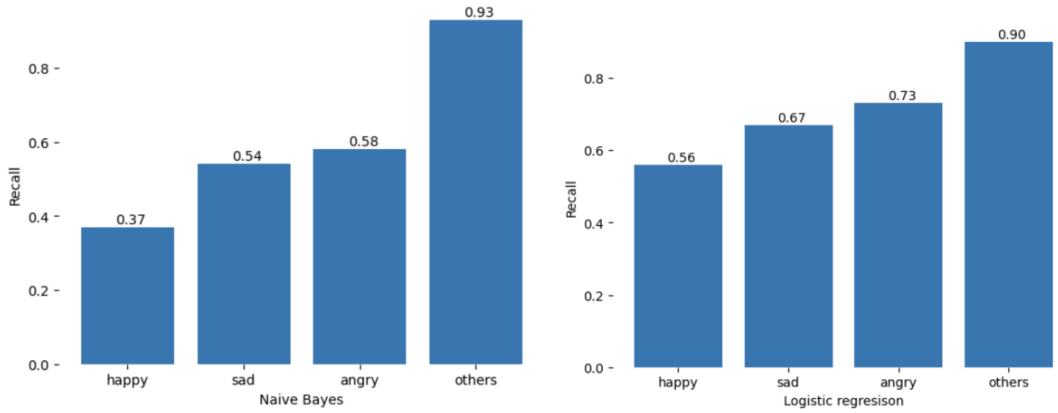

Figure 5.2: compare recall of Naive Bayes (left) and logistic regression (right) with "others" label present in the training dataset

LSTM, however, performs a lot better on an unbalanced dataset due to its structure and can learn from less data. The model is able to identify important features and detect more complicated relationships between words. In scenarios where it is difficult to obtain a perfect dataset where each class has many well labelled samples, LSTM is a better choice for emotion detection compared to traditional machine learning methods.

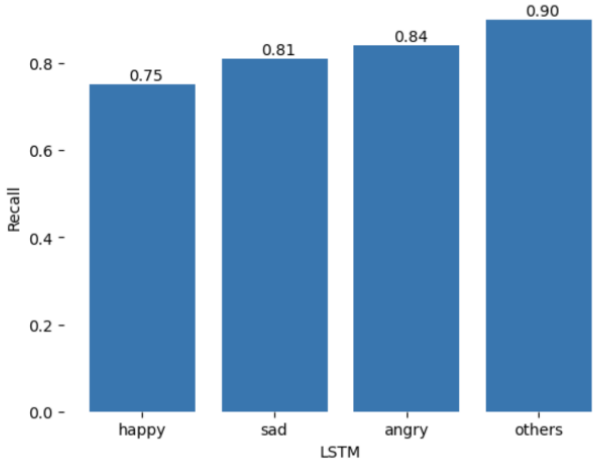

Figure 5.3: recall of each emotion class by LSTM model

Chapter 6

Conclusions

The problem of emotion detection in text greatly benefits a variety of applications that include communication with people in written format. Marketing and advertising benefit from extracting opinions from customer feedback or social media posts. Chatbots that are designed to sound more human-like can adjust their response to be more empathetic based on the identified emotion of the person. In the field of healthcare, especially when considering mental health, emotion detection is able to analyse and track the emotional state of the patients to offer better solutions. As opposed to determining whether a statement is positive or negative, emotion detection involves specifying a particular state. Hence, this paper presented an overview of a basic psychology background understanding of which is essential for developing effective models and being able to identify errors produced in the process. Major aspects covered include the definition of what emotions are, challenges and limitations arising from the way they are conveyed in text, factors contributing to the choice of an emotion model and the importance of a well labelled dataset.

There are several important factors to consider when writing a model such as an appropriate emotion model and dataset suited for the purpose of a task, accurately choosing a feature extraction technique to reduce dimensionality while preserving the meaning of words and their context, choosing the classifying model and finally, identifying what are the most relevant evaluation metrics. Although there are various approaches to automatically classifying pieces of text into emotion categories, with the two most prominent being lexicon and machine learning based ones, the focus of this paper is on the latter one. Machine learning has its limitations and disadvantages, however, it has been shown to be an effective tool for identifying the sentiment conveyed in a text. Past research has explored many models suitable for this task including traditional ones like SVM, Naive Bayes and logistic regression, and a more recent approach of utilising neural networks, for example LSTM. Some advantages of this method are its ability to process large amounts of data and learning from provided features instead of having to manually label words and symbols for a lexicon based algorithm. Taking context into account is a crucial element contributing to higher performance of deep learning and neural network based word embedding strategies.

The choice of a feature extraction model directly affects how much information ends up

actually being captured from a sample of text and what features are being used to train an algorithm in the following step. Word embeddings are a popular way of representing the text in a numeric way without the need for manual feature engineering. Frequency based embeddings are a fast and transparent approach where the result is determined by the number of times a word appears in the dataset. They also take into account other symbols that could potentially provide a richer insight into the emotional content of a sample, such as emojis on the example of the provided dataset. Neural network based methods are generally more effective as they contain both semantic and contextual information about words as opposed to frequency based and can be fine tuned for specific requirements. However, they are more computationally expensive and depend on words only unless a specific model for emojis is used at the same time.

This paper explores different combinations of word embeddings and machine learning models to find the most optimal solution for the provided dataset. The highest performing model with f1 score of 91% was built using an LSTM model and TF-IDF word embedding.

Bibliography

[1] Aman, S., and Szpakowicz, S. Identifying expressions of emotion in text. pp. 196–205.

[2] Bandhakavi, A. S., Wiratunga, N., P, D., and Massie, S. Generating a word-emotion lexicon from emotional tweets.

[3] Barrett, L., and Russell, J. Independence and bipolarity in the structure of current affect. *Journal of Personality and Social Psychology 74* (04 1998), 967–984.

[4] Binali, H., wu, C., and Potdar, V. Computational approaches for emotion detection in text. pp. 172 – 177.

[5] CBalabantaray, R., mohd, M., and Sharma, N. Multi-class twitter emotion classification: A new approach. *International Journal of Applied Information Systems 4* (09 2012), 48–53.

[6] Chatterjee, A., Narahari, K., Joshi, M., and Agrawal, P. Semeval-2019 task 3: Emocontext contextual emotion detection in text. pp. 39–48.

[7] Christopher D Manning, Raghavan, P. S. H. *Introduction to information retrieval*. Cambridge University Press, 2019.

[8] Heidari, M., Jones, J., and Uzuner, O. Deep contextualized word embedding for text-based online user profiling to detect social bots on twitter. pp. 480–487.

[9] Hudlicka, E. Guidelines for designing computational models of emotions. *IJSE 2* (01 2011), 26–79.

[10] Kao, E., Chieh Chun, L., Yang, T.-H., Hsieh, C.-T., and Soo, V.-W. Towards text-based emotion detection: A survey and possible improvements. pp. 70 – 74.

[11] Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. *Proceedings of Workshop at ICLR 2013* (01 2013).

[12] Mohammad, S., and Turney, P. Crowdsourcing a word-emotion association lexicon. *Computational Intelligence 29* (08 2013).

[13] Mumtaz, D., and Ahuja, B. *A Lexical and Machine Learning-Based Hybrid System for Sentiment Analysis*. 01 2018, pp. 165–175.

[14] Nandwani, P., and Verma, R. A review on sentiment analysis and emotion detection from text. *Social Network Analysis and Mining 11* (08 2021).

[15] Nandwani P, V. R. A review on sentiment analysis and emotion detection from text.

[16] Paul, E. "an argument for basic emotions." cognition and emotion. 169–200.

[17] Plutchik, R., . C. H. R. Circumplex models of personality and emotions. american psychological association.

[18] Raisa Varghese, J. M. A survey on sentiment analysis and opinion mining.

[19] Russell, J., and Barrett, L. Core affect, prototypical emotional episodes, and other things called emotion: Dissecting the elephant. *Journal of personality and social psychology 76* (06 1999), 805–19.

[20] Seyeditabari, A., Tabari, N., and Zadrozny, W. Emotion detection in text: a review, 06 2018.

[21] Shivhare, S. N., and Khethawat, S. Emotion detection from text. vol. 2.

[22] Strapparava, C., and Valitutti, A. Wordnet-affect: an affective extension of wordnet. *Vol 4. 4* (01 2004).

[23] Zad, S., Heidari, M., Jones, J., and Uzuner, O. Emotion detection of textual data: An interdisciplinary survey. pp. 0255–0261.