International Journal of Innovative Research in Engineering and Management (IJIREM)
ISSN (Online): 2350-0557, Volume 13, Issue 3, June 2026
DOI: https://doi.org/10.55524/ijirem.2026.13.3.2
Article ID IJIRE-1451, Pages 7-18
www.ijirem.org

A Survey of Text-based Emotion Detection: Lexicons, Machine Learning, Deep Models and the Move to Large Language Models

Nidhi Singh

Department of Computer Science & Engineering, Institute of Technology and Management, Lucknow, Uttar Pradesh, India

Correspondence should be addressed to Nidhi Singh; nidhisin525@gmail.com

Received: 27 April 2026; Revised: 12 May 2026; Accepted: 24 May 2026

ABSTRACT- Reading the feeling behind a piece of writing is something humans do without thinking. Teaching a computer to do it has turned out to be much harder. This paper walks through how the problem has been attacked over the last twenty years or so, from the first hand-built word lists in the early 2000s right up to the large language model approaches we see today. We start with the lexicon era, when researchers tagged words with emotions and counted them. Then comes the classical machine learning period. Naive Bayes, Support Vector Machines, Maximum Entropy. These were a real step up but they had limits we will talk about. Deep learning showed up next, first with recurrent networks like LSTMs and then with transformers. BERT and its cousins pushed the accuracy numbers into the eighties. Most recently, models like GPT-4 and Claude have started doing emotion classification with no training data at all, just by being prompted. Alongside the methods we cover the datasets people actually use to measure progress, things like ISEAR, GoEmotions, EmoContext, DailyDialog, and the SemEval task series. We talk about the metrics. And we end with the problems that are still open, including sarcasm, the imbalance between common and rare emotions, code-mixed writing of the kind you see in Indian chat, and the difficulty of evaluating models that change every few months. The aim is to give a student or a working engineer something they can actually use as a starting point, without assuming they have read every paper in the field.

KEYWORDS- Affective Computing, BERT, Claude, Emotion Classification, Emotion Detection, Foundation Models, GoEmotions, Large Language Models, Lexicon Methods, Natural Language Processing, Sentiment Analysis, Survey, Text Classification, Transformers

I. INTRODUCTION

Whatever else writing does, it carries feeling. A short text to a friend, a Slack message to a colleague at 11 PM, a one-star review of a restaurant, a comment under a YouTube video. All of those have an emotional load that the reader picks up almost instantly, usually without realising they are doing it. The trouble is that machines do not do this. They see tokens and characters. The feeling behind the words is something they have to be taught to recognise, and that teaching is what the field of emotion detection has been working on for a long time now.

Why does this matter? A few examples make it clear. Support teams handle hundreds of conversations a day, and telling which customer is genuinely angry versus which one is just asking for a refund decides how the queue should be ordered. Mental health applications flag worrying patterns in writing before someone does something dangerous. Content moderation systems try to spot harassment, which almost always has an emotional shape. Marketing teams want to know how people feel about a product launch, not just whether they bought it. Even conversational AI does better when it can tell the user is getting frustrated.

So, the demand has been there for years. What has actually changed is what we can build. Twenty years ago, all you really had were word lists and basic classifiers. Today you can call a single API and get a fairly good classification back without training anything. This survey is about how we got from one to the other. In below Figure 1 gives a rough timeline of the four main eras the field has gone through.

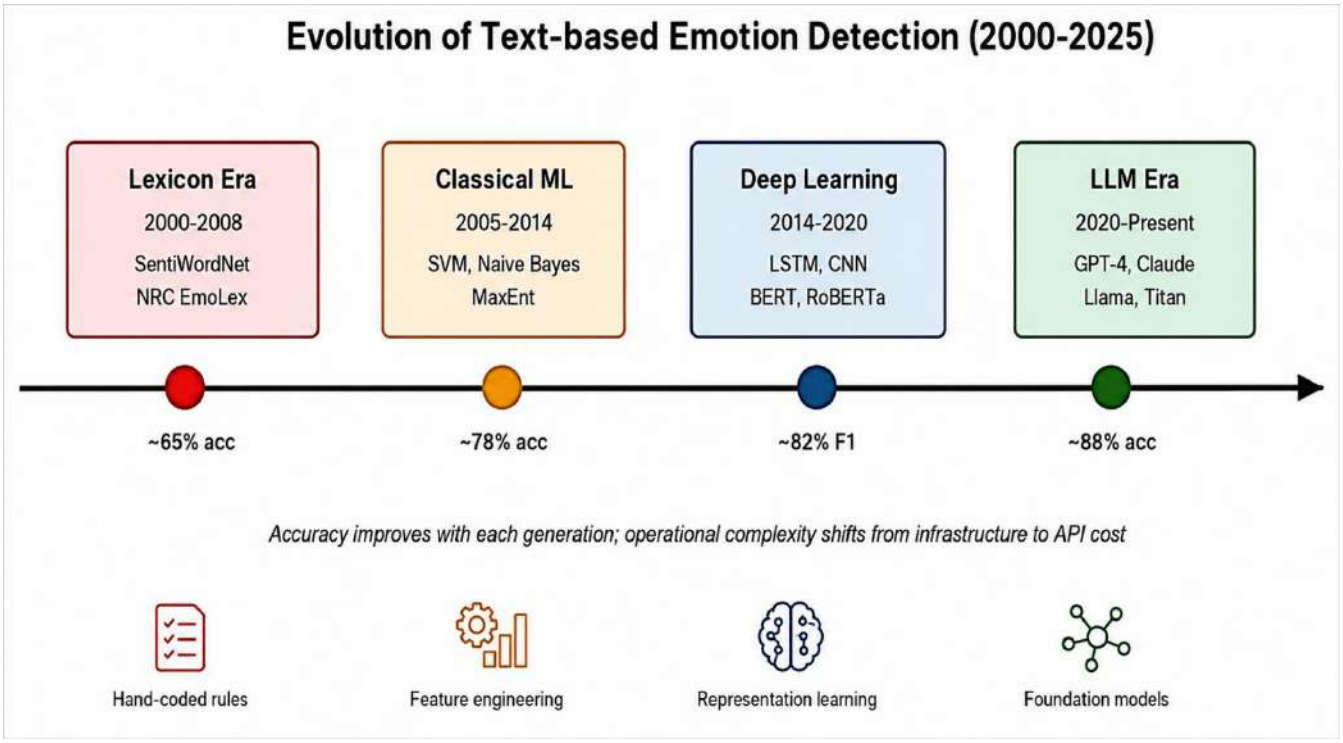

Figure 1: Four generations of text-based emotion detection methods and their typical accuracy bands.

A. How this paper is organised

This paper is meant for readers who are new to the area and do not necessarily have a strong AI background. The focus stays on emotion detection itself, on the data, on what each generation of methods can and cannot do. Section II defines the problem more carefully. Section III covers the early lexicon-based methods. Section IV moves into classical machine learning. Section V looks at deep learning and the move to transformer encoders. Section VI covers the recent shift to large language models. Section VII surveys the benchmark datasets. Section VIII is about evaluation metrics. Section IX is the messy part, the open challenges that nobody has really solved. Section X concludes and points to the implementation work that follows this survey.

Two related areas come up often enough that it is worth saying upfront they are not covered here. The first is multimodal emotion recognition, which combines text with audio, facial expressions, or even physiological signals. The second is speech emotion recognition that works on raw audio without a transcript. Both are interesting but they would each need their own survey. The focus of this one is purely text.

II. WHAT EMOTION DETECTION IS

A. Sentiment versus emotion

Two terms get thrown around a lot here, sometimes as if they meant the same thing. They do not. Sentiment analysis is the older, simpler version. It classifies text along a polarity axis. Positive or negative, sometimes with a neutral class added. That is useful for some things, but it has obvious limits. It tells you a review is negative but not whether the customer is upset, scared, or just disappointed [1]. Emotion detection is the finer-grained version. Instead of two or three polarity buckets, it tries to assign one of several emotion labels. Joy. Sadness. Anger. Fear. Surprise. Disgust. The label set varies from study to study but the idea is always the same. Pick the actual feeling, not just whether it is good or bad [2].

This distinction matters in practice. Two pieces of text with the same negative polarity can need completely different responses. A customer who is angry needs the temperature lowered. A customer who is fearful needs reassurance. A customer who is disgusted has probably already decided to leave. Polarity by itself cannot tell them apart.

B. Discrete versus dimensional models

There has been a long debate in psychology about how to model emotions, and that debate has carried over into computer science. The discrete view says there are a small fixed set of basic emotions that all humans recognise across cultures. Paul Ekman is the name most associated with this, and his canonical list is anger, disgust, fear, joy, sadness, and surprise [3]. Robert Plutchik proposed something similar but with eight categories arranged on a wheel, allowing for intensity gradations [4]. The competing view is the dimensional model. It says emotions are not discrete categories at all, but points in a continuous space, usually two-dimensional, with valence (pleasant to unpleasant) on one axis and arousal (calm to excited) on the other. Russell's circumplex model is the most-cited example [5].

Most computational work goes with the discrete view. Discrete labels map cleanly to a classification task, and human annotators find it easier to agree on a category than on a point in a continuous space. Nearly every dataset and shared task in the field uses a discrete label set.

C. How many labels?

Once you decide to go discrete, the next question is how many emotion labels you actually want. This is one of the most consequential design decisions you can make. Some early studies used as few as four. Most modern work uses six or seven, usually the Ekman six with sometimes a neutral class added on top. At the high end, the GoEmotions dataset

from Google [6] used twenty-eight labels (twenty-seven fine-grained emotions plus neutral), which is closer to how a human might describe emotional nuance in everyday writing. Table 1 lists the most common taxonomies and where they show up in the literature.

Before the table, a word about the tradeoff. More labels give you finer output, which sounds good. But more labels also mean annotators disagree more often, training data per class shrinks, and the boundaries between similar emotions become genuinely ambiguous. Telling annoyance from anger reliably is hard even for humans. Most production systems land at seven labels because that is the spot where accuracy and granularity balance out reasonably well.

Table 1: Common emotion taxonomies used in the literature, with typical use cases

Taxonomy	Number of labels	Example labels	Used in
Polarity (sentiment)	2 or 3	positive, negative, neutral	Movie reviews, product reviews
Ekman basic	6	anger, disgust, fear, joy, sadness, surprise	ISEAR (uses 7 with guilt/shame), SemEval-2018
Ekman + neutral	7	Ekman six plus neutral	Many production systems, chat data
Plutchik	8	joy, trust, fear, surprise, sadness, disgust, anger, anticipation	NRC emotion lexicon
EmoContext	4	happy, sad, angry, others	SemEval-2019 Task 3
GoEmotions	28	admiration, amusement, anger, annoyance, neutral, etc.	Google Reddit-based dataset

Looking at this table, the pattern is straightforward. Smaller schemes are easier to label and easier to learn, but they lose nuance. Larger schemes capture more nuance but they are harder to label consistently and harder to learn from limited data. Seven is a comfortable middle. The Ekman basic six dominates the older work because Ekman's psychology framing was the obvious starting point. EmoContext used only four labels because the focus was on dialogue context rather than fine-grained emotion. GoEmotions stretched the label set to twenty-eight because the goal was to study just how granular you could push the categories before annotation broke down. The cost of that granularity shows up in the agreement numbers we will see later in Section VIII.

D. The basic pipeline

Before getting into the methods, it is worth showing the basic shape of an emotion detection system, because all the methods we are about to discuss fit into roughly the same skeleton. Figure 2 sketches the common pipeline.

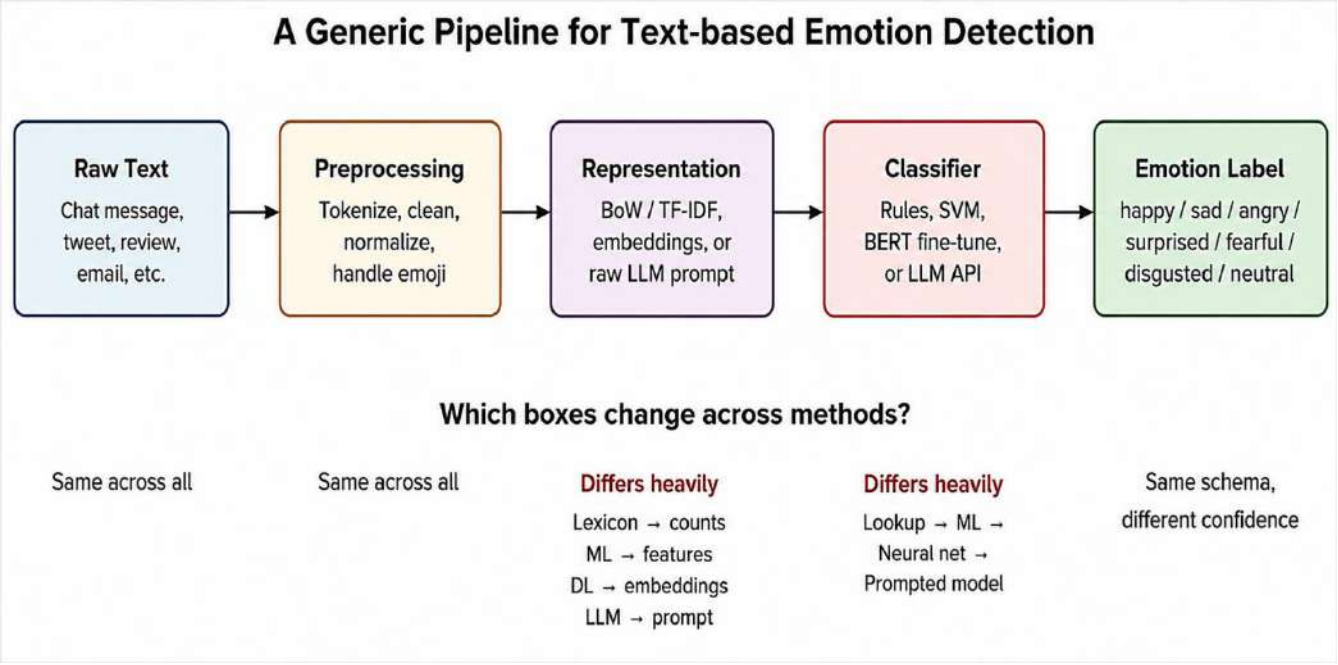

Figure 2: A generic emotion detection pipeline, with the stages that differ most across methods highlighted.

The first two stages are almost the same regardless of method. Take in raw text, clean it up. The last stage is also similar. Pick a label from a fixed set. What changes is the middle. How the text gets represented and how the classifier itself works. In the lexicon era these were word-count features and a rule-based picker. In classical ML they were engineered features and an SVM or Naive Bayes model. In the deep learning era, they were learned embeddings and a

neural network. In the LLM era the representation is whatever the model attends to internally, and the classifier is a prompted language model.

III. THE LEXICON ERA

A. The idea, simply put

The first wave of emotion detection systems took a direct approach. Build a list of words associated with each emotion. Count how many words from each list appear in a piece of text. Pick the emotion with the highest count. This is sometimes called the bag-of-words approach. It dominated through the mid-2000s and is still around today in lightweight settings.

Interestingly, the work that put this approach on the map came from sentiment analysis rather than emotion detection. Pang, Lee and Vaithyanathan studied movie reviews and showed that a simple word-counting classifier could reach around 77 to 83 percent accuracy on binary polarity, with SVMs at the top of that range [7]. Turney took a related approach using pointwise mutual information from web data to derive sentiment orientation without any labels at all [8]. Extending the idea to multiple emotion categories was a natural next step.

B. The lexicons that mattered

A handful of lexicons became standard during this period and they are still cited in modern work. Table 2 summarises the main ones, with sizes verified against the original publications.

Table 2: Key emotion lexicons developed during the lexicon era.

Lexicon	Size	Labels assigned	Notes
SentiWordNet 3.0 [9]	~117,000 synsets	Positive / negative / objective scores	Built on top of WordNet, still used as a feature source
NRC EmoLex [10]	14,182 words	Plutchik's 8 emotions plus polarity	Crowdsourced, most-cited resource in the field
LIWC2015 [11]	~6,400 words	Emotional and psychological categories	Originally for psychology research, widely reused in NLP
WordNet-Affect [12]	1,309 words (~911 synsets) in core	Affective labels on WordNet entries	Larger counts in the literature include relational expansions
AFINN-111 [13]	2,477 words	Integer valence from -5 to +5	Small, simple, popular for quick prototyping

A few things about in table 2 are worth pointing out. The sizes vary by a lot, from a couple of thousand entries up to over a hundred thousand. That matters because coverage of slang and informal vocabulary depends directly on how big the lexicon is. The labelling schemes are also different. SentiWordNet and AFINN focus on polarity. EmoLex, LIWC, and WordNet-Affect cover actual emotion categories, which makes them more useful for emotion detection proper. The newest of these is over a decade old now, which tells you something about the maturity of this whole approach. Lexicons are not really an area of active research anymore, but the resources are still used as features inside larger systems.

C. What lexicon methods get right and wrong

The strengths of lexicon methods are easy to list. They are simple to implement. They need no training data. They are very fast at inference time. The decisions they make are interpretable, in the sense that you can point at the exact words that triggered the classification. For some applications, especially ones with low traffic and a need for transparent decisions, that is genuinely enough.

The weaknesses are also easy to list, and they are harsh. Lexicon methods are blind to context. The phrase "not happy" gets classified as happy because the word happy is in the joy lexicon. Negation handling needs extra rules and those rules never quite cover every case. Sarcasm goes completely undetected, because sarcastic writing uses positive vocabulary to express negative feelings. Slang, abbreviations, and the typos that dominate chat text are all problems. And there is no real notion of intensity. "A little annoyed" and "absolutely furious" each contribute one anger word and get treated the same. Practical accuracy on multi-class emotion detection with a pure lexicon usually falls in the 50 to 70 percent range depending on the dataset and the number of classes.

IV. THE CLASSICAL MACHINE LEARNING ERA

A. Why ML helped

The next big step was treating emotion detection as a supervised classification problem. Instead of writing lexicons by hand, you labelled a corpus of texts with emotion categories and trained a classifier to learn the patterns on its own. The classifier could pick up on combinations of words, on context features, and on patterns a hand-built lexicon would never include.

Alm, Roth and Sproat were among the first to study sentence-level emotion classification carefully [13]. They worked with annotated children's stories and trained classifiers including SVMs. The work was useful but it surfaced a problem that has haunted the field ever since. Class imbalance. Some emotions are much more common in natural text than others, and classifiers tend to over-predict the common classes and under-predict the rare ones. Disgust and fear are usually the rarest classes in any general-purpose dataset, and their classification accuracy lags behind even today.

B. Classifiers and features

Classical ML emotion detection followed a recipe that became standard. Tokenise and clean the text. Extract features. Train a classifier. The interesting work was almost all in the feature extraction step. Table 3 lists the main families of classifiers used during this era. The accuracy bands shown are for binary polarity on the Pang et al. movie review benchmark [7], because that is the cleanest comparison point for which numbers across classifiers are directly available. On finer-grained multi-class emotion datasets like ISEAR, all of these classifiers tend to score five to fifteen points lower because the task is harder.

Table 3: Classifiers most commonly used in the classical ML era of emotion detection.

Classifier	How it works (briefly)	Typical acc. on polarity	Notes
Naive Bayes	Assumes features are independent given the class, then uses Bayes' rule	77-82%	Very fast to train, simple baseline
Support Vector Machine	Finds the hyperplane that best separates classes	78-83%	The strongest single classifier of the era
Maximum Entropy	Logistic regression generalised to multi-class	77-81%	Common in NLP, similar shape to SVM
Random Forest	Ensemble of decision trees with feature bagging	75-80%	Less common but robust to feature scale
Gradient Boosted Trees	Boosted ensemble of decision trees	76-82%	Strong with engineered features

A few patterns stand out. SVMs were generally the strongest single classifier, with both gradient boosted trees and SVMs landing in the high seventies and low eighties. Naive Bayes was almost always weaker but it was so fast to train that it remained the go-to baseline. Maximum entropy classifiers, which are mathematically just logistic regression for multi-class, were popular in NLP because they fit neatly with the kind of sparse text features people were using. The range from the weakest to the strongest in this table is only about five percentage points, which tells you the choice of classifier did matter, but not as much as choosing good features. That feature engineering, which used bag-of-words and TF-IDF as the base, augmented with n-grams, lexicon counts, negation handling, punctuation features, and emoticon features, was where most of the practical effort went.

C. The limits of classical ML

Despite the gains, classical ML methods had two structural problems. The first was that they relied on features the human had to design. If your feature set did not include some relevant pattern, the model could not learn it. Sarcasm, irony, idioms, figurative language. All of these use words in ways that surface features cannot capture. The second problem was that the methods had no real understanding of word meaning. The word "furious" and the word "livid" were treated as completely unrelated tokens. For a classifier to know they meant similar things, it had to learn that from the training data, which it could only do if both words appeared often enough in the labelled set. Those two limits set the stage for the deep learning era.

V. THE DEEP LEARNING ERA

A. Word embeddings

The first real breakthrough of the deep learning era was something called word embeddings. The idea is simple. Instead of treating each word as an opaque token, you represent words as vectors in a high-dimensional space, where the vectors are learned from large amounts of unlabelled text. Words with similar meanings end up close to each other in that space. The original work that made embeddings practical was Word2Vec by Mikolov and his collaborators [14]. GloVe from Stanford followed shortly afterwards [15]. These were not emotion-specific tools, but they immediately changed how emotion detection was done, because suddenly the classifier could see that "furious" and "livid" were related, even if only one of them appeared in the training data.

B. Recurrent and convolutional networks

Embeddings gave us better word representations. Recurrent neural networks gave us a way to actually read a sequence of words in order. Long Short-Term Memory networks (LSTMs) and their slightly newer cousins, Gated Recurrent Units (GRUs), were the dominant architecture for text classification through the mid-2010s [16]. For emotion detection, the typical setup was a word embedding layer feeding into a bidirectional LSTM, with the LSTM output passed through a small classifier head. Attention mechanisms were often added on top to let the model focus on the most informative words.

EmoContext at SemEval-2019 [17] was particularly important. It focused on three-turn dialogues and proved that prior conversational turns carry useful signal for classifying the latest message. Mid-tier LSTM-with-attention systems on that task reached F1 scores in the 0.70 to 0.72 range, with the best overall system reaching about 0.7959. That gap tells you something. The basic recurrent architecture had a ceiling. Beyond that ceiling, you needed a fundamentally different architecture. Convolutional networks adapted from image classification were also competitive in this period, particularly Kim's CNN-for-text [18], but they never decisively beat LSTMs on emotion tasks.

C. Transformers and BERT

The next big shift was the transformer architecture, introduced by Vaswani and his collaborators in 2017 [19]. Transformers replaced recurrence with a self-attention mechanism that lets each token in a sequence directly attend to every other token, rather than passing information

through a sequential chain. The architecture parallelises well, trains faster, and captures long-range dependencies more reliably than RNNs.

BERT (Bidirectional Encoder Representations from Transformers) from Devlin and his collaborators turned the transformer into a general-purpose NLP backbone [20]. BERT is pre-trained on huge amounts of unlabelled text using two self-supervised objectives, and then fine-tuned on a specific downstream task with a small labelled dataset. For emotion detection, this was a real change.

On standard emotion datasets the picture is dataset-dependent, but fine-tuned BERT and its descendants generally outperform the LSTM-era systems. On EmoContext, BERT-based systems reach F1 in the 0.78 to 0.80+ range. On GoEmotions, however, the original BERT-base baseline reported by the dataset authors reached only macro F1 of 0.46 across all twenty-seven emotion categories, illustrating how much harder fine-grained classification is than the headline numbers from coarser datasets suggest [6]. In Table 4 we compares the main transformer encoder variants by their architectural footprint.

Table 4: Transformer encoder models commonly used for emotion classification.

Model	Parameters	Notes
BERT-base [20]	110M	The standard baseline, widely fine-tuned
BERT-large [20]	340M	Better accuracy, much heavier
RoBERTa-base / large [1]	125M / 355M	Better-trained BERT variant, common in current work
DistilBERT [30]	66M	40% smaller, retains 97% of BERT's understanding, ~60% faster
ALBERT-base [6]	12M	Parameter-sharing trick, small footprint
DeBERTa-base / large [15]	134M / ~400M	Disentangled attention, among the strongest encoders

Table 4 shows two useful things. First, the parameter counts tell you about operational cost. A DistilBERT at 66 million parameters can run on modest hardware. A DeBERTa-large model at around 400 million parameters needs a GPU for any kind of serious throughput. Second, the headline accuracy differences between BERT, RoBERTa, and DeBERTa on emotion tasks are real but small, often only one or two points. The choice of model in practice often comes down to whether you have the infrastructure for a bigger one, not whether the bigger one is decisively more accurate. That said, the operational reality of fine-tuning and serving any of these still needs GPU hardware, model versioning, monitoring, and an MLOps stack. For a small team that just wants emotion detection as one feature inside a larger product, that overhead is genuinely hard to justify. This is the gap that the next generation of methods would step into.

VI. THE LARGE LANGUAGE MODEL ERA

A. What changed

Large language models like GPT-3, GPT-4, Claude, and Llama are transformer-based models pre-trained on much larger amounts of text than BERT-era models, with many more parameters. Their key property for our purposes is that they can perform a wide range of NLP tasks without any task-specific fine-tuning, just by being prompted in natural language. This is called zero-shot prompting, or few-shot if you include some examples in the prompt.

For emotion detection, the effect is large. No labelled training dataset. No fine-tuning pipeline. No GPU-backed inference infrastructure. You describe the task in a prompt, send a message to the API, and get back a classification. Wang and his collaborators were among the first to systematically study ChatGPT on sentiment tasks, finding the model competitive on many benchmarks though not always better than fine-tuned baselines [21]. Zhao and his collaborators examined ChatGPT's emotional dialogue capabilities and reached similar conclusions [22].

B. The main LLM families

Three families of LLMs dominate the current landscape for emotion detection work. OpenAI's GPT line, including GPT-3.5 and GPT-4, has been extensively benchmarked on affective tasks. Anthropic's Claude family, which includes Claude 3 Haiku, Sonnet, and Opus, has shown strong empirical performance on empathy and emotion benchmarks. Chen and his collaborators introduced the EmotionQueen benchmark and reported that Claude 2 and Llama-2-70B were among the strongest performers across four empathy-related sub-tasks [23]. Meta's Llama family is available in open-weights form, which allows fully internal self-hosted deployments. Amazon Titan, Cohere Command, and Mistral round out the commonly evaluated set

In the below table 5, it reflects something important about the current state of the field and gives a comparative summary. Choosing an LLM for emotion detection is not just a matter of picking the most accurate one. The access mode matters too. If your application cannot send data to a third party for compliance reasons, you need open-weights models that can be self-hosted, which mostly means Llama or Mistral. If you want managed infrastructure but flexibility across providers, AWS Bedrock gives you Claude, Titan, and several others through one interface.

Table 5: Major LLM families relevant to emotion detection, with their typical access modes.

Model family	Provider	Access mode	Notes on emotion-related performance
GPT-3.5 / GPT-4	OpenAI	API only	Strong on broad NLP tasks, often used as baseline in LLM emotion studies
Claude 3 (Haiku/Sonnet/Opus)	Anthropic	API and AWS Bedrock	Claude 2 was a top performer on EmotionQueen empathy benchmark [23]
Llama 3	Meta	Open weights, self-hostable	Llama-2-70B was competitive with Claude 2 on EmotionQueen [23]
Titan Text	Amazon	AWS Bedrock	Solid baseline, common managed-cloud fallback
Command R	Cohere	API and Bedrock	Good multilingual coverage
Mistral / Mixtral	Mistral AI	API and open weights	Strong for size, popular self-hosted

Cost varies by an order of magnitude across these options, so the right pick depends as much on the engineering and business constraints as on headline accuracy.

C. Prompt engineering and chain-of-thought

Getting good emotion classification out of an LLM is not just a matter of asking it nicely. Prompt design has a measurable effect on accuracy. A handful of techniques have become standard. Role priming tells the model who it is supposed to be, for example a computational linguist with affective psychology training. Explicit label definitions help the model understand exactly which emotion categories you want it to use. Output format constraints, usually a strict JSON schema, eliminate stray prose around the classification. Few-shot examples, usually two or three per class, work especially well for edge cases at the boundaries between similar emotions.

Chain-of-thought prompting, where the model is asked to reason step by step before giving its final answer, was shown by Fei and his collaborators to give measurable gains on implicit sentiment reasoning [24]. Their three-hop reasoning framework asked the model to first identify the aspect, then the opinion expression, and only then the sentiment polarity, and outperformed direct prompting baselines. The same general pattern is reported across other affective subtasks. The tradeoff is that chain-of-thought responses become longer and slower, and the cost per call goes up. For high-volume applications, whether the accuracy bump is worth the extra cost is genuinely open.

D. Reported accuracy and the honest picture

It is tempting to claim that LLMs have decisively beaten fine-tuned encoders on emotion detection. The honest picture is messier. On many sentiment and emotion benchmarks in zero-shot or few-shot settings, LLMs reach accuracy that is competitive with fine-tuned baselines but does not always exceed them. Wang and his collaborators reported that ChatGPT typically underperformed fine-tuned BERT on standard sentiment benchmarks by several points in pure zero-shot conditions [21]. Where LLMs really shine is in tasks that need broader reasoning, contextual understanding, and handling of implicit emotional cues, which is what the EmotionQueen and EmoBench benchmarks were designed to measure [23][25]. Numbers from a representative implementation built around Claude 3 Sonnet on Amazon Bedrock, evaluated on a balanced seven-class benchmark of 1,000 chat messages, reached 88.4 percent overall accuracy [26]. Whether that number generalises to other datasets depends heavily on the prompt design and the test distribution.

In below Figure 3 it puts the four eras side by side on the same accuracy axis. The ranges shown are typical for the main benchmarks each generation was evaluated on, with the understanding that fine-grained datasets like GoEmotions push the numbers lower across the board.

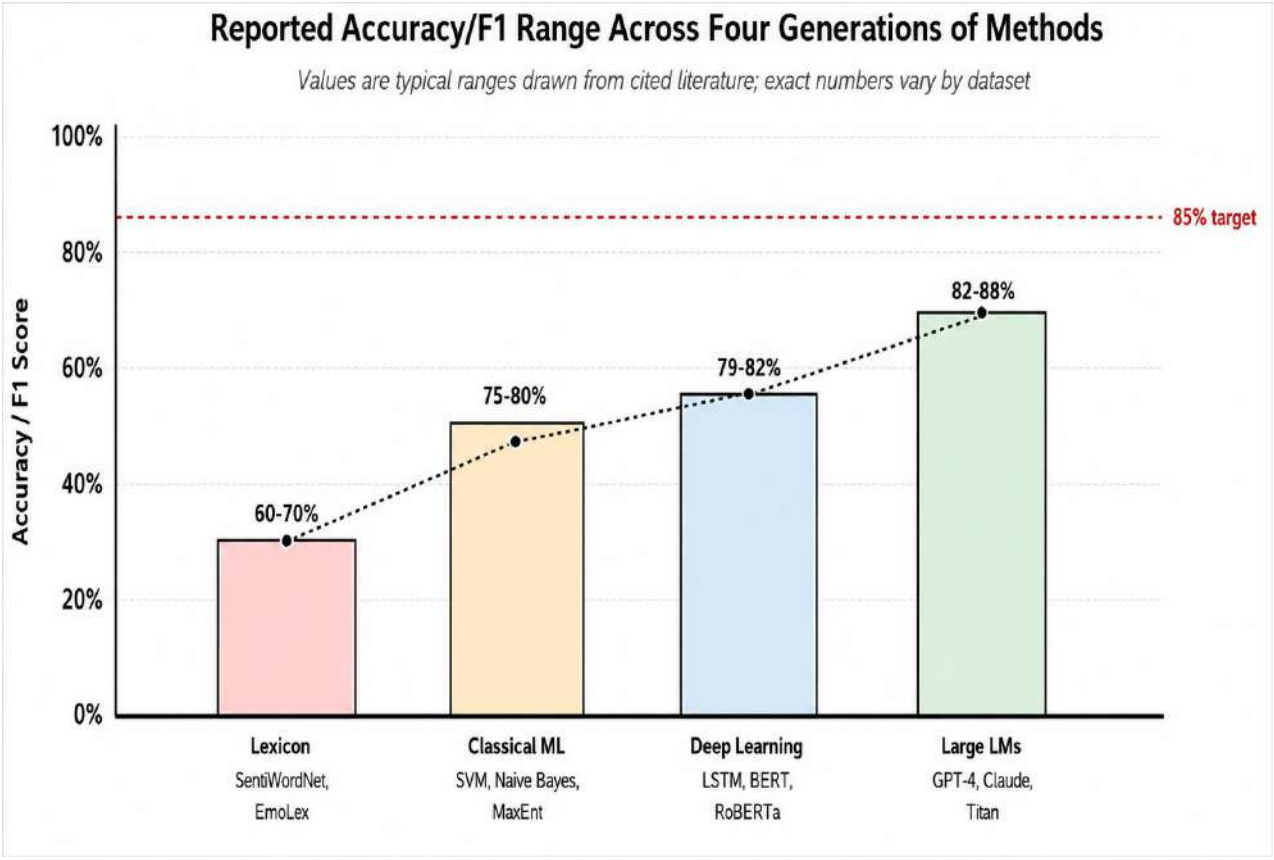

Figure 3: Typical accuracy or F1 ranges across the four method generations on common emotion benchmarks

E. Practical tradeoffs

LLMs are not a free win. API calls cost money per request, and at high volumes the cost can exceed what self-hosting a fine-tuned model would cost. Inference latency for LLM calls is usually higher than for a tuned encoder, often 100 to 400 milliseconds per call versus 20 to 50 milliseconds for a small fine-tuned model. The non-determinism of LLM outputs means the same input can occasionally produce slightly different outputs, which complicates testing. Vendor lock-in is real.

That said, for most production applications under roughly half a million requests per day, the operational simplicity of using a hosted LLM through a managed API outweighs these concerns. No training data needed, no GPU infrastructure to maintain, and strong accuracy on most tasks. Table 6 summarises the tradeoffs across the four generations.

Table 6: Comparison across the four generations of emotion detection methods.

Generation	Typical performance	Training data needed	Infrastructure	Main weakness
Lexicon-based	50-70% (multi-class)	None (just a dictionary)	Trivial, runs anywhere	No context, no sarcasm handling
Classical ML	75-83% on polarity	Thousands of labelled examples	Modest, CPU is enough	Shallow features, no semantics
Deep learning (BERT)	0.78-0.82 F1 on coarse; 0.46 macro on GoEmotions [6]	Thousands of labelled examples plus pre-trained model	GPU for fine-tuning and serving	MLOps overhead, dataset-sensitive
Large language models	Competitive zero-shot; benchmark-dependent	None (zero-shot) or few examples	Managed API or self-hosted	Per-request cost, latency, non-determinism

Each generation pushed accuracy up. Each generation also shifted where the operational pain lived. Lexicons had no real ops cost but they were inaccurate. Classical ML added a training pipeline but kept things on CPU. Deep learning brought GPU infrastructure into the picture. LLMs moved the GPU infrastructure to someone else's cloud but introduced per-request cost and reproducibility challenges. There is no one-size-fits-all answer, and the right choice depends on volume, budget, and the level of accuracy that the application actually needs.

VII. BENCHMARK DATASETS

Good benchmarks are what let the field measure progress. A handful of datasets have become standard reference points. Table 7 lists the main ones, with sizes verified against the original publications and dataset documentation.

Table 7: Standard emotion detection benchmark datasets and their characteristics.

Dataset	Size	Labels	Domain
ISEAR [27]	7,666 sentences, ~2,900 respondents from 37 countries	7 emotions (Ekman 6 + guilt + shame)	Survey responses, cross-cultural
SemEval-2018 Task 1 E-c [28]	~10,983 tweets (multi-label)	11 emotions	Twitter
EmoContext [17]	~38,000 three-turn dialogues	4 (happy, sad, angry, others)	Chat dialogues
GoEmotions [6]	58,009 comments	28 labels (27 emotions + neutral)	Reddit
DailyDialog [29]	13,118 dialogues	7 emotions + 4 dialogue acts	Everyday conversations
MELD [30]	~13,000 utterances from 1,433 dialogues	7 emotions	TV dialogues (Friends)
EmoBank [31]	10,062 sentences	Dimensional (V-A-D)	Mixed sources

A few observations about in table 7 are worth making. The size range is huge, from under eight thousand examples up to almost sixty thousand. That matters because supervised methods need enough examples per class to learn each one reliably. A 7,666-example dataset with seven labels averages out to barely a thousand examples per class, which is on the low side. The domains are also wildly different. Survey narratives, tweets, three-turn chats, Reddit comments, TV dialogues. Each has its own writing style and emotional register, and a model trained on one does not automatically work on another.

In practice, most researchers report on multiple datasets at once. ISEAR for general-purpose emotion classification. SemEval-2018 for social media work. EmoContext when conversation context matters. GoEmotions when fine-grained labels are needed. DailyDialog when natural conversation is the focus. Reporting on multiple benchmarks is the norm in published work because it shows whether a method is genuinely robust or just well-tuned for one dataset.

VIII. EVALUATION METRICS

How emotion detection systems get evaluated has converged on a small set of metrics. Understanding what they actually mean is essential for reading any of the literature.

A. Accuracy, precision, recall, F1

Accuracy is the proportion of test examples that the system classifies correctly. It is the most intuitive metric and the one most often reported in informal contexts. The problem with accuracy is that it can be misleading when the classes are imbalanced. A system that always predicts the majority class can have a high accuracy number while being useless. Emotion datasets are almost always imbalanced. Neutral and happy are usually overrepresented while disgust and fear are rare. So, accuracy by itself is rarely enough.

Precision and recall are computed per class. Precision asks, of the messages predicted as anger, what fraction were actually anger. Recall asks, of the messages that were actually anger, what fraction did the system label correctly. The F1 score is the harmonic mean of precision and recall, and it summarises the two into a single number. When the literature reports an F1 score on a multi-class emotion task, it is usually the macro-averaged F1, which is the unweighted mean of the per-class F1 scores. Macro-F1 is the right metric when you care about performance across all classes, including the rare ones.

B. Confusion patterns

A confusion matrix shows, for each true class, how many examples got predicted into each of the predicted classes. It is the most informative single view of a classifier's behaviour. In emotion detection, the predictable confusions are anger versus disgust, sadness versus fear, and surprise versus joy. Almost every published study reports a confusion pattern along those lines. In below Figure 4 visualises these typical confusion patterns.

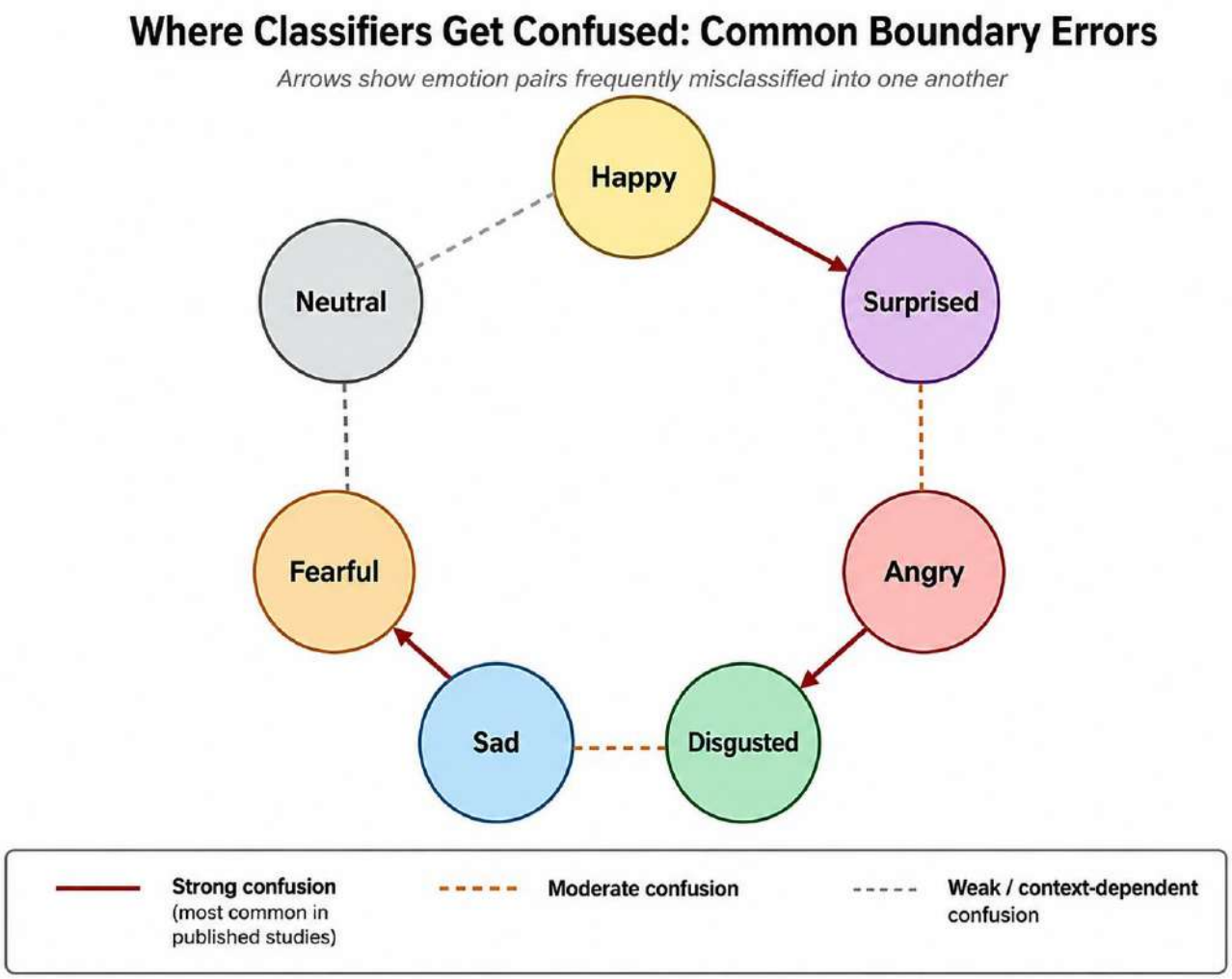

Figure 4: Common emotion confusion patterns reported across the literature. Solid arrows mark strong, frequently reported confusions

The strongest confusions cluster around emotion pairs that share vocabulary or surface markers. Anger and disgust both use rejection language, both often use intense punctuation, both can use profanity. Sadness and fear both use low-energy, worry-tinted language and often share words around loss or threat. Surprise and joy both use exclamations and positive intensifiers. These are the same places where human annotators most often disagree during labelling, so a chunk of the residual error in any system is genuine linguistic ambiguity rather than a model weakness.

C. Inter-annotator agreement

This is not a model metric but a dataset metric, and it puts a ceiling on what any model can be expected to achieve. Inter-annotator agreement measures how consistently human labellers agree on the emotion of a given text. It is usually reported as Cohen's kappa for two annotators or Fleiss' kappa for three or more. The values vary widely with task granularity. For coarse two or three-way sentiment, kappa scores of 0.6 to 0.8 are common, which is called substantial agreement. For finer-grained emotion taxonomies, agreement drops sharply. GoEmotions, with its twenty-eight labels, reports an average Cohen's kappa of just 0.27 across its emotion categories [6]. This puts a real ceiling on accuracy. A model that reports 90 percent on a dataset where annotators only agree two-thirds of the time should be read carefully, because it may have overfit to one particular labelling style.

IX. OPEN CHALLENGES

A. Sarcasm and irony

Sarcasm uses positive vocabulary to express negative feelings, which directly contradicts the surface signal that emotion classifiers rely on. "Oh great, another Monday" looks positive but means the opposite. Modern LLMs handle sarcasm better than earlier methods because they have been trained on huge amounts of sarcastic text and can incorporate conversational context, but sarcasm remains a category where even LLMs make consistent errors.

B. Class imbalance

Real-world emotional text is dominated by neutral and positive messages. Anger, fear, and disgust are rare in absolute terms even though they are often the most operationally important categories to detect. This imbalance affects every supervised method. Common mitigations are class weighting during training, oversampling minority classes, or synthetic data augmentation. None of them fully solves the problem.

C. Cultural, linguistic and code-mixed text

Emotions are not expressed identically across cultures. The vocabulary, the level of directness, and the use of emotional language in formal versus informal settings all vary considerably. Most emotion datasets are dominated by English and by Western cultural patterns. Models trained predominantly on such data perform worse on text from other cultures even when translated to English. Related to this, in many parts of the world, including India, written informal text often mixes languages. A single chat message can contain Hindi words written in Latin script, English words, and emoji, all in one sentence. This kind of writing breaks tokenisers, breaks lexicons, and confuses models trained on monolingual corpora. Research on code-mixed emotion detection is growing but the resources are still limited.

D. Evaluation reliability for LLMs

Because LLM outputs are non-deterministic and the models change every few months as providers release new versions, reproducing emotion detection results across papers is genuinely harder than it was in the BERT era. There is active discussion in the community about how to standardise LLM evaluation protocols, what to do about model version drift, and how to make benchmarks more robust to the rolling updates that LLM provider's ship.

X. CONCLUSION

Emotion detection has come a long way from the bag-of-words classifiers of the early 2000s. Each generation of methods, from lexicons to classical machine learning to deep learning encoders to large language models, has shifted the accuracy frontier upward and shifted where the operational pain lives. The most recent shift, to LLM-based methods accessed through managed APIs, is particularly consequential for practitioners. It collapses the gap between research prototypes and production systems.

The choice of method should always follow the requirements of the application. For low-volume, transparency-critical applications, lexicon methods may still be the right answer. For very high volumes with tight latency budgets, fine-tuned BERT-style encoders remain competitive on cost. For applications that need strong accuracy with minimum operational burden, LLM-on-managed-API is increasingly the default. Real applications often combine methods, using a fast first-pass classifier with an LLM fallback for ambiguous cases. The open challenges that remain (sarcasm, class imbalance, multilingual coverage, code-mixed text, and evaluation reliability under LLM drift) are all areas where there is meaningful room for new contributions. The move to LLM-based methods has changed the shape of the open problems rather than eliminating them.

As a direct extension of this survey, the authors are developing a production reference implementation of an emotion detection REST service based on the LLM-on-managed-cloud approach identified here. That implementation paper presents the end-to-end system architecture, prompt engineering details, accuracy and latency evaluation, and deployment patterns for an AWS Bedrock-backed service [26]. Together, this survey and the accompanying implementation paper are meant to give readers both the theoretical grounding and the practical engineering details needed to build emotion detection systems that work.

REFERENCES

[1] Liu, "Sentiment analysis and opinion mining," *Synthesis Lectures on Human Language Technologies*, vol. 5, no. 1, pp. 1–167, May 2012. Available from: https://aclanthology.org/anthology-files/anthology-files/pdf/J/J14/J14-2009.pdf

[2] S. Poria, E. Cambria, R. Bajpai, and A. Hussain, "A review of affective computing: From unimodal analysis to multimodal fusion," Information Fusion, vol. 37, pp. 98–125, Sep. 2017. Available from: https://www.sciencedirect.com/science/article/abs/pii/S1566253517300738

[3] P. Ekman, "An argument for basic emotions," Cognition and Emotion, vol. 6, no. 3–4, pp. 169–200, May 1992. Available from: https://doi.org/10.1080/02699939208411068

[4] R. Plutchik, "The nature of emotions," American Scientist, vol. 89, no. 4, pp. 344–350, Jul. 2001. Available from: https://www.jstor.org/stable/27857503

[5] J. A. Russell, "A circumplex model of affect," Journal of Personality and Social Psychology, vol. 39, no. 6, pp. 1161–1178, Dec. 1980. Available from: https://doi.org/10.1037/h0077714

[6] Demszky, D. Movshovitz-Attias, J. Ko, A. Cowen, G. Nemade, and S. Ravi, "GoEmotions: A dataset of fine-grained emotions," in Proc. ACL, Online, 2020, pp. 4040–4054. Available from: https://aclanthology.org/2020.acl-main.372/

[7] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? Sentiment classification using machine learning techniques," in Proc. EMNLP, Philadelphia, PA, USA, 2002, pp. 79–86. Available from: https://aclanthology.org/W02-1011.pdf

[8] P. D. Turney, "Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews," in Proc. ACL, Philadelphia, PA, USA, 2002, pp. 417–424. Available from: https://aclanthology.org/P02-1053.pdf

[9] S. Baccianella, A. Esuli, and F. Sebastiani, "SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining," in Proc. LREC, Valletta, Malta, 2010, pp. 2200–2204.

[10] S. M. Mohammad and P. D. Turney, "Crowdsourcing a word-emotion association lexicon," Computational Intelligence, vol. 29, no. 3, pp. 436–465, Aug. 2013. Available from: https://doi.org/10.1111/j.1467-8640.2012.00460.x

[11] J. W. Pennebaker, R. L. Boyd, K. Jordan, and K. Blackburn, "The development and psychometric properties of LIWC2015," Technical Report, University of Texas at Austin, Austin, TX, USA, 2015.

[12] Strapparava and A. Valitutti, "WordNet-Affect: An affective extension of WordNet," in Proc. LREC, Lisbon, Portugal, 2004, pp. 1083–1086. Available from: https://tinyurl.com/3j7p7faw

[13] O. Alm, D. Roth, and R. Sproat, "Emotions from text: Machine learning for text-based emotion prediction," in Proc. HLT/EMNLP, Vancouver, BC, Canada, 2005, pp. 579–586. Available from: https://aclanthology.org/H05-1073.pdf

[14] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv:1301.3781, Jan. 2013. Available from: https://arxiv.org/abs/1301.3781

[15] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in Proc. EMNLP, Doha, Qatar, 2014, pp. 1532–1543. Available from: https://aclanthology.org/D14-1162.pdf

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. Available from: https://ieeexplore.ieee.org/abstract/document/6795963

[17] Chatterjee, K. N. Narahari, M. Joshi, and P. Agrawal, "SemEval-2019 Task 3: EmoContext contextual emotion detection in text," in Proc. SemEval, Minneapolis, MN, USA, 2019, pp. 39–48. Available from: https://aclanthology.org/S19-2005/

[18] Y. Kim, "Convolutional neural networks for sentence classification," in Proc. EMNLP, Doha, Qatar, 2014, pp. 1746–1751. Available from: https://aclanthology.org/D14-1181.pdf

[19] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, Long Beach, CA, USA, 2017, pp. 5998–6008. Available from: https://cir.nii.ac.jp/crid/1370849946232757637

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. NAACL-HLT, Minneapolis, MN, USA, 2019, pp. 4171–4186. Available from: https://aclanthology.org/N19-1423/

[21] Z. Wang, Q. Xie, Y. Feng, Z. Ding, Z. Yang, and R. Xia, "Is ChatGPT a good sentiment analyzer? A preliminary study," arXiv:2304.04339, Apr. 2023. Available from: https://arxiv.org/abs/2304.04339

[22] W. Zhao, Y. Zhao, X. Lu, S. Wang, Y. Tong, and B. Qin, "Is ChatGPT equipped with emotional dialogue capabilities?" arXiv:2304.09582, Apr. 2023. Available from: https://arxiv.org/abs/2304.09582

[23] Y. Chen, S. Yan, S. Liu, Y. Li, and Y. Xiao, "EmotionQueen: A benchmark for evaluating empathy of large language models," in Findings of ACL 2024, Bangkok, Thailand, 2024, pp. 2149–2176. Available from: https://aclanthology.org/2024.findings-acl.128/

[24] H. Fei, B. Li, Q. Liu, L. Bing, F. Li, and T.-S. Chua, "Reasoning implicit sentiment with chain-of-thought prompting," in Proc. ACL (Short Papers), Toronto, ON, Canada, 2023, pp. 1171–1182. Available from: https://aclanthology.org/2023.acl-short.101/

[25] S. Sabour et al., "EmoBench: Evaluating the emotional intelligence of large language models," arXiv:2402.12071, Feb. 2024. Available from: https://aclanthology.org/2024.acl-long.326.pdf

[26] N. Singh, "Detecting emotions in chat messages with AI: A real-time REST API built on AWS Bedrock," International Journal of Innovative Research in Engineering and Management (IJIREM), to appear, 2026.

[27] K. R. Scherer and H. G. Wallbott, "Evidence for universality and cultural variation of differential emotion response patterning (ISEAR)," Journal of Personality and Social Psychology, vol. 66, no. 2, pp. 310–328, Feb. 1994. Available from: https://psycnet.apa.org/buy/1994-29654-001

[28] S. M. Mohammad, F. Bravo-Marquez, M. Salameh, and S. Kiritchenko, "SemEval-2018 Task 1: Affect in tweets," in Proc. SemEval, New Orleans, LA, USA, 2018, pp. 1–17. Available from: https://aclanthology.org/S18-1001/

[29] Y. Li, H. Su, X. Shen, W. Li, Z. Cao, and S. Niu, "DailyDialog: A manually labelled multi-turn dialogue dataset," in Proc. IJCNLP, Taipei, Taiwan, 2017, pp. 986–995. Available from: https://aclanthology.org/I17-1099/

[30] S. Poria, D. Hazarika, N. Majumder, G. Naik, E. Cambria, and R. Mihalcea, "MELD: A multimodal multi-party dataset for emotion recognition in conversations," in Proc. ACL, Florence, Italy, 2019, pp. 527–536. Available from: https://aclanthology.org/P19-1050/

[31] S. Buechel and U. Hahn, "EmoBank: Studying the impact of annotation perspective and representation format on dimensional emotion analysis," in Proc. EACL, Valencia, Spain, 2017, pp. 578–585. Available from: https://aclanthology.org/E17-2092/